Title of Thesis:

An Architecture for Secure Provisioning and Usage of IOT Services

A thesis

submitted in fulfilment

of the requirements for the degree

of

Masters of Engineering in Software Engineering

at

The University of Waikato

by

Rashid Mustafa

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

Year of submission

2017

Acknowledgements

This thesis is based on research and efforts to consider network security in service oriented architecture. The completion of thesis was not possible without the support of many individuals and organizations. I wish to extend my sincere thanks to all of them.

I am thankful to Dr. Vimal Kumar for his kind support and the guidance that was necessary to complete this thesis.

I must express my thanks to my family and parents for their kind co-operation and encouragement towards the completion of this thesis.

Abstract

The growth in the internet of things plays a central role in the future internet having a common platform for a complex network of computers, wired and wireless sensors, actuators and radio frequency identification devices. Our simplified view of this complex network is of a 'network that provides various services'; let's call it the internet of services. There is though a lack of a platform that can bring together the services offered by the internet of things/internet of services and the future user of these services. In this thesis, an architecture has been proposed for providing secure services to connected users. We propose a layered architecture that has an application layer, a service layer and a sensing-network layer to accommodate the requirements of the internet of services in a timely and effective manner.

In this architecture, data communication security of the internet of services has been taken into consideration. A web security interface has been introduced, which provides security for user credentials. This interface is introduced in the application layer to protect and ensure data communication security using a single sign on. We have proposed a node to node communication security solution after considering the vulnerabilities and threats around connecting a future integrated service. We have designed and evaluated our architecture by performing a range of experiments. These experiments were conducted using database connection round trip time, total execution code round trip time and average query round trip time to compare the services that were designed on different programming platforms.

Table of Contents

Section 01: Introduction

The Internet of Things (IoT) [1] is an emerging technology that has both commercial and economic importance. Many items such as consumer products, industrial components, sensors, food items and other items in daily life may be described as things connected through the internet by powerful data processing and analysing techniques, thus improving our quality of life [2]. The term Internet of Things [3] refers to connecting things like sensors, daily food items and industrial instruments with computing ability. These IoT devices, when connected to a network, can exchange and consume data with minimal human interference. The idea of integrating computers, networks and IoT devices is not new; it has existed for decades. Recent technological market trends integrate IoT devices in a platform that serves customers commercial needs and wants to their satisfaction.

1.1 Historical Perspective

The history of IoT is as old as the internet itself. The first interconnectivity of machines occurred with the invention of circuit switch land-line telephones in the early 19th century [4]. The beginning of the 20th century [4] saw the emergence of early wireless connectivity in the form of radio. Until this time communication mostly meant voice transmission rather than the flow of data, which is a hallmark of the digital communication era. The first significant step towards the internet [5] was made by DARPA with its ARPANET project in the late 1960s, when computers were connected to physical networks to communicate data. It wasn't until the 1980s that similar computer networks began to emerge on a larger scale, which later became the internet [6]. In the early 1990s, mobile phones went through a similar transformation, starting from voice-only communication to later enter the age of digital communication. The progressive innovation in digital technology that started with GSM and GPRS later gave birth to [7] the 3G, 4G/LTE and 5G standards. At the same time as internet and digital communication technologies were improving, [8]hardware was increasingly becoming cheaper, more reliable, and smaller, thus consuming less power. This allowed a lot of functionality and computer power to be packed into smaller devices.

1.2 Future of IoT

Recently, several companies and research organizations have made predictions about the potential impact of the IoT on the internet and the economy during the next decade. Cisco has forecast that there will be more than 24 billion internet–connected devices by 2019

[9]. Morgan Stanley however have projected more optimistic figures of 75 billion networked objects by 2020 [10]. Huawei has raised the stakes even higher and predicted there will be 100 billion IoT connections by 2025 [11]. The McKinsey Global Institute survey shows that the financial impact of the IoT on the international economy will be $3.9 to $11.1 trillion by 2025 [12].

Despite these forecasts, a fundamental challenge revolves around how to put together the many IoT infrastructures that are currently built in silos with a lack of exchange of information at the gateway, service and application layers. The changing IoT market and the availability of new products becomes challenging for developers to support all these products. Due to the growth of IoT technologies and communication protocols, there is a need to develop application programming interfaces and data models to facilitate the services and link them with new technological advancements. We concluded that there are many IoT infrastructures and technologies available currently. For each specific new product, developers have to implement or integrate a gateway to collect device data and control its settings. Many devices in the IoT might add to the mountain of big data, but to generate services with real value to a user requires hiding the IoT complexity by replacing it with an Internet of Services that focuses on and delivers the required business value from the IoT. Generally, a service may be considered as a business activity where a provider grants access to a resource so the requestor can perform a function and get the related benefit. The resource could be a workforce, an industrial system, an information system, consumables, traffic and many other things. In this work, by service we mean a consistent stream of sensor data originating from either one or many, homogeneous or heterogeneous sensors. The Internet of Services (IoS) vision brings together users and and service providers by providing a platform where services could be easily provided, provisioned and consumed

The IoS provides interaction between services, information and IoT devices. Information and services security is a major concern and needs our attention. As security is a broad area, we have taken in to consideration confidentiality, integrity and accessibility. Confidentiality means an assurance of preventing information or services from reaching unauthorised users, while integrity is ensured by maintaining the consistency and trustworthiness of data communication or services communication. Also, we need to ensure that services or data are not modified or changed in transit by unauthorised users.

Availability means that a service using any server or infrastructure is easily available and accessible.

A service is a digital transaction that gets resources from other devices or servers to perform a specific function that brings a related benefit. The hardware for the IoTs is becoming smaller and has more processing power meaning the devices have more computational capacity. Radio frequency identification (RFID) is widely linked to applications for sensing and tracking items and products in daily use, and may be called a growing technology. This growing technology is linked with exchanging data or information taken from wired or wireless sensors and takes actions on behalf of related running processes that create services to satisfy the user request without human intervention. This growing technology over the internet uses interfaces in the shape of services, and facilitates services by querying any change in information that is associated with this technology.

Due to fast and continuous development in the growth of technology with associated services, connected users may need a common platform of services, termed an Internet of Services (IoS), to access daily life services such as tracking a child in a mall, smart city services, and other things. The IoS also helps its users to achieve the required business value from the internet of things. The IoS is a way to think about the internet of things on a large scale through the provision of effective and efficient services in a timely manner on behalf of requesting users or processes. The services reside in different layers of the organization; that is, in different operational units or IT networks, or they can even run directly on devices and machines within the company. There is a need for an architecture that can provide a platform for the secure provisioning and use of services initiated by computing devices, mechanical and digital machines, objects, animals or people. The IoS responds to and serves the user query for a resource or information in an efficient and timely manner, with the help of a variety of service components: service proxy, a service repository, a service registry and broker, a service constructor and a multitude of service providers. The component ‘service proxy’ which is working in between user request and service served. The component ‘service constructor’ is constructing or composing services from multiple services not finding compatible services with its service contract using the service registry and broker. The component ‘service registry and broker’ is responsible for registering service and coordinating the connections between the service components and providing appropriate contract for services also reads the data from registry and make the right

connection with other service component. The component ‘service repository’ is responsible for storing and managing all sorts of information or data related to services, instead of direct access with all services components. The ‘service providers’ ensure that the availability of service matches with its service contract and advertises this to the service registry and broker. The component ‘service driver’ sends data on request to client through service proxy also storing data in service repository. To get efficient, effective services there is an urgent requirement to secure the interaction between these services and all the architecture that is deployed.

There is a lack of platforms available for putting users and the internet of services together. Many researchers are separately involved in research into the IoT, the IoS and security. However, there is a gap in the current research around proposing a platform for bringing users and the IoS together in a secure manner. We also need to ensure security in our architecture. The proposed platform is designed after consideration of combining the IoS with an assurance of services, served in a secure layered architecture, where connected users can interact, communicate and access a required service with ensured availability. An architecture is designed to incorporate all the issues addressed using secure data communication. Different programming languages are used to develop each platform while keeping the efficient delivery of services in view. In the given architecture, we try to consider security threats such as confidentiality, integrity and availability and attempt to mitigate the security vulnerabilities that cause the threats.

An architecture is a software design where services are available to the other components of application using a secure communication protocol. Secure communication protocol ensures confidentiality, integrity and the availability of data over the network and in the infrastructure. We propose a layered architecture that divides the overall services and the task requested into layers and defines the services provided by the individual layers. We designed a responsibility for the different layers, keeping in mind the services and requests handled in each layer and ensured the reliability, security and efficient response for each service requested. The architecture blocks direct communication between nonconnected layers, while the communication between the connected layers is limited to establishing service calls and service responses. In designing the architecture, we respected the responsibility of each layer. Each layer is designed in such a way that the upper layer makes use of the services in the lower layer and is not involved in the details of other services.

The proposed web service introduces a secure interface between the application layer and the service layer. This interface is used to ensure message level data security, while any service request comes from the application layer to the service layer using a web security token. A web security token is used to establish a secure communication between the application and services using encryption and a digital signature. The attacks and vulnerabilities observed during services communication and the remedial actions needed to mitigate these threats for improving the architecture. Later, the sensing-network layer and other layers data communication security discussed in detail. In the next paragraph, we discuss the problem statement.

1.3 Problem Statement

Due to the interconnection of services, some researchers have highlighted problems related to integrated services. In the internet of services many heterogeneous services are available, but they do not take efficient and effective service delivery in to consideration. There must be a consideration of these things along with smart matching from the available updated database, which should occur in a timely manner. After ensuring the effectiveness and efficient delivery of service, a related issue needs to be addressed, which is the IoS data communication security and end to end infrastructure security. There are many security vulnerabilities related to our architecture, but we address the protection of confidentiality, integrity and availability of data during data communication and in our infrastructure. Confidentiality is an assurance that information or services will be prevented from reaching unauthorised users. Integrity is ensured by maintaining the consistency and trustworthiness of the data communication or services communication. Availability is ensured by the ease of services accessibility, meaning that whenever we need a service using any server or infrastructure it will be easily available and accessible.

Many researchers are involved in research into the IoT, and also separately into the IoS and security. However, there is a gap in the research as there is no single platform available for combining users and the IoS together in a secure manner. The main issues are about how to achieve full functionality for the interconnected services, and how to make a platform for enabling their adaptability while guaranteeing trust, security, and the privacy of users and their financial assets. There must be an architecture that can provide services

to users through trusted infrastructures, different design platforms and multiple types of services with a secure provision of services.

Moreover, the internet of services has new problems and issues related to efficient end-to-end network communication in a secure manner. Few researchers are currently involved in finding a solution for secure data communication in the internet of services and in searching for solutions to make services available in a smart and efficient manner. However, there is still a lack in the research, as there is no single platform available for combining users and the internet of services together with an assurance of confidentiality, integrity and the availability of services. The major objective of this work is to provide reliable services for the internet of things, with a focus on the design and development of an architecture that provides secure data communication when a service is requested and keeps an eye on secure hardware and end to end security. Also, the adaptability of a secure service-oriented architecture needs to be tested using different programming platforms.

1.4 Assumptions:

In this research, several assumptions have been made.

1. The service provider makes their own arrangements for the collection of data, such as deploying a network or aggregation of data from IoT devices.
2. It has been assumed that service providers will ensure the availability of the IoT data based on the assumption that we will ensure effective services to users.
3. There is an available trusted domain where we deploy our server, which will aide in the secure communication of data from one layer to another and between the components. This can be done by using technologies such as Software Defined Perimeter (SDP).

1.5 Limitations:

Some limitations of this work are as follows.

1. If the service provider does not give us updated data, there is no alternative available.
2. During evaluation, we did not consider the entire system when integrated with the platform, rather we tested the components.
3. We are dependent on available service providers.

.

1.6 Terminology used:

IoT: internet of things

IoS: internet of service

User: A person or process requesting a resource.

Authentication: a way by which a user proves his/her identity to a requestor, normally through the use of a credential. Authentication where both requestor and requestee verify their identity.

Object: A resource protected by our security policy.

Authorization: The platform whereby a user can access an object and by which we determine whether a subject can access or use an object.

Trusted domain: An administrative structure that has a consistent local security policy.

Service: A digital transaction resourced by other devices such as radio frequency identifiers to perform specific functions and get the related benefits.

Functionality: User request for requesting some resources to fulfil their needs in effective manner

1.7 Structure of Thesis

Section 01 consists of the introduction to the thesis including the problem statement, a summary of research, assumptions, a limitation review and the structure of thesis. In section 02 there is a detailed discussion about related work, the research objectives and a critical analysis. Section 03 discusses the architecture for services in the IoT including the service layer components, which consider the temperature of wireless sensors at different times and in different locations that can be fetched using different programming languages. Section 04 discusses the detailed architecture for services related to IOT devices. The layers of the model are the applications, the service layer and lastly the sensors-network layer. In the service layer, we explain the service proxy, service repository, service provider, services-registry-broker, service facilitator or service proxy and the service driver. Section 05, discusses the conclusion and future directions of research.

Section-02 Related Work

A number of research works related to service oriented architecture for the internet of things, services, network security in the internet of services, the software defined perimeter and the communication security of architecture were analysed. Researchers have different points of view about services oriented architecture and a gap was found in the research around providing a platform for the user and the internet of services to interact together using secure data or service provision. Researchers proposed various types of architecture to secure network communication security, but we tried to explore the research objectives and make a critical analysis of the research papers. After analysis, we identified a gap in the research in services oriented architecture and highlighted the areas needed for future secure services architecture.

In a study, [13] explains the services as a middleware system that allows the specification of tasks at a high level. They [13] recommended hardware and distribution details from a programmer's point of view and suggested ways to improve efficiency and take energy consumption into consideration. The authors [13] considered programmability and recommended it be flexible to allow for the configuration or reconfiguration of features and functioning. Authors [13] recommended adoptability, and suggested adapting network changes to continue the correct working of the network. Scalability was also considered to allow for growth in the number of network nodes. For robustness, a topology has been proposed to support the ever-changing network [13]. Authors [13] have taken into consideration several design requirements for end users - typically biologists, meteorologists and others -. who need to use wireless sensors networks in their applications. Re-configurability explains that applications may be able to change elements such as sampling rates and the types of data collected. Health monitoring allow us to know the overall network health. Authors [13] also explained the need to understand the available energy level in network nodes. In this article, the authors [13] did not pay attention to the data communication security of the services provided or requested by users.

In another study, [14] the author explains that the internet of things is emerging from the internet and other networks with wireless technologies to make physical objects interact online. They [14] also explain that the internet of things has developed to become a

promising technology that has received significant research attention in recent years because of the development of wireless communications and micro-electronics. The authors [14] highlight that data communication security risks and privacy are also taken into consideration. Authors [14] consider that research and the applications of the internet of things are in the early stages. Authors [14] emphasize the provision of communication for network security in the internet of things. In this thesis, first the internet of things is compared to the internet. Authors [14] explain that the internet of things is based on the internet, the internet of things end to end security protocol and proactive measures are not enough to provide security through the perceptual layer, the transport layer and the application layer. Further, this thesis provides the object access control and privacy protection through the object application layer, addressing the DNS and IP addressing phases. Finally, combining the internet of things object addressing security model with a practical application scenario, this thesis designs the internet of things object security access model. In this model, the access requester can access objects in different domains through a single sign-on. This model provides protection for end-to-end communication with the access requester. However researchers [14] have used the TLS and IPsec systems for end to end security. TLS and IPsec still have vulnerabilities and threats when compared to a software defined perimeter and SAML for proving end to end session level security.

A number of points need to be considered in a critical analysis. The security issues of the internet need to be further segregated and analysed separately. Secondly, the author [14] has not considered the multifactor authentication and session key agreement protocol for end-to- end communications in the object security access model. In future, the key agreement protocol needs to be designed to be fully integrated into the mutual communication end-to-end authentication. Thirdly, the internet of things object security access model still requires security communication between the terminal equipment and the access gateway. The end-to-end authentication key negotiation and cross-domain access in the internet of things still needs in-depth study.

Service-oriented computing, in general, aims to make services available and easily accessible through standardized models and protocols without having to worry about the underlying infrastructures, development models, or implementation details. Service-oriented middleware could play an important role in facilitating the design, development,

and implementation of service-oriented systems. In another study, authors [15] did not consider end-to-end and data communication security. They did not pay attention to services discovery details.

In another study the authors [16] presented vehicular networks, which they expect will play an important role in the future wireless communication service market. Service-oriented vehicular networks rely on both vehicle to infrastructure (V2I) and vehicle to vehicle (V2V) communications. Authors [16] consider a combination of infrastructure-based broadband wireless networks and ad hoc networks. In this article, the authors [16] discuss the key security requirements, and point out that existing solutions may face the challenge of vehicle to infrastructure authentication delay and public key revocation issues. The proposed vehicle to infrastructure fast authentication scheme is based on vehicle mobility prediction and a road side unit (RSU). Authors [16] considered a short-time certificate scheme, which successfully addresses the authentication latency and public key certificate revocation issue. Further research focuses on how to seamlessly integrate security at the vehicle to infrastructure and vehicle to vehicle levels to obtain a general security platform for the service-oriented vehicular ad-hoc network (VANET). In this research the researcher [16] considered the network security of an ad-hoc network but was unable to pay attention to the services of the internet of things and data communication security between interconnected devices.

In this survey the authors [17] explains a complex cyber-physical system. The internet of things integrates various devices equipped with sensing, identification, processing, communication, and networking capabilities. Sensors and actuators are getting increasingly powerful, less expensive and smaller, which makes them useful in the internet of things. Industries have a strong interest in deploying devices in the internet of things to develop industrial applications such as automated monitoring, control, management, and maintenance. Due to the rapid growth in technology and industrial infrastructure, the internet of things needs to be used in industry. For example, the food industry is integrating wireless sensor networks and RFID to build automated systems for tracking, monitoring, and tracing food quality along the food supply chain to improve food quality. This paper reviews recent research on the internet of things from the industrial perspective. The author [17] firstly introduces the background and the service-oriented architecture models of the internet of things and then discusses the fundamental technologies needed in the internet of things.

Secondly, the author introduces some key industrial applications for the internet of things. Next there is an analysis of the research challenges and future trends associated with the internet of things. A main contribution of this review paper is that it considers only the industrial internet of things applications and explores the challenges and possible research opportunities for future industrial researchers. This research paper focuses on the industrial internet of things, but ignores the user requested services in architecture and data communication security.

Authors [18] discusses the internet of things and the rapid growth in modern wireless telecommunications. The basic idea of this research is that a variety of things or objects such as Radio-Frequency Identification (RFID) tags, sensors, actuators and mobile phones can, through unique addressing schemes, interact with each other and cooperate with their neighbours to reach common goals. The main strength of the internet of things is the influence it has on several aspects of everyday-life and the behaviour of potential users. The internet of things is commonly used by corporate and domestic users. In this context, living things, e-health and enhanced learning require possible application scenarios in which the new paradigm will soon play a role. Similarly, from the perspective of business users, the important fields are automation and industrial manufacturing, logistics, business/process management and the intelligent transportation of people and goods. The authors [18] foresee that internet nodes may reside in everyday things – food packages, furniture, paper documents, and more. They highlight the future opportunities that will arise, starting from the idea of popular demand combined with technology advances. The possible threats that may derive from the widespread adoption of such a technology are also stressed. The authors emphasize that everyday objects can become information security risks, and the internet of things could distribute those risks far more widely.

The authors [18] provide visions of the model of the Internet of Things that come from different scientific communities. They also review the enabling technologies and explain the benefits for everyday life. There is a review of the major research issues the researchers [18] still have to face. The authors [18] introduces and compares the different versions of the model of the internet of things available in the literature. The future benefits from the full deployment of the internet of things is also described.

The authors [18] consider the enabling technologies that explain how the concept of the internet of things is feasible, but lot of research effort is still required. Standardization

activities need to be carried out on related technologies for the internet of things. More specifically, the authors focus on addressing and networking issues, rather than the problems related to security and privacy.

In another study the authors [19] explain that many different technical alternatives are available for healthcare applications, which means that the concrete projects can review the whole set of possible solutions to determine the optimal ones, considering the constraints and priorities of the corresponding applications. The authors [19] explain that we should have a structured system engineering methodology to guide the corresponding decision processes for developing the internet of things health ecosystems. The authors [19] did not consider network security, routing issues, or services and discovery issues.

Authors [20] presents a mechanism for evaluating service-oriented architectures for pervasive computing systems. The authors [20] explore the type of tasks in the available services. They explain that the service environment that supports user tasks must have the service potential to maximize resource utilization. There is also a discussion of highly dynamic environments where the state of resources changes frequently. The authors [20] introduce redundant services and limiting the impact of a single service. Due to the redundancy features, the re-configurability of a service can be guaranteed. Service composition explains an effective mechanism to support user tasks. The authors explain that composition can support the reuse of services, the quality of service assurances, and mobility and fault tolerance within service oriented architectures. In this paper, the authors [20] has considered that the composition mechanism's seamless service composition could perform better than the traditional discover match-based approach. In future, the authors [20] will extend the ongoing work to develop a framework for service creation, service composition, service deployment and the maintenance of services in a dynamic and heterogeneous environment. Services network security is not considered and is ignored in this article.

Authors [21] presents the responsibilities and needs of service oriented architecture (SOA) stakeholders, such as service providers and service consumers. Various service-oriented architecture verification and testing techniques are presented, including monitoring, modelling, and reliability analysis. The authors [21] performed integration testing, functional testing, non-functional testing, and regression testing of service-oriented architectures. They discussed the commercial tools available for service testing. The authors also describe IBM's Web Services Navigator as a testing and debugging tool that traces and

visualizes the execution of services and helps programmers find bugs. Authors [21] also explain WS interoperability across platforms and released a service testing tool in March 2004. The tool has two components: a web service interoperability organization (WS-I) monitor and a WS-I analyser. The WS-I monitor is placed between the client and the web service to log all messages, requests, and responses as they travel back and forth. The WS-I analyser then analyses these logs against the interoperability requirements.

Authors [21]describes improved interoperability and the most prominent benefits of service oriented architecture. With web services technology, service users can transparently call services implemented in disparate platforms using different languages. In this research paper, the goal of syntactic interoperability is supported by two basic standards: web service descriptive language (WSDL) and simple object access protocol (SOAP). Modifiability is also explained as the ability to make changes to a system quickly and cost-effectively. Services are explained as modular and self-contained, reducing the number of usage dependencies between service users and providers, meaning the cost of modifying these services is reduced. Performance in a service-oriented architecture is measured by average case response times or throughput. The authors [21]claim that the performance of service-oriented architecture is negatively impacted because service-oriented architecture enables distributed computing, and the need to communicate over a network raises the response time. But the author ignores the message level and end-to-end communication security.

In another study the authors [22] explain that the internet of things (IoT) represents the current and future state of the internet. The large number of things or objects connected to the internet produce a huge amount of data needing much effort and processing operations to transfer it into useful information. Moreover, the organization and control of this large volume of data requires novel ideas in the design and management of the internet of things network to accelerate and enhance its performance. As per the authors, [22] the software defined systems provide us with a new model that appears to hide all complexity in the traditional architecture system by summarising all the controls and management operations in the underlying devices (things in the internet of things) and setting them inside a middleware layer, a software layer. In this paper, a comprehensive software defined base framework model is proposed to simplify the management process of the internet of things and to provide a vital solution for the traditional architecture of the internet of things to forward, store, and secure the data produced by the objects by integrating the software

defined network, software defined storage, and software defined security into one software defined base control model.

Another study [23] defines advanced computing in cloud computing infrastructures and explains that that can only become a viable alternative for the enterprise if these infrastructures can provide proper levels of non-functional properties. A company that focuses on service-oriented architecture needs to know what configuration would provide the proper levels for individual services if they are deployed in the cloud. The authors [23] present an approach for the performance evaluation of cloud computing configurations. While cloud computing provides certain service levels, these are typically done for the platform and not for a specific service instance. The authors approach focuses on individual services and therefore provides more relevant and granular information. An experimental evaluation conducted in Amazon Elastic Compute Cloud [23] was used to verify their approach. The authors [23] have not taken into account the services and service-oriented architecture security threat and vulnerabilities.

Section 03 Architecture Overview

3.1 Overview of the Architecture for the Internet of Services

Due to the growth of IoT technologies and communication protocols, there is a need for the development of an architecture that ensures reliability and efficient responses from requested resources. We propose an architecture to facilitate the requests made by the users. In this section, we discuss in detail the services requested by interconnected users that can help them in their businesses and in daily life.

The service providers ensure a continuous updated data delivery to our server. The proposed architecture has been designed in a way that provides a separate layer to do each of the specified tasks that help the user. This layered model reduces the complexity of interactions with the IoT devices and users together and tries to simplify it. The lowest layer is connected physically with the real world, handling data transport issues, and the upper layer is connected to the end user.

We configured and designed the [24] responsibility of the different layers, keeping in view the functionality handled in each layer. We also ensured the reliability and efficient response time for a requested function. The following important factors motivated us to split our architecture into three different layers:

The proposed layered architecture incorporates the following features:

- Reduced design complexity.
- Increased modularity.
- Highest layer is closest to the end user.
- Low level layers are physically connected to handle data transportation issues.
- Maintainability.
- Scalability.
- Ease of testing.

The architecture does not allow communication between the layers that are not connected to each other. In the architecture, we provide a platform to create an effective service for the requested user. The lower level layer is connected physically to the real world when handling data transport

issues, while the upper layer is connected to the end user who requests functionality. We discuss the layered architecture in more detail below:

3.2 Layered Architecture

We built an architecture to facilitate the user requests for resources in an effective manner, which is associated with new technological advancements. The. proposed architecture has been designed in a way that provides each layer with the ability to achieve specified tasks that facilitate the functionality (user request for requesting some resources to fulfil their needs in effective manner) to end user or process on behalf of user. The three layers are the application layer, the service layer and the sensing-network layer. A user can query any service using simple object access protocol (SOAP). The SOAP query may, for example, request the average temperature in Auckland on 10/03/2017. This request is then sent from the application layer to the service layer where it becomes an input in the service layer. The service layer takes the appropriate actions for an effective delivery of the service to the requested user, but has an interaction with the sensing-network layer to fetch updated data. In our architecture, we rely on the service provider (responsible for ensuring IoT sensor data to our server) to provide continually updated data. All the interactions of the service provider to obtain updated data from the sensor or sensors network are handled and managed in the sensing-network layer.

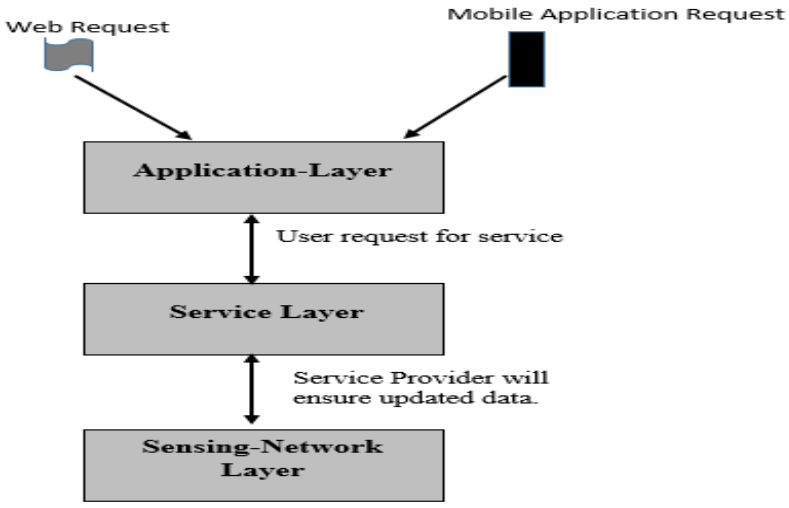

Figure 3.1: Overview of Layered Architecture

With reference to Figure 3.1, the user requesting a web or mobile application request forwards it to the service layer to get an effective and efficient service. Sensors connected to devices interacting with each other can be handled in the sensing-network layer. Communication between connected layers is limited to procedure functionality calls and functionality responses. Each layer is designed in such a way that the upper layer makes use of the functionality of the lower layer without any concern for the details of functionality in the other.

3.2.1 Application Layer:

This layer provides interaction between [25] users and applications like the web and mobile applications that request resources by querying simple object access protocol (SOAP). For example, a web query about temperatures in a specific location is forwarded from the application layer to the service layer.

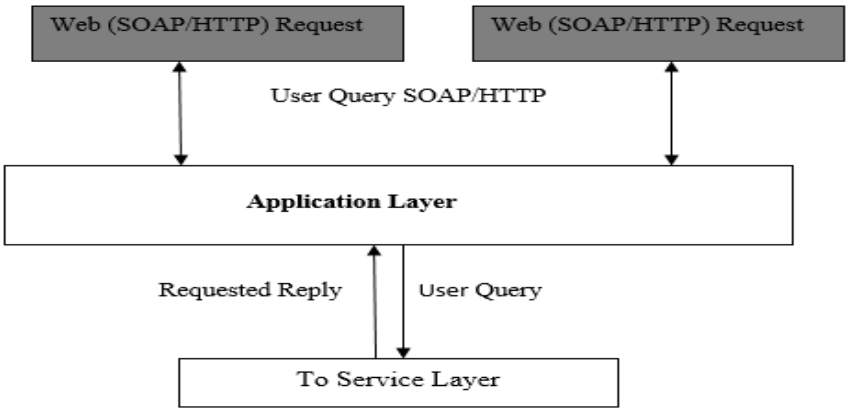

Figure-3.2: Application Layer Interaction with Service Layer

The application layer facilitates the user request to the lower layers. This layer has a direct connectivity with the user and all user applications. The application layer can forward a simple object access protocol message or mobile application request to the lower layers such as the service layer and the sensing-network layer. A user request takes the form of a query, as shown below.

```
SELECT AVG(Temperature)
FROM Temperature.Sensor R WHERE location IN region
HAVING AVG(Temprature) > 100
DURATION (now, now+3600) EVERY 10
```

The SELECT clause specifies the attributes and aggregates from the sensor records. The FROM clause specifies the distributed relation of the sensor type. The WHERE clause filters the sensor records. The GROUP BY clause classifies sensor records into different groups with same attributes. The HAVING clause eliminates groups by the records found.

The importance of this layer is that it has the capability to provide high-quality smart functionality (a user request for resources that fulfils their needs in an effective manner). The user's request, in the form of SOAP query, is forwarded to the service layer for an effective, efficient response from the services layer. In our example, the query of a user about the temperature in Auckland is forwarded from the application layer to the service layer for further action. The service layer resolves this query in a timely and efficient way and returns it back to the application layer. The application layer provides a software interface for the user and passes the user's query on to the services layer. After getting an effective response from the service layer, the application layer receives this service as an input to the application layer and provides the secure interface for the

user requesting the service. The application layer covers numerous vertical market applications such as smart homes, smart buildings, transportation, industrial automation and smart healthcare. In our architecture, we developed applications using different programming languages to provide a platform for users to access multiple services in an effective and timely manner.

3.2.2 Service layer:

This layer manages the services that are generated to fulfil user requests from the application layer. The major role of this layer is to provide effective and appropriate services in such a way that the request will be resolved in an effective, smart and timely manner. The service requirement explains the agreement between the user and the initiator of the service. A properly designed service will be able to identify common application requirements and provide an application programming interface to support the requested services.

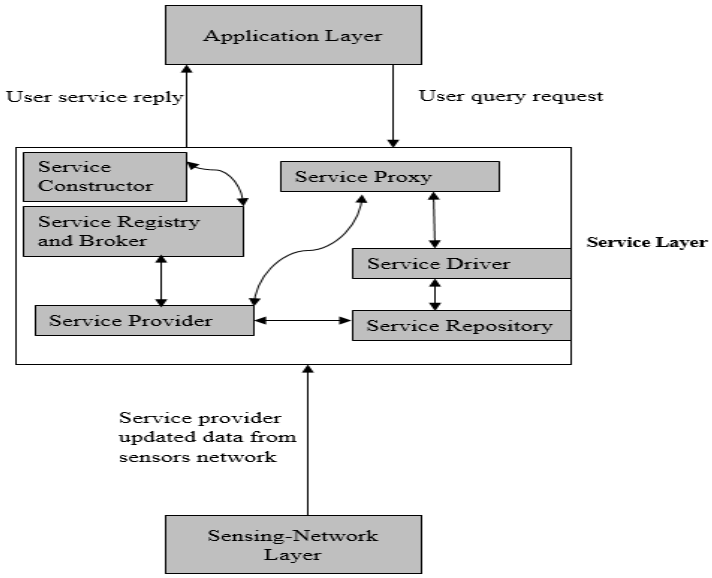

Figure 3.3: Service Layer Overview

For example, as in Figure-3.3, due to the increase in interconnected IoT devices, service is the key player that works in the middle of the architecture. The main activity in this layer involves defining the services requirements of the users, satisfying the requests that come from the application layer, and routing to the service proxy to find an appropriate match for the service. The service proxy

finds the service broker and registry to serve the request appropriately and arranges a contract for the available services. If the service is not in the registry it will arrange for further discovery. The service layer receives a user query from the application layer. This query is generated using the SOAP web service or any application that has been transferred to the service discovery module to find an appropriate service match. The service proxy component finds the appropriate service broker to serve the request. The service registry and broker will further make an appropriate contract for the service if an appropriate match of service is found in the repository. The service provider ensures updated sensor data is made available to our server. The service provider makes all the arrangements and interactions with the sensor network to gather the updated data. The term services considers general services, such as the services generated by radio frequency identification devices; however we ignored the warehouse and customer relations model services. To achieve efficient, effective services there is an urgent requirement to secure the interaction between services and all deployed architecture. For example, a user needs data about a temperature sensor in Mount Roskill, Auckland. The Mount Roskill sensors may be out of order, but this service request can still be served by matching data from nearby location sensors to form an updated database. This layer is also known as middleware, as it works in the middle of the sensing-network and the application layers.

3.2.3 Sensing-Network layer: This layer is integrated with the existing hardware (RFID. sensor, actuators and wired or wireless sensors) and provides the interface that senses and controls the physical world. In our architecture, the service provider, which is responsible for ensuring that the IoT sensor data is sent to our server, is connected through this layer where it continuously fetches updated data from the sensors and ensures delivery of the data through their own networks till it reaches our servers. The sensing-network layer interacts with the sensors and actuators, forwarding fresh temperature data from the location to the application layer via the service layer. This layer also takes care of all networking, routing, switching and tunnelling tasks.

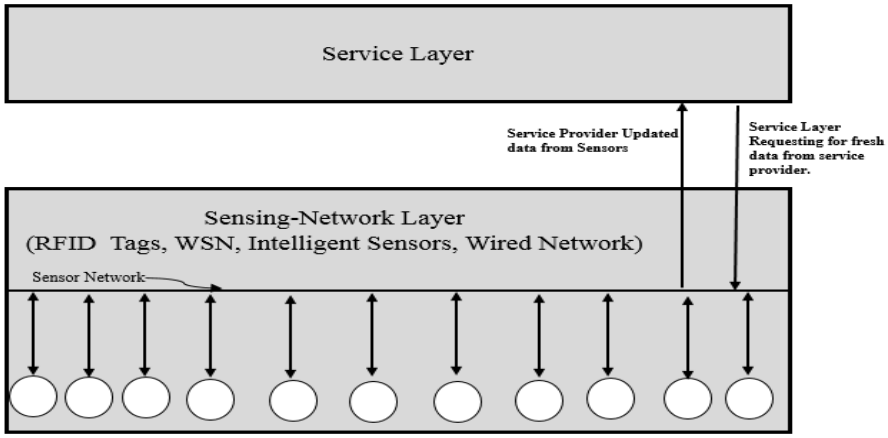

Figure-3.4: Sensing-Network Layer Interaction with Service Layer

The sensing-network layer is a physical interconnection to IoT devices and is responsible for the interactions among the devices that are connected in a network with the internet of things. This layer provides physical and real world connectivity with the IoT. In our architecture, the service provider, which is responsible for ensuring IoT sensor data is sent to our server, has access to the sensing-network layer, making arrangements to fetch fresh data from the sensor network. After fetching the updated data, the service provider (responsible for ensuring IoT sensor data to our server) ensures the correct arrangements so our server continuously updated. Additionally, this layer service provider has connectivity with the sensors and fetches the updated data, while also taking care of and managing all networking functionality. Network functionality means that all routing, switching and tunnelling tasks are performed in this layer. The service provider has more devices related to networks that have RFID or intelligent sensors, wired nodes and wireless things that are connected to the sensing-network layer that provides the requested functionality. In the sensing-network layer, smart sensors with tags or sensors automatically sense and transfer information among various interconnected devices and communicate with the upper layers. Information collected from the sensors is sent through wired or wireless receivers for further processing. The service provider needs resources that have interactions with physical world connectivity in this layer.

After discussing the overview of the architecture, there is a detailed discussion of effective services that have secure data communication in Section 4.

Section-04 Architecture Details

In this section, we present the detailed structure of the proposed architecture.

4.1 Detailed view of the Layers

With a changing IoT market, the availability of new products becomes a challenge for developers, who cannot support to them all. For each specific product, developers must implement or integrate a gateway for collecting device data and controlling the settings. Many devices in the IoT add a large amount of data, but to generate services with real value to a user requires hiding the IoT complexity and replacing it with an easy to use services platform that focuses on and delivers the required business value from the IoT by including valuable information. Generally, a service may be considered as a business activity, where a requestor is granted access to a resource for the requesting party to perform a function and get a related benefit. The resource can be a workforce, industrial systems, information systems, consumables, traffic and others. The Internet of Services means many connected services, which are used for buying, selling and offering on a global network that has both users and brokers. Services must be defined in a way such that the commercial aspect and the technology aspect come together. A service here is an interface that is not dependent on a platform, a programming language or the operating system that was used to implement the service.

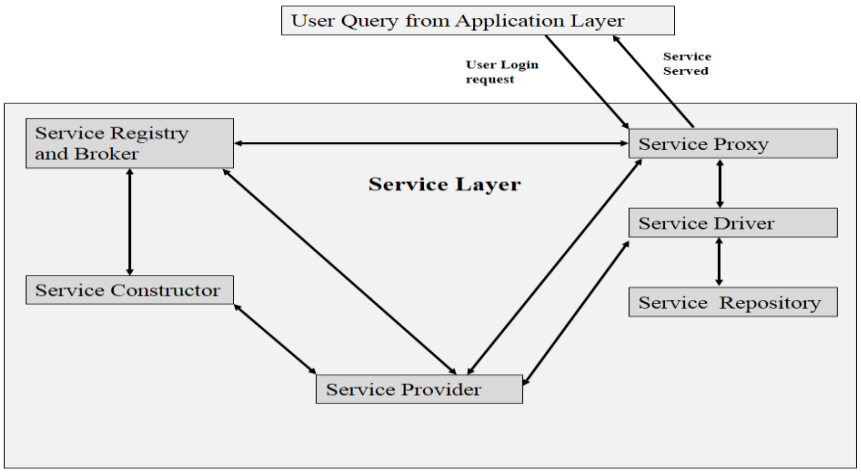

Figure-4.1: Services Structure Detailed Overview

During the services interactions, communication security is a major concern and needs our attention. Security is a broad area, but we have taken confidentiality, integrity and availability into consideration. Confidentiality means the assurance of preventing information or services from reaching unauthorised users. Integrity is ensured by maintaining the consistency and trustworthiness

of data communication or services communication. We also ensure that services and data are not modified or changed in transit by unauthorised users. Availability means any service using any server or infrastructure can be easily available and accessible.

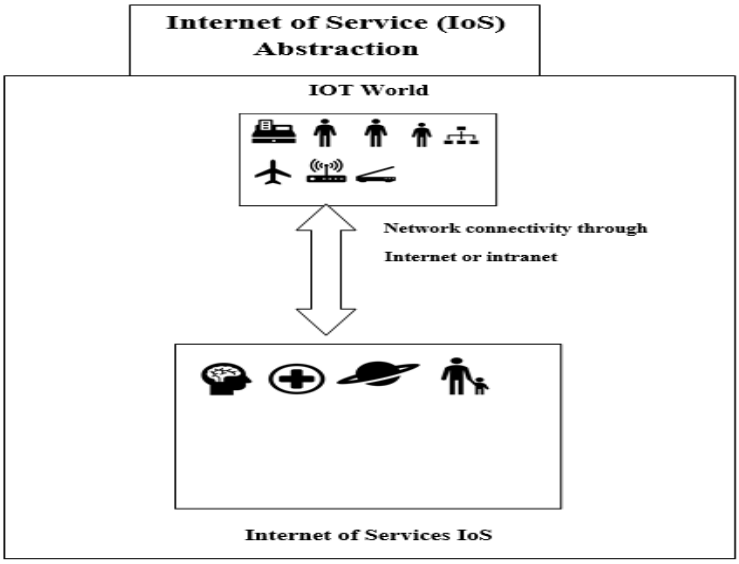

Figure-4.2: Internet of Services abstraction

In this section, we explain the functionality of the application layer and the service layer in detail. We do not discuss the details of the sensing-network layer because it is the responsibility of the service provider to provide secure use and to update data from the network's sensors.

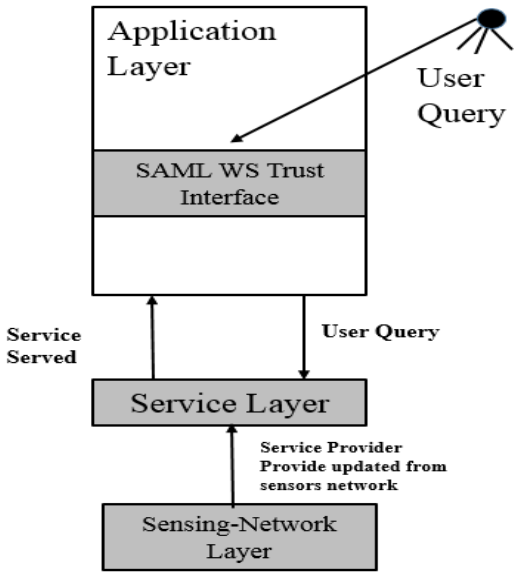

Figure 4.3: Application Layer Detail

With reference to Figure 4.3, the application layer forwards the user query to the SAML WS trust interface for secure user authentications, and later the user query is forwarded to the service layer for effective services. The services layer arranges effective services for the user and returns with a secure service reply for the user query. The sensing-network layer functionality in this section is not examined, as the service provider will ensure updated data to our servers.

4.1.1 Application Layer: The application layer forwards the user SOAP query to the service layer using a web site or customized application, which is designed for mobiles through a security interface known as the web security trust interface. After service, this request becomes a service reply and is sent back to the user for the appropriate services. The SOAP query or mobile application request is further sent to the low-level layers for further functionality. After getting an authorized reply service, the user can access any service through a single sign on. The importance of this layer is that it can provide high-quality smart services to meet user's needs. There are three key roles in the proposed secure solution:

1. ***Service Provider (SP):*** The service provider in SAML provides secure data communication by ensuring encryption and a digital signature for the identity provider. Verification gives secure access to the user so they can get secure services or resources.
2. ***Identity Provider (IDP):*** The identity provider gives proof of identification for authorised users requesting a service.
3. ***Client or User:*** A requestor requesting a service.

Every user query must pass to the service layer through a secure software interface known as the web service security trust (WS Trust). The application layer covers secure communication between the IDP and SP through a user requesting a service. Afterwards, verification of the web service security token, SP arranges a secure communication between the client and the service layer, using a single sign on. A secure solution is proposed using security mark-up language (SAML) and single sign on.

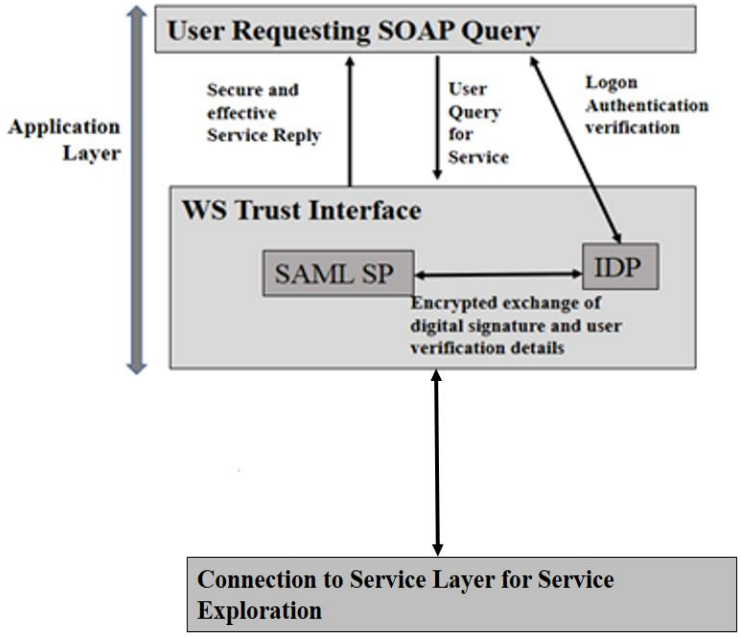

Figure-4.4: SAML Token Exchange

Before building an architecture using a secure platform, it is necessary to understand how the web security token will function, as some concepts need more clarity about security assertion mark-up language (SAML). In Figure 4.4 the user on a website or using a mobile application can sent a web request or query to a web service, which can be seen in what follows.

The data and communication security is our major concern, and therefore we have used SAML deployment in our architecture. After authentication from an identity provider, the SP serves to secure access with a single sign on.

Like SAML there are three players such as SP, Identity provider (IDP) and the client. After the deployment of these players using the SAML token service to exchange data, we ensure the confidentiality, integrity and authenticity of data communications.

People using a web browser or any mobile application can request any available service in our architecture. The requesting user must reach the service proxy in the service layer, after passing from the web service trust interface to further match and explore the services. We are presenting a web service security trust (WS-Trust) interface between the service proxy and the user request using a security assertion mark-up language (SAML) infrastructure. The SAML infrastructure is a major role player for ensuring data security.

4.1.2 WS Trust Interface token exchange: There is a responsibility of the [26] IDP to provide a secure token for the client to get further access and verification of their token in SP. The service provider of SAML (SP) can further verify and try to validate the SAML token. If the client has a security token issued by a trusted security token service, the SP accepts that token sent by the client. The security token service can issue security tokens based on the requirements provided by the client or the SP.

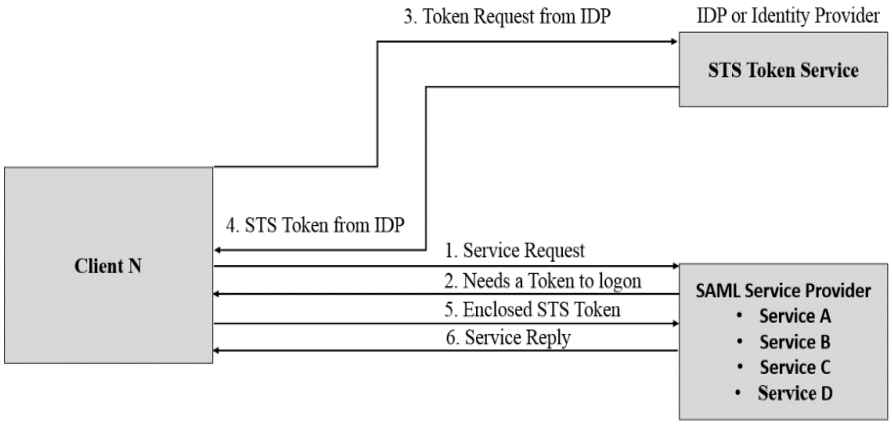

Figure-4.5: SAML Token Exchange

Note: Client N is any client sending a SOAP query to get any services in our architecture after passing through the web security interface of our architecture.

Explanation of example Figure-4.5

1: Client N can be any client, such as a web client user sending a SOAP query who needs to get serviced through service A and issued a service query.

2: SP needs to verify a token to forward the user query to the service layer. For this purpose, the request sends back to the client N with the requirement of ensuring the SAML Token, which can be further verified by SP.

3: The token request is sent to a security token service (STS) in IDP for security. The security token service is the role of the IDP and further verifies the identity of client N by contacting the user for proof of identity. If the user verifies his identity by using a username and password, then the IDP further will verify the authentication of client N.

4: STS will then issue a token consisting of X.502 and a private key, after verification of the signature. X.509 certificates are installed on SP. Whenever a client requests a service, the X.509 certificate, together with a private key is inserted into the token. Tokens also have digital signatures of clients to ensure authenticity.

5: Client N sends an enclosed STS token to SP, which was provided by IDP after verification from client logon credentials. SP verifies the signature and authenticity of token through a digital signature and the X.509 certificate and further gives access to the services layer for the right service.

6: Client N get multiple services like Service A, Service B and Service C available in the services layer of our architecture with a single sign on.

After setting up the SAML infrastructure, there are three role players like SP, IDP and the requester plays their role in infrastructure and afterwards, we feel the requirement to analyse the vulnerabilities of SAML web service security interface:

In the web security trust (WS Trust), the interface token exchange takes place between the IDP and SP with the user plays role in middle. We will further explain the details of the secure token exchange between the IDP and SP:

4.1.3 WS Security Trust Interface: This is a single sign on using a standard browser that deals with the issuing, renewing, and validation of SAML security tokens, as well as the ways this is established. It assesses the presence and work as a broker that makes trust relationships between IDP and SP through the requesting user in a secure message exchange. Whenever the user requests a service, they need a SAML token provided by IDP to accomplish the request. To get the SAML token, the user contacts the security token service of the IDP. The identity provider verifies the multifactor authentication of the user. Then the encrypted response is forwarded to the service provider. If the response is verified by the service provider, the user is able to get a secure encrypted service. The X.509 certificate is used for token verification.[1]

4.1.4 Analysing vulnerabilities in the SAML Web Security Trust Interface

- ***Denial of Service:*** The SAML protocol is vulnerable to a denial of service (DOS) attack. It requires the client's authentication before the SAML protocol works. The

[1] . R Housley, *Network Working Group, The Internet Society (1999)*

authentication is used to track unauthorised users but it does not prevent attacks. In the infrastructure, the DOS attacks from non-insiders are blocked. We can handle a DOS attack by using a client multifactor authentication scheme.

- ***Eavesdropping:*** As there is not a data in-transit confidentiality requirement, a hacker can access both simple object access protocol (SOAP) messages containing requests and corresponding replies. Eavesdropper can get the details of request and elaborate the response. The details of a request can cause security vulnerability for the requesting party and expose the types of assertions that are required. Some cases may show the request itself, which composes a violation of privacy.
- ***Replay:*** There are fewer chances for replay attacks at the SOAP level, because we use WS Trust interface between the service layer and authorised user to further make use of effective service from available services. The vulnerability at the SOAP level is that potential users can expose themselves to a replay as a denial of service attack method. The best way to protect replay attacks is to protect the message capture in the first place. We have some transport level schemes that protect confidentiality of transit data, which is helpful in achieving protection.
- ***Message Insertion:*** This kind of attack is designed for SOAP binding and leads to binding a message with a request. SAML has immunity against these types of attacks.
- ***Message Deletion:*** We have signature verification and have immunity against this attack.
- ***Message Modification:*** Message modification is a vulnerability for SOAP messages as the message can be modified in both directions, when the token is going to be sent. This can change the meaning of the sent message. The system can be compromised when the assertion returns, and as a result, a denial of service attack can happen. The use of a SOAP signature will resolve this at the SOAP binding level. If messages are digitally signed (with a proper key management infrastructure) then the recipient has can be sure that the message has not been altered in transit, unless the key has been compromised.
- ***Man-in-the-Middle***

There is a vulnerability to man-in-the-middle (MITM) attacks. To prevent these, we used bilateral and multifactor authentication. In my research, we used salesforce site authentication, a mobile security code and the SAML token authentication. In this way, we have ensured that the conversation request comes from a trusted party.

After discussing the details, there must be an analysis of SAML, with an example.

4.1.5 Analysing SAML application with an example

Whenever a user query is forwarded to the WS trust interface, the interface requires the user to show authentication, as below in Figure 4.6. This request will be responded to after the logon query is received from the IDP. The IDP further verifies the credentials as shown in the figure below. If it is verified, the next step begins, or otherwise the user will receive an invalid response.

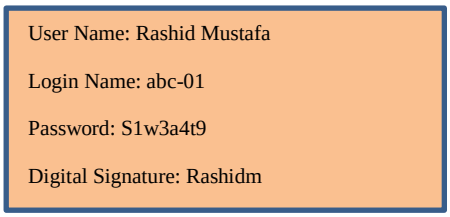

Figure-4.6: Login Structure

A trusted user is already created in IDP before sending user logon credentials. If it has not done prior to the user request, the user cannot be verified.

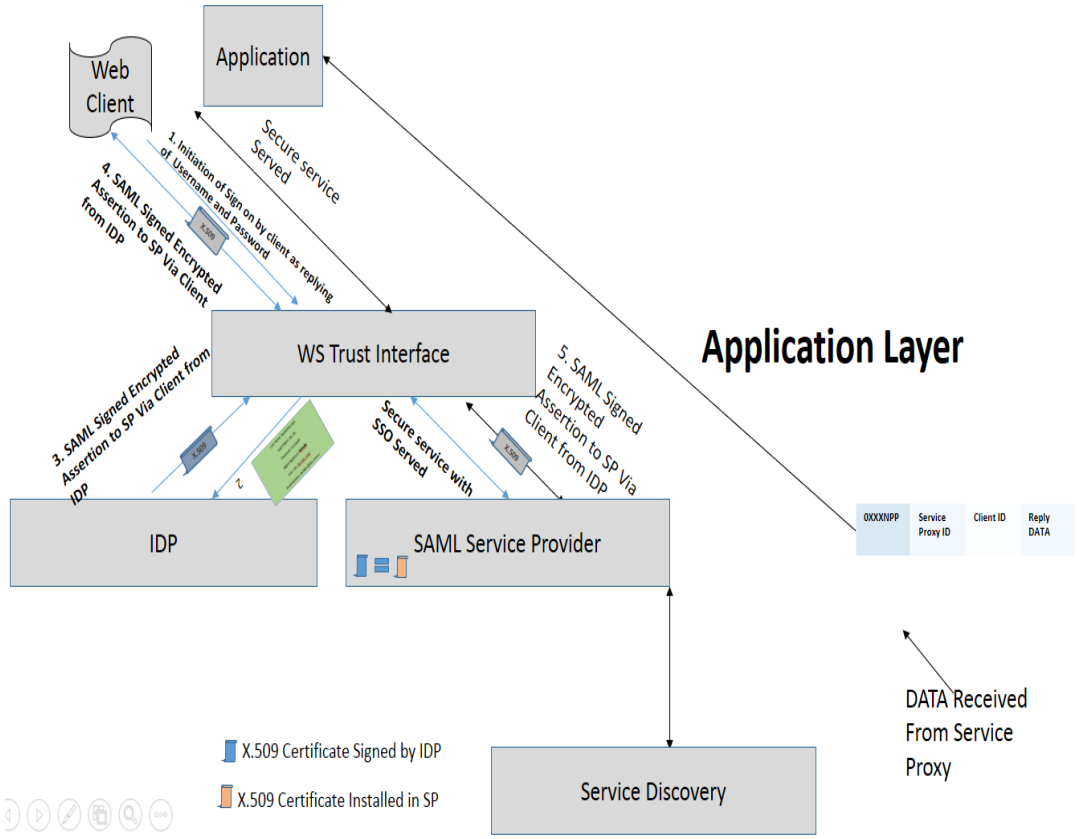

Figure-4.7: Detailed View of SAML Token Exchange

As per Figure-4.7, after creation of user on identity provider, [27] user request will be as follows:

The verification algorithm will be as follows

```
Step1. User Query reach to SAML SP /* User query addressed to and receive
by SAML SP*/

Step2. SAML SP needs verification token proof and Forward request to IDP
/* SAML SP require secure token from IDP through client channel*/

Step3. IDP needs user proof of Identity/* IDP will verify client
credentials by requesting username-password*/

Step4. User provide or forward proof of identity to IDP/* User will receive
the token from IDP */

Step4. After verification of IDP credentials, secure Token issued to
User/User will receive secure token if credentials verified*/

Step5. User forward secure token with x.509 cert. to SAML SP/* User forward
secure token with X.509 certificate with encrypted digital signature to
SAML SP.*/

Step5. SAML SP verifies token a) If Verified then forward request to
Service                                    Layer
```

```
b) If not then return to step 1
```

When the user credential verification between user and IDP has been completed:

If a successful logon is made by a user then IDP forward (X.509 certificate encrypted with digital signature and user identification) details to the SP or SAML service provider through the requesting user. Afterwards, the SP verifies the details in the already Installed X.509 through a digital signature. If the details are verified, then the user query for the service can be sent to the service layer for further service exploration.

Algorithm for the user query initiation is as follows:

```
Step1. SELECT Avarage(Temperature);

Step2. From Table=Temperature.sensor Where Location='reqion'

Step3. Having Average (Temperature)> 100

Step4. Update.Duration = 20 millisecond

The SELECT clause explains attributes and added from sensor
records.

The FROM clause explains the distributed relation of sensor
type.

The WHERE clause filters sensor records found.

The GROUP BY clause clarify sensor records into different
groups per same attributes. The HAVING clause removes groups
by a record found.
```

Note: Each query must have a unique query ID as proof of the query. We assume the service provider will ensure updated fresh data from the sensor-network layer to our server. After detailed discussion about the application layer, our SOAP query is transferred to the services layer. Assurance of effective delivery of services is the responsibility of the services layer.

4.2.1 Services Layer

Services respond to and serve the user query for a resource or information in an efficient and timely manner with the help of a variety of service components: service proxy, repository, service registry and broker, constructor and service provider. The component of service proxy works between the user request and the service provided and also keeps a record of all incoming and outgoing services. The component service repository stores and

manages all sorts of information and data related to services, instead of directly accessing all services components. It is the responsibility of our service provider to ensure continuous updating of the availability of secure and effective services. The service repository continuously updates its database content from the service provider and uses this to store data for the services that are offered. The service registry and broker is responsible for registering services, coordinating the connections between the components, and providing the appropriate contract for services. It also reads the data from the registry and makes the right connection with other service components. The service constructor constructs or composes services from the multiple services available that cannot find compatible services through its service contract using the service registry and broker. The service provider ensures the availability of service matches with its services contract and advertises these to the service broker. In order to have efficient, effective services there is an urgent requirement to secure the interaction between the services and all the deployed architecture.

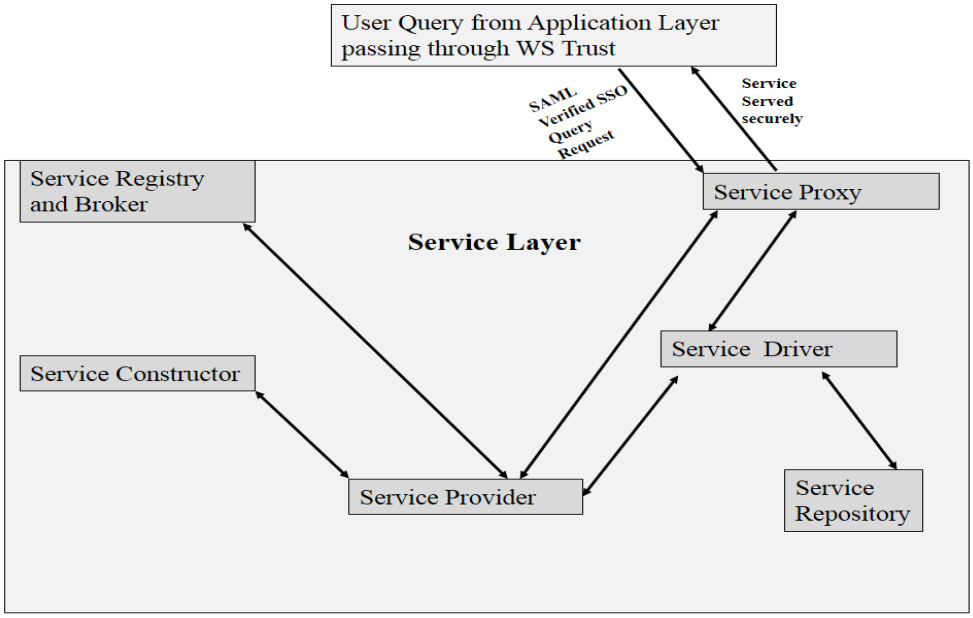

Figure-4.8: Services Layer Detailed View

As user queries requesting effective and secure services are received in the service layer, there is a requirement to describe some of the service components used in our architecture. Without understanding these components, we cannot build an effective architecture.

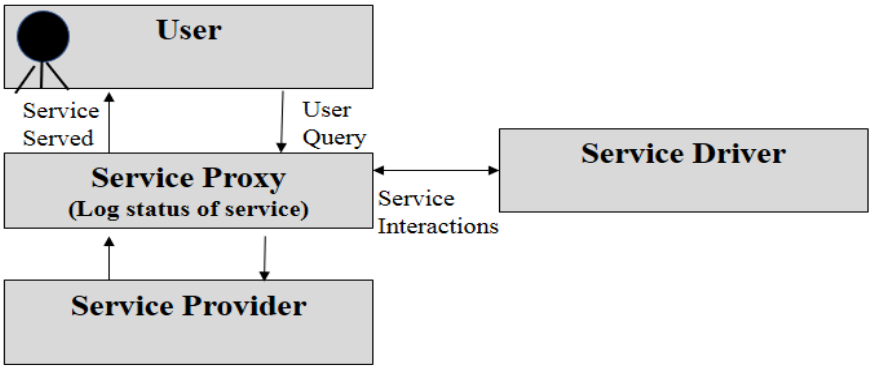

Figure 4.9: Service Proxy View

- **Service Facilitator or Services Proxy:** With reference to Figure 4.9, this shows a component of services architecture that works between the user request and the service used. It keeps track of all the users' requests that appear in the service layer and all the services to the satisfaction of the requestor. It keeps a log of the service status of the user request delivery and forwards the user request to find the appropriate service. It also works as the middleman between the user request and the service.

Main tasks include:

1. Working between the user request and the service.
2. Keeping a log of the service status of a user's request.
3. Forwarding the request to the service registry and broker to find the appropriate service for the user query.

Ensuring successful delivery of services to the user.

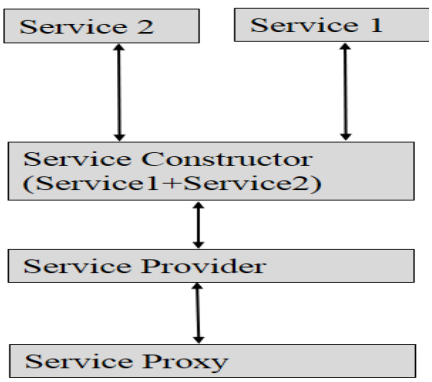

Figure 4.10: Service Constructor View

- ***Service Constructor:*** With reference to Figure-4.10, service construction constructs or composes services from multiple services that cannot find a compatible service with its service contract using the service registry and broker. For example, if a user requests "Select (Average (temperature)) from a temperature sensor where the temperature is > 100 and location= 'Acukland' ". In this query, we have to combine all temperatures from the sensor table and find a location that is equivalent to Auckland with temperature greater than 100. We must average the output of the query. This query is a combination of gathering the related data into a container and getting the related output.

Main Tasks are:

Composing services from multiple services.

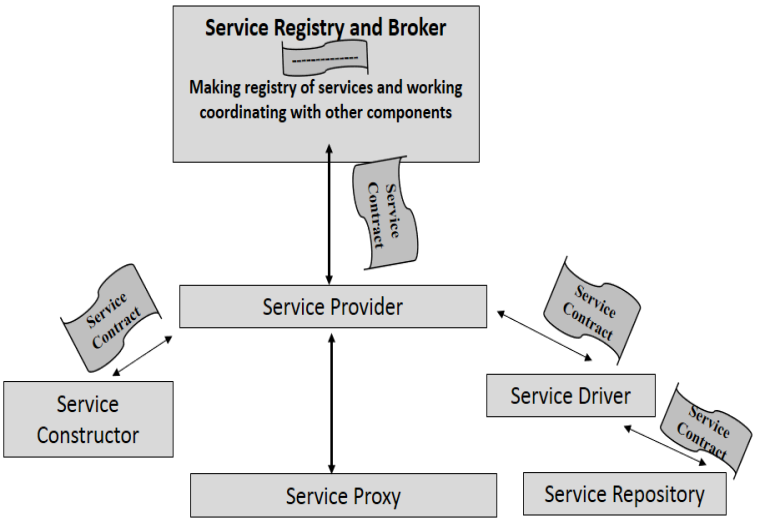

Figure 4.11: Service Registry and Broker

- ***Service Registry and Broker:*** With reference to Figure 4.11, the service registry and broker is responsible for registering services, coordinating the connections between the service components and providing the appropriate contract for the services. It also reads the data from the registry and makes the right connections to the other service components. The broker works between the service provider and the service requester and tries to find the appropriate service. Its main tasks include:
  1. Registering the status of each service
  2. Coordination and connecting between service components

3. Making appropriate contracts for the available service

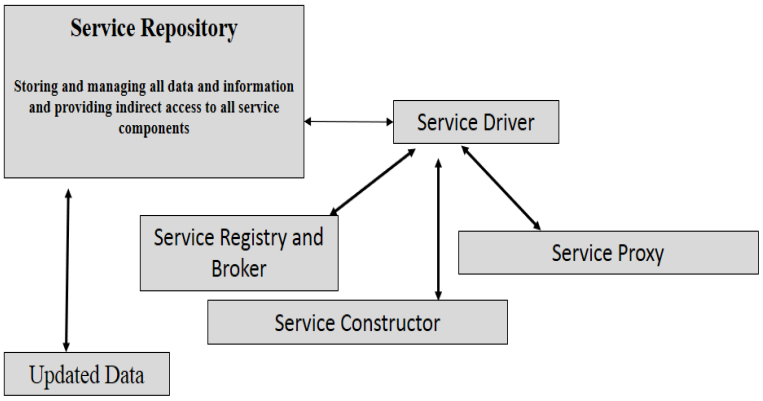

Figure 4.12: Service Repository

- ***Service Repository:*** With reference to Figure 4.12, the service repository in our architecture is responsible for storing and managing all sorts of information and data related to services, instead of directly accessing all services components. The service repository is used to facilitate indirect access to services and is used as a database where the user query will be serviced after finding a match.

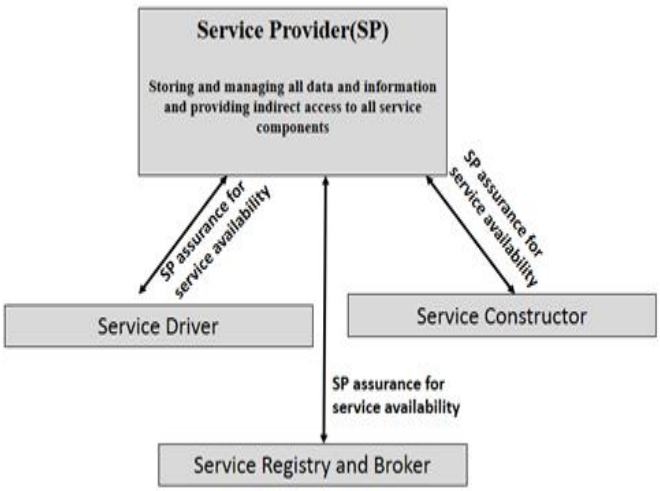

Figure 4.13: Service Provider View

- ***Service Provider:*** As seen in Figure 4.13, the service provider takes care of the availability of services and matches the services contract on behalf of the service broker. On the basis of available information from requestor, service provider is providing services to user.

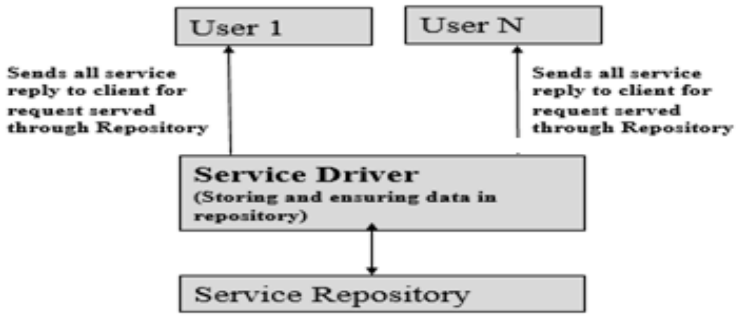

Figure 4.14: Service Driver Overview

- ***Service Driver:*** As shown in Figure 4.14, the service driver performs three responsibilities, which are as follows:
    1. Sends data on request to the client through the service proxy.
    2. Stores data in the service repository.
    3. Ensures the validity of the data in the repository.

- **Service Contract:** A service contract can be defined as a set of rules that define a service.

  These rules are represented in a data model, which is shown below.

SERVICE CONTRACT

(Agreement Initiator-User-ID). An agreement initiator is a party to an agreement. The initiator creates and manages an agreement on the availability of a service on behalf of either the service facilitator or service provider, depending on the domain-specific requirements. The initiator invokes the creating agreement or creating pending agreement operations from this specification. e.g In my SOA service initiator on behalf of ether service facilitator or service provider creates or invokes the creation of services agreement.

(Expiration Time): Expiration time defines a time when an agreement is no longer valid, and the parties are no-longer obligated by the terms of the agreement. e.g , In my SOA expiration time can be ten minute (service will be expired after 10 minutes) and service will be no more available after that.

(QOS): Guarantee terms define the assurance on service quality or service availability associated with the service described by the service definition terms. They refer to the service description that is the subject of the agreement and define service level objectives (for example the quality of service on execution needs to be met in my SOA), qualifying conditions (for example when some objectives must be met in my SOA) and business value expressing the importance of the service level objectives.

(Configuration-Name): Gets the name used to locate the service in application configuration file.

(Session-mode): Gets whether sessions are allowed, not allowed or needed.

(Type of Service): Services are major concept in any service oriented architecture. This explains a standard scheme for services. Services are organized as to what they do.; i.e., service function or purpose, for giving aid in ensuring both coverage and shared understanding. Other categorization schemes are also possible and helpful.

(Event based or Periodic): This attribute will explain the service is outcome of any event depending upon some set parameters or it can be periodic after some period.

(Cost): This attribute explains the cost of service accessed. This is depending upon lesser cost if service is easily accessible but high vice versa.

(Service Frequency): This attribute explains how many times service has been accessed i.e 10 times per second.

(Real Time or Buffered): This attribute explains the real time or buffered service. If service is accessed real time, then real time attribute is active otherwise buffered.

(Data Aggregation) can be yes or no depending upon information as can be gathered and expressed in a summary form, for purposes such as statistical analysis.

Security level- RSA Security) RSA Security Services help organizations to define security strategies and implement solutions to mitigate risk, ensure compliance, and accelerate business.

4.2.2 Detailed overview of effective services combined with SAML Security:

A user request received application layer after passing through the web security trust interface. The service layer received the user query from application layer. Afterwards forwarded to service layer for effective service. The service components: service proxy, repository, service registry and broker, constructor and service provider. The component 'service proxy' works in between user request and service served also keep record of all incoming and outgoing service. The service repository stores and manages data related to services, instead of directly accessing all the services components. It is the responsibility of our service provider to ensure the continuous updated availability of secure and effective services. The service repository continuously updates its database from the service provider and uses it to store the data that offers services, The 'service registry and broker' is responsible for registering services, coordinating the connections between the components, and providing appropriate contracts for the services. It also reads the data from the registry and makes the right connections to other service components. The service constructor constructs or composes multiple services, and if it cannot find a compatible service in its service contract, it uses the service registry and broker. The service provider ensures the availability of service matches with its service contract and advertises to the service broker.

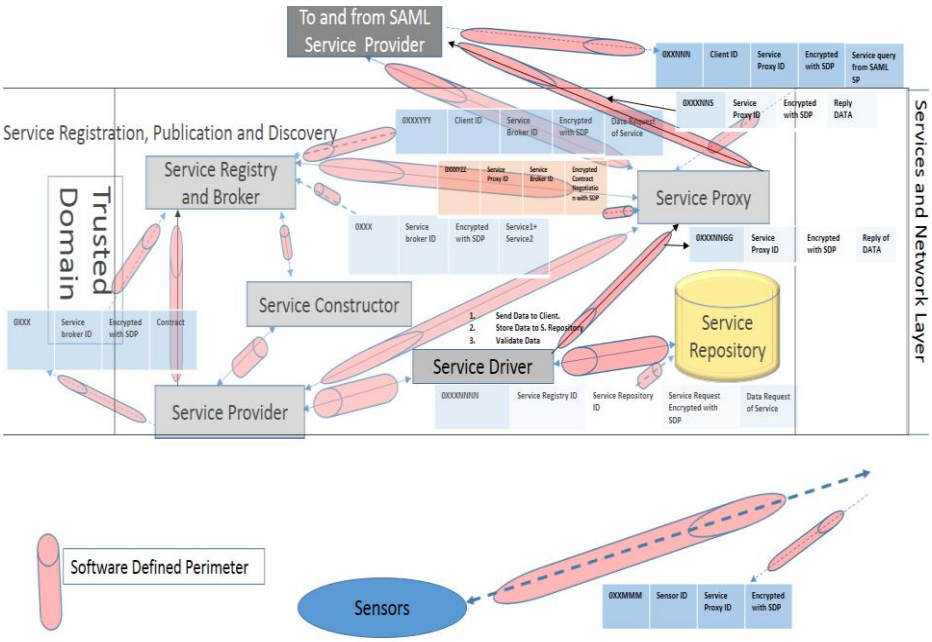

Figure-4.16: Detailed Overview of WS Trust and Services components

In Figure-4.16, the SAML secure user query packet from the application layer is shown as a client to service proxy.

All these services are available in a trusted domain. The service registry and broker registers all the services that are available online or in the service repository and coordinates the connections between the service components. Packets coming from a user query and received by service proxy may look like as below, also explore the following details:

Session ID: is unique for each session and issued by the operating system for each new session.

Source ID: is unique for each packet that has an identification number for each packet initiator.

Destination ID: is also unique for each packet that shows an identification number according to a destination. This is the destination for the packet.

SAML Protected user: DATA field of packets containing data or services as service replies that are secured by SAML.

DATA: a data packet that consists of a user query or a service reply.

Status of incoming user request: the status of a request that comes from user=1 if the request is arriving, but which in the case of a service reply becomes 0.

Status of outgoing request: is a status of request sent to a user with service reply and become 1 if service replied, but in the case of user request in progress becomes 0.

The general packet will be as follows

Session ID	Source ID	Destination Id	DATA

SAML protected user query: user query protected by the SAML WS Trust interface.

The packet below was received from the application layer. It has the following information:

Session Id=001

Source id= Client Id

001	Client-ID	Service Proxy ID	SAML protected User Query

The service proxy maintains a table like the one below:

Session Id	001
Status of Incoming user request	Send to Service provider
Status of outgoing service	Not served

The Service Proxy works between the user request and the service that results, and contacts the service provider to get the services. It always forwards the received request to a service broker and registry, through the service provider, by inserting a destination id= service provider identity.

The service proxy will then log this and try to find the appropriate service provider for effective service to the requested user. The user query comes from the application layer and requests a service, which look like the packet mentioned below:

Session ID: 002

Source ID: Service proxy identity number

Destination ID: Service provider identity number where the packet should reach. Packet consist of having destination ID and SAML protected query forwarded to service provider to find appropriate service.

002	Service Proxy ID	Service Provider ID	SAML Protected 'User Query'

As soon as query reaches the service provider, the service provider contacts the service registry and broker to find and register the service in the service registry, then the query is forwarded to the service registry and broker to find the appropriate service for the client.

The packet below shows:

Session ID=003

Source ID= Service provider identity

Destination ID= Service registry and broker identity

Three tasks are performed by the service registry and broker.

1. Registering the status of each service.
2. Coordinating and connecting between the service components.
3. Making an appropriate contract for the available service.

The service registry and broker checks the service availability if it is found in the register. Otherwise it tries to find the appropriate service component that can serve the request in an effective manner. This packet consists of a SAML protected user query. The activity that happens next starts if the service is not found. The service may be composed of two services, in which case the service registry and broker forwards the packet to the service constructor for further composition of service.

003	Service Provider ID	Service Reg. & Broker ID	SAML Protected 'User Query'

This request is forwarded to the service provider to find a service constructor component. The packet looks like the packet below:

Session Id=011

Source ID=Service registry and Broker

Destination ID= Service provider

The service provider further transfers this request to the constructor. Packet is shown below.

011	Service Reg. & Broker ID	Service Provider ID	SAML Protected 'User Query'

The services now need to construct or compose the services from a multiple service through its services contract. The query packet is forwarded to the service constructor to make up the components for the appropriate service The packet look like below:

Session ID= 005

Source ID= Service Provider Identity.

Destination ID= Service constructor Identity.

After constructing two or more services like service1 +Service 2, the reply is forwarded as a service proxy to log the status of the service and further forward it to the appropriate user.

005	Service Provider ID	Service Constructor ID	SAML Protected 'User Query'

If the appropriate service is found, the service reply is sent to the service provider and afterwards to the client through service proxy. The packet is forwarded to the service provider, as shown in the packet below.

009	Service Constructor ID	Service Provider ID	SAML Protected 'Service Reply or Response'

If the service is not found by the constructor, the packet will be forwarded to the service repository through the service driver, as shown in the packets below:

Session id=004,012 respectively.

004	Service Provider ID	Service Driver ID	SAML Protected 'User Query'

012	Service Driver ID	Service Repository ID	SAML Protected 'User Query'

The packet received in the repository performs the following tasks:

Checks the data in the repository to provide an appropriate service to the user.

If a service is found in the repository, it is forwarded to the service driver, as shown in the packets below:

Session ID= 008 and 013 respectively.

The service driver has the following three responsibilities:

1. Send data request to client through service proxy.
2. To send data to the service repository.
3. Ensuring the validity of data in repository.

The service driver performs the third task, which is to ensure whether the data is valid or invalid. If it is valid then it is forwarded to the service proxy for further service delivery to the requested user.

008	Service Driver ID	Service Proxy ID	SAML Protected 'Service reply'

013	Service Repository ID	Service Driver ID	SAML Protected 'Service reply'

The request is then forwarded to the service proxy, similar to the packet above having session ID=008 but with different session ID, and onwards to the client. The request contains a service response as an SAML protected response, as shown in the packet below with session ID=007.

007	Service Proxy ID	Client ID	SAML Protected 'Service reply'

We did not have a discussion about the sensing-network layer, as it is the responsibility of the service provider to ensure updated sensor data to our server.

4.3.1 Overview and Evaluation of the Architecture

After discussing all the details of this three-layered architecture, we attempted to simulate the architecture in a way that would ensure effective services and network security using a security access mark-up language. The application layer is responsible for handling the initiation and forwarding of user queries to the WS Trust Interface using SAML (security access mark-up language). This SAML WS Trust interface passes through the process of authentication between three role players such as SAML SP, the requesting user and IDP. This process was discussed earlier in detail.

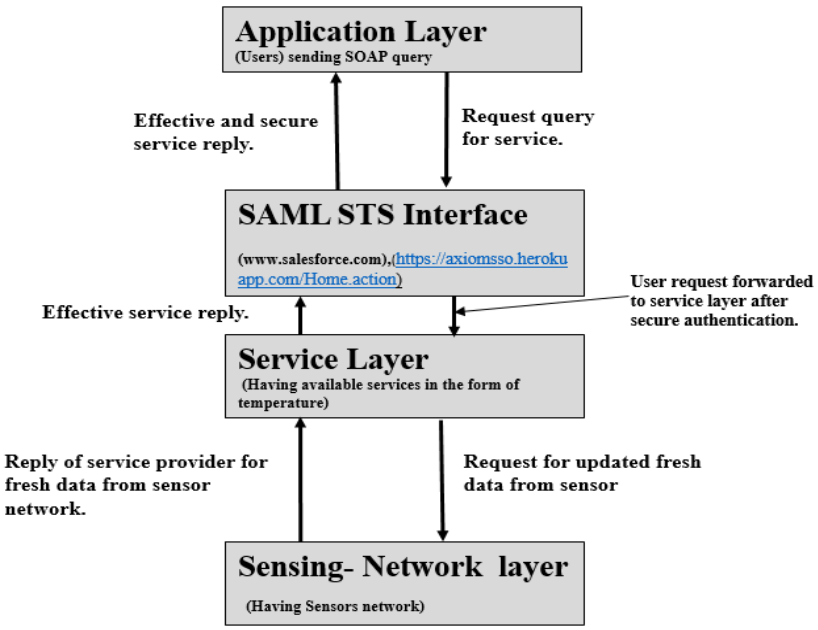

Figure 4.17: Overview of SAML Evaluation of Architecture

Referring to Figure 4.17, these three-key roles player are as follows:

1. SAML SP: www.salesforce.com
2. Token Provider: https://axiomsso.herokuapp.com/Home.action
3. IDP: www.salesforce.com

In the next step, we design an effective architecture that will be deployed further on the SAML secure infrastructure, as previously discussed. We designed our architecture in following different programing languages and then combined in single platform to perform our evaluation, based on different performance metrics:

1. PHP
2. Visual Basic
3. C#.

After constructing the three architectures using different programming languages, we deployed and configured the roles using the following role players in the WS Trust Interface. SAML SP= www.salesforce.com and takes the roles of the token provider and of the IDP in their respective servers. The main architecture was designed and deployed

by a free hosting web site =www.tvsupport-001-site1.btempurl.com. The host site is used as a platform to combine the three architectures into one. The following performance and quality metrics were tested in our platforms:

1. The average query processing round trip time for each programming language.
2. The average execution round trip time for the designed architecture.
3. The average database connection round trip time in each architecture.

We ensured effective service delivery in our design by taking care of all the service components and features, like the service registry and broker, the service proxy and the service driver. We also enabled an auditing feature in our database.

The performance was tested using the metrics mentioned above, using 1000 data values fetched from the database then 10000 values fetched from the database. As the service provider ensures updated data from sensing-network layer to our server, we simulate the random number (between 80 to 90) to simulate a sensor network.

The test initiating system configuration is as below:

Operation System: 8.1 Pro 64 bit

Hardware Specification: Make: HP, Ram: 8 Gb, Processor: Intel (R) Core™ -i5-4200 M Speed:2.5 Ghz.

Database Details:

The database used to test the architecture was designed to use different languages, which are as follows:

We used the MYSQL database for PHP. We integrated this database in our platform to gather the data on port no: 2207, which was used as an interface to populate our database in Microsoft SQL.

For Visual Basic and C#, we used MS SQL in our platform. The results after deployment are shown below:

Designed Architecture	Program Execution Time in milli-seconds	Query Execution time in milli-seconds	Database connection time in milli-seconds
PHP	0.0946140138	27.225001490	0.09265322542
Visual Basic	45.176875	10.135725	0.018175
C#	124.42355	47.8765	0.01775

Complexity based on programming languages: - PHP language is less complex as compared to C# and Visual Basic, because the syntax is simple and easy to understand. However C# is a more flexible language because of its programmability feature. Our conclusion is that different programming languages have their own performance parameters. PHP is simple and easy to learn but it does not provide flexibility in programming when compared to C#. In C# polymorphism, abstraction and inheritance properties exist, which make C# more flexible. Therefore, code reusability is available in C#. In our results, C# was faster to connect to a database connection when compared to the others.

Section-05 Architecture Evaluation

In this section, we present the detailed testing and evaluation of the proposed architecture.

5.1 Architecture Testing and Detailed Evaluation:

After discussing the details of three layered architecture, we tried to simulate the architecture in a way that would ensure effective services and network security using security access mark-up language. We used a WS Trust interface to provide network security to the available services.

5.2 Application Layer Evaluation

The application layer is responsible for handling the initiation and forwarding of user queries to the WS Trust Interface using SAML (security access mark-up language). This SAML WS Trust interface passes through the process of authentication between the three role players; SAML SP, the requesting user and IDP. In this phase the details of the role players are as follows:

With reference to Figure 5.1, our deployed architecture role players are hosted in different web servers and these key roles are:

SAML SP and IDP are hosted on the same site, 'www.salesforce.com', and the token provider is hosted on a different website, 'https://axiomsso.herokuapp.com/Home.action'.

The following are key role players of SAML infrasture:

1. SAML SP= 'www.salesforce.com'
2. Token Provider= 'https://axiomsso.herokuapp.com/Home.action'
3. IDP= 'www.salesforce.com'
4. User='Architecture is deployed on system having following specifications'

The architecture is deployed on this system:

Operation System: 8.1 Pro 64 bit

Hardware Specification: Make: HP, Ram: 8 Gb, Processor: Intel (R) Core™ -i5-4200 Processor Speed: 2.5 Ghz.

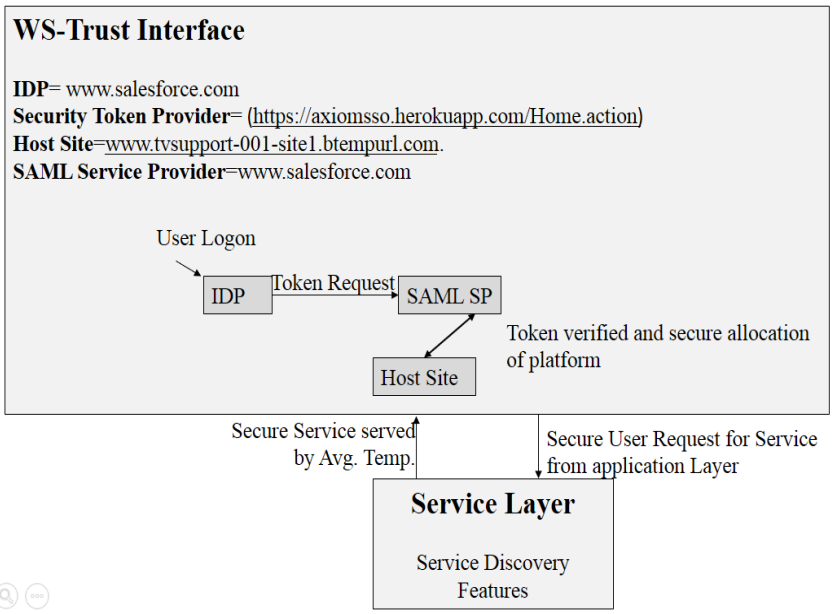

Figure-5.1: Architecture testing and detailed Evaluation

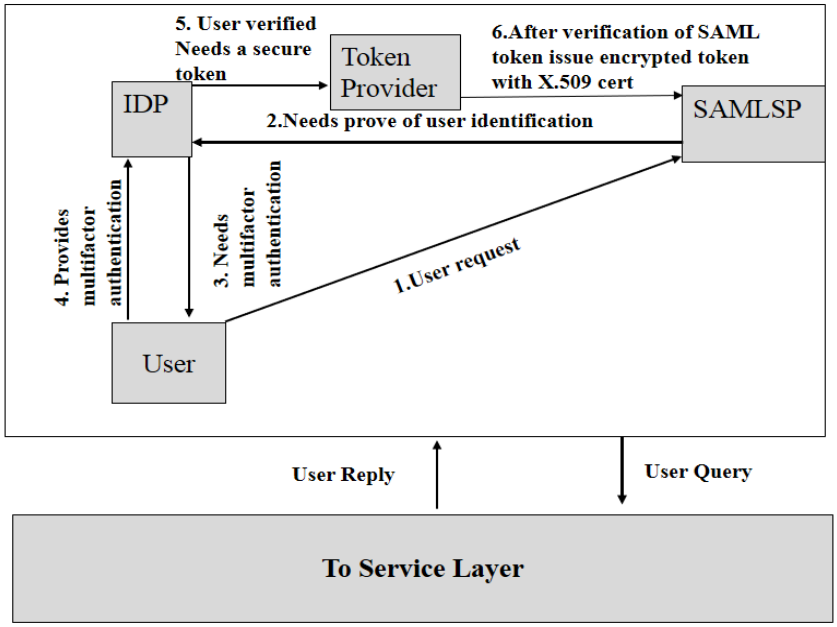

Figure-5.2: Application Layer Evaluation

As shown in Figure 5.2, the following steps take place during secure token exchange:

1. User sends requested service query to SAML SP.
2. SAML SP forwards the request to IDP for proof of identity of the user.
3. IDP requests user for proof of identity.

4. User replies with multifactor authentication with a mobile code and username/password.
5. User is verified at IDP, then IDP forwards the request to the token provider to provide a token.
6. After verification of token service, an encrypted token is sent to SP.

The user query is then sent to the service layer for effective service.

5.3 Service Layer

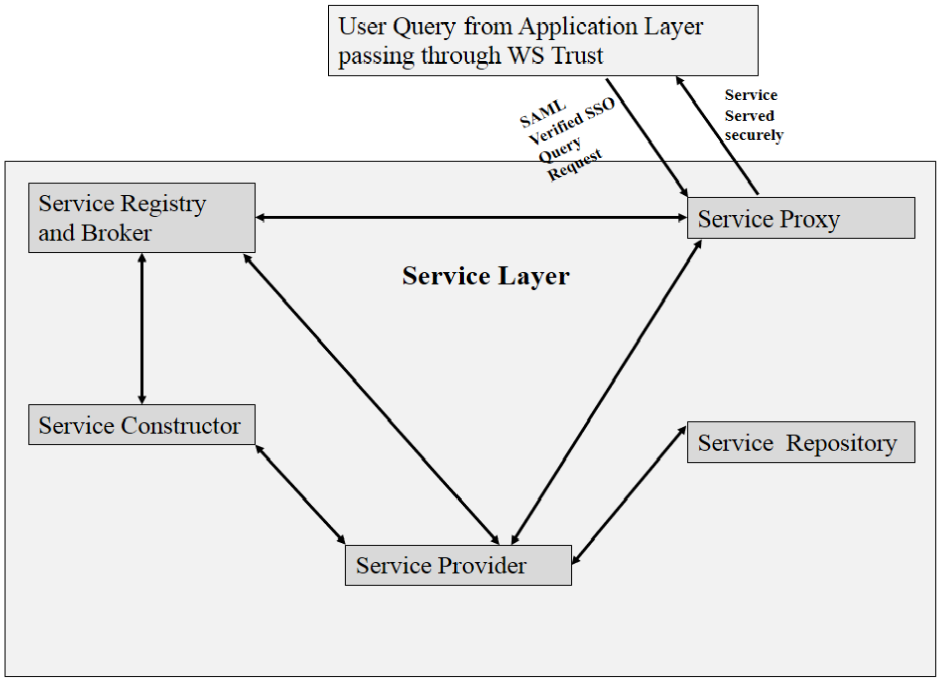

Figure-5.3: WS Trust Interface and Services Layer Interactions

The service layer receives the request from SAML SP with a user query for service access. After designing a secure architecture, we considered the provision of efficient and effective services using the SAML platform. Tests were conducted on the basis of some performance metrics.

We designed the architectures using the following three programming languages:

1. PHP
2. Visual Basic
3. C#

During the construction [28] of the architecture, we incorporated the roles of the service components, as in Figure-5.3.

The testing phase consisted of checking the performance and quality metrics on the three architectures that were designed using different programming languages.

1. Average connection round trip time of database.
2. Average performance of query processing round trip time.
3. Average total program execution round trip time.

Database Details:

Database Details for each architecture:

1. PHP (5.4.17) database is MYSQL
2. C# database is Microsoft SQL
3. Visual Basic database used Microsoft SQL

We integrated MYSQL database of PHP in the platform Microsoft SQL to gather the data from port no 2207.

Analysis with graphs and figures

We analysed the output of the performance and quality parameters using graphs configured in different programming languages. The Figure 5.4, is a platform for testing all three newly designed architectures in three programming languages. The PHP data is the architecture that was designed to use PHP Language; Blue Skies Weather was designed in C#, and Weather watch was designed using Visual Basic.

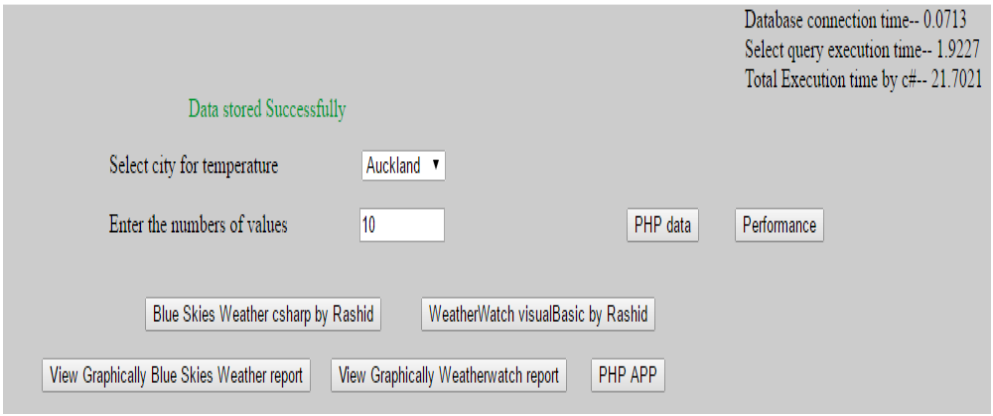

Figure-5.4: Evaluation Platform Overview

The Figure 5.5 below, was used to configure our IDP and multifactor authentication for mitigation of denial of service attack on web site 'salesforce.com' SAML identity provider use a multifactor authentication with our username, rashidmus@gmail.com, and a mobile

code received by a mobile phone with a combination that gave us access to the identity provider:

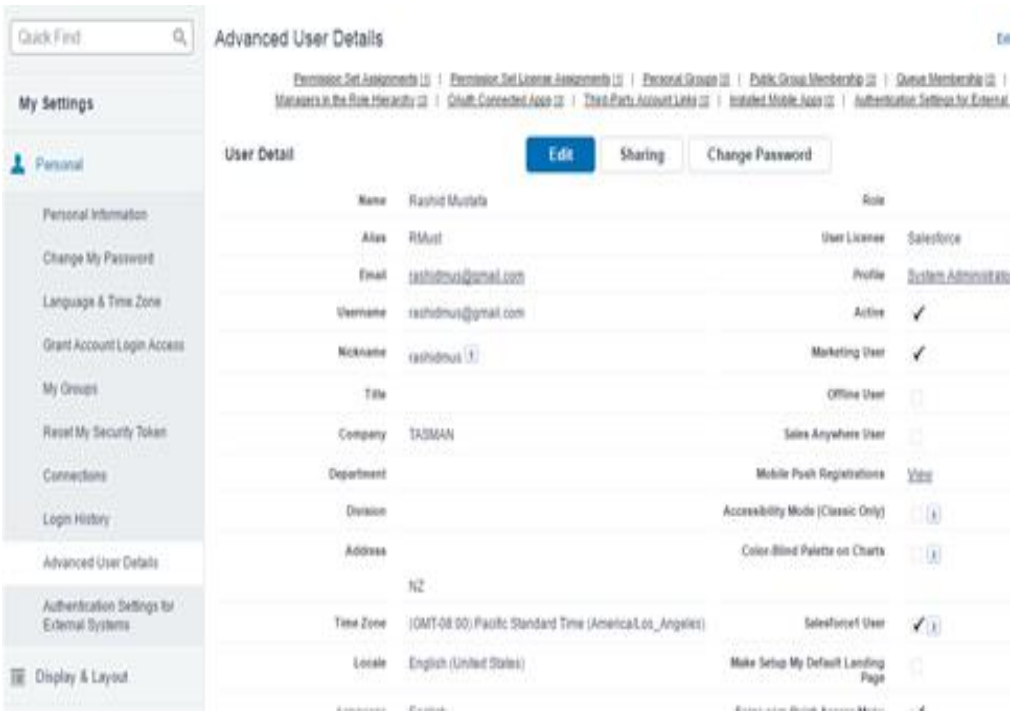

Figure 5.5: SAML User Multifactor Authentication

Figure 5.6 below shows the detail configuration of the Axiom interface used for the SAML token configuration. The Axiom server has a link: The screen shot attached below is showing the configuration interface for IDP and Axiom secure token service having web-link: https://axiomsso.herokuapp.com/Home.action

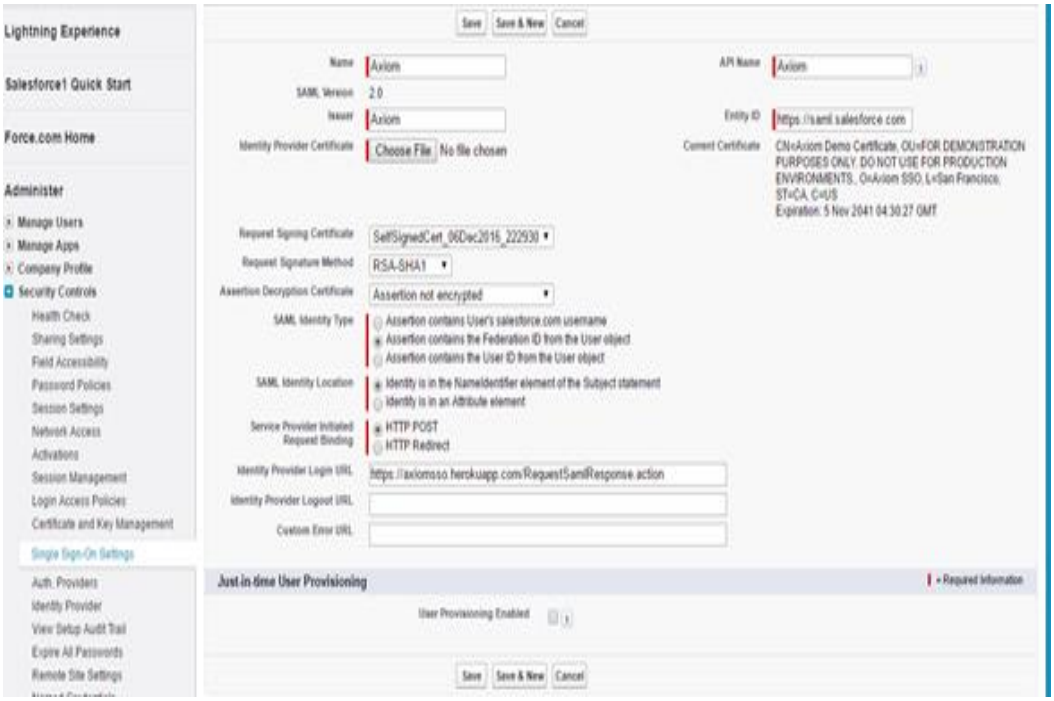

Figure 5.6: SAML Token and Axiom Interface Overview

Figure 5.7 shows a complete SAML secure token that was used to send for a secure single sign on authentication.

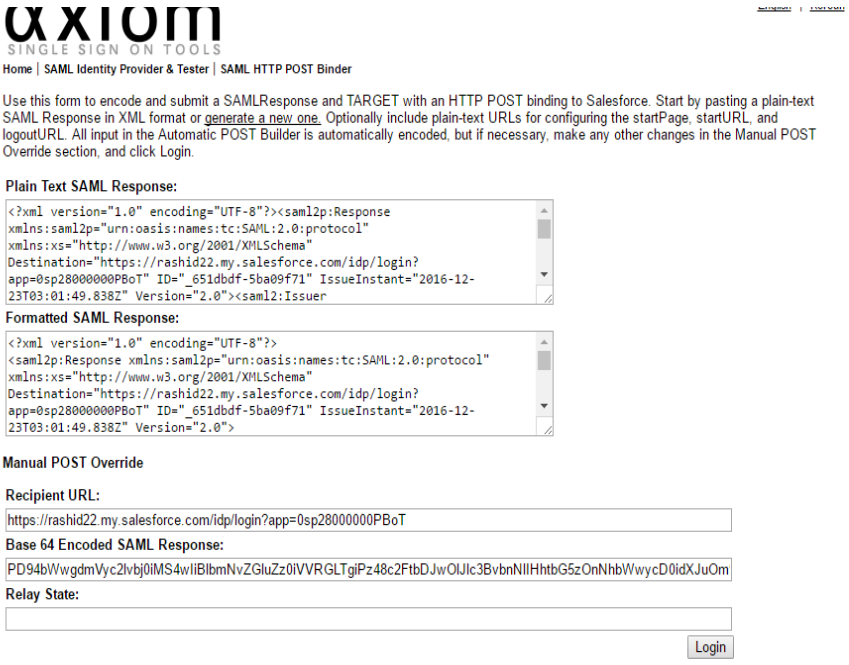

Figure-5.7: SAML Security Token Sample

5.4 Analysing Results

We did research on different research papers and considered some best practices that have been adopted by researchers to ensure the reliability of our architecture.

1.Abstraction: We considered the abstraction that hides the hardware details from the programmer in our architecture.

2.Programmability: Our architecture is flexible, we can easily program and re-program its features.

3.Scalability: Allows the database to grow in the future.

4.Modularity: Should be self-contained and independently deployed.

5.Discoverable: It should be registered for all services and discoverable for future services.

Performance: evaluate the performance of architecture.

Two performance attributes are mostly used by some researchers for evaluation of service oriented architecture

Test 1: Service A received by the server during normal operation - the system should process this request in B seconds.

Test 2: The roundtrip time for a request from service user A, keeping the response time from the server at less than B seconds.

Tests performed using three performance metrics (round trip time of database connection, round trip time of query processing and round trip time of total program execution time) on the SAML platform after considering the SAML token acquiring time from the server, including authentication time for a secure request, after passing the WS Trust Interface.

Some of the vulnerabilities were mitigated using the SAML WS Trust Interface.

1.Man in middle attack.
2.Message modification.
3.Message insertion attack.
4.DOS attack (this is protected after using multi-factor authentication).

Note: in multifactor authentication, we used a username and password verification plus mobile code sent using a mobile phone.

Evaluation performed on three architectures designed in different programming languages deployed using WS Trust of SAML.

The average database connection round trip times in milliseconds of three architectures designed on three different programming languages as follows:

Average Database Connection Time PHP	**Average Database Connection Time C#**	**Average Database Connection Time VBasic**
0.0926532254 ms	**0.018175 ms**	**0.01775 ms**

1.PHP: 0.0926532254 ms

2.Visual Basic: 0.018175 ms

3.C#: 0.01775 ms

The average query round trip times in milliseconds of three architectures designed on three different programming languages and deployed in SAML infrastructure as follows:

Average Query Processing Time PHP	**Average Query Processing Time C#**	**Average Query Processing Time VBasic**
27.225001490	**47.8765 ms**	**10.135725**

1.PHP: 27.225001490

2.C#:47.8765 ms

3.Visual Basic:10.135725

The average program execution roundtrip times in milliseconds of three architectures designed on three different programming languages and deployed in SAML infrastructure as follows:

Average Program Execution Time PHP	Average Program Execution Time C#	Average Program Execution Time VBasic
0.0946140138	**45.176875**	**124.42355**

1.PHP: 0.0946140138

2.Visual Basic: 45.176875

3.C#: 124.42355

The Comparison of average query execution roundtrip times in milliseconds of three architectures designed on three different programming languages and deployed in SAML infrastructure as follows in Figure 5.8 below:

If we compare all architectures with respect to querying round trip times, Visual Basic is best. After testing, the results verify that Visual Basic language takes less time to execute a query; 10.13ms compared to PHP (27.22ms) and C# (47.87ms). Therefore, Visual Basic is faster and best in this performance.

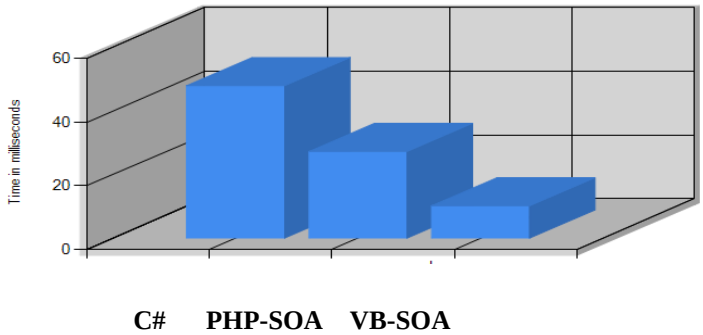

Figure 5.8: Average Query Execution Round Trip Time

***Comparing average program execution round trip times for three architectures using different programming languages with graphical view, as in Figure 5.9 below*: -** If we compare all architectures with respect to the average execution of program round trip times, PHP is best. The executing time of PHP is less when compared to other languages, taking about 0.09ms. Visual Basic takes 45.18ms and C# takes 124.42ms, so PHP is faster when compared to other languages on the basis of execution time.

AVERAGE PROGRAM EXECUTION ROUND TRIP TIME OF SOA USING DIFFERENT PROGRAMMING LANGUAGES

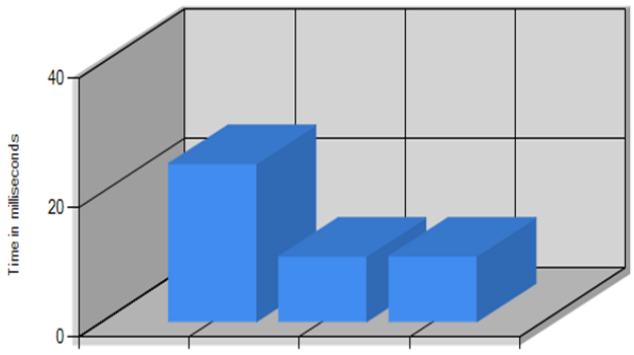

Figure 5.9: Average Program Execution Round Trip Time

Compare average round trip connection times of the database for three architectures designed in three different programming languages, as in Figure 5.10 below: - We compared the round-trip time of database connections for each architecture and found that C# was best. C# took 0.017ms to perform this task, while Visual Basic took much longer at 0.018ms. PHP took 0.092ms. C# is therefore much faster and takes less time to connect to a database.

AVERAGE CONNECTION ROUND TRIP TIME OF SOA DATABASE USING DIFFERENT PROG-LANGUAGES

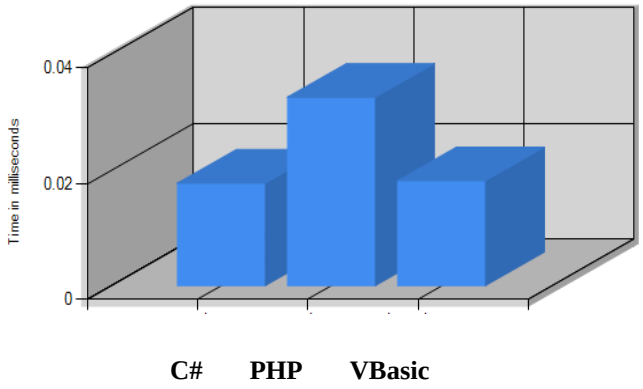

Figure-5.10: Average database connection round trip time of SOA

5.5 Complexity based on Programming languages: - PHP language is less complex as compare to C# and Visual Basic because Syntax is simple and easy to understandable. But C# is more flexible language as compare to others. It has concluded that, Different programming languages have their own performance parameters. PHP simple and easy to learn but it does not provide flexibility as compare to C#. In C# polymorphism, Abstraction and Inheritance properties exist, which make C# more flexible. So, code reusability is available is C#. In given result, C# is faster to connect to a database connection as compare to others.

Section-06 Conclusions

The concept of viewing the IoT as the Internet of services (IoS) has not been explored in detail in research. In this work we have moved forward with this idea and introduced an architecture that can enable the provisioning and consumption of IoT services in an effective and secure manner. While we consider the services in IoS as streams of information, we envision that in the future, IoS will play an active role in business and social process too, enabling them to interact and communicate by exchanging data. In our architecture, we proposed a layered architecture for handling the services in a secure manner where we consider the confidentiality, integrity and availability of data.

Our architecture for the secure provisioning of IoT services consists of three layers, the application layer, service layer and finally the sensing-network layer. Application layer handles the user's requests and the interface security issues, while the service layer handle the strategies for providing effective services requested by the user. We proposed the sensing-network layer, however in the current work we are assuming that service providers, provide our system with continuous stream of data. In future we would like to explore more on how that can be done and the issues regarding this. In the last section, we evaluated our architecture based on some performance metrics to ensure that our architecture can provide services efficiently to the users. In our current work we have performed the testing of the different components separately, however in future we would like to do testing on the entire system as whole, where we would be able to measure our system in a much better way. We have also currently considered IoS using a single server platform however in future we would like to distribute our architecture on multiple servers or even multiple cloud platforms to perform thorough testing.

Section 07 References

[1] Karen Rose, Scott Eldridge, Lyman Chapin, “Internet of Things an Overview,” Internet Society, [Online]. Available: http://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151022.pdf.

[2] Stroud, Forrest, “IoT - Internet of Things,” [Online]. Available: http://www.webopedia.com/TERM/I/internet_of_things.html.

[3] Inglada, Jordi, “Automatic recognition of man-made objects in high resolution optical remote sensing images by SVM classification of geometric image features,” [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092427160700055X.

[4] “Internet of things,” [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_things.

[5] “Good news for European citizens a significant step towards a digital single market without borders,” 2017. [Online]. Available: https://ec.europa.eu/commission/commissioners/2014-2019/oettinger/blog/good-news-european-citizens-significant-step-towards-digital-single-market-without-borders_en.

[6] Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, Stephen Wolff, “Brief History of the Internet,” [Online]. Available: http://www.internetsociety.org/internet/what-internet/history-internet/brief-history-internet.

[7] Hamza DAHMOUNI, Bertrand MORIN, Sandrine VATON, “Performance Modelling of GSM/GPRS Cells with Different Radio Resource Allocation Strategies,” [Online]. Available: https://www.researchgate.net/profile/S_Vaton/publication/4140406_Performance_modelling_of_GSMGPRS_cells_with_different_radio_resource_allocation_strategies/links/548177530cf22525dcb6163f.pdf.

[8] J. R. G. Doug Burger, “Billion-Transistor Architetcure,” [Online]. Available: http://www.inf.ed.ac.uk/teaching/courses/pa/Papers/billion_introduction.pdf.

[9] “Cisco Visual Networking,” 2016. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[10] “Morgan Stanley,” [Online]. Available: http://www.businessinsider.com.au/75-billion-devices-will-be-connected-to-the-internet-by-2020-2013-10?r=US&IR=T.

[11] Dr. Ovidiu Vermesan, Dr. Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Dr. Alessandro Bassi, Ignacio Soler Jubert, Dr. Margaretha Mazura, Dr. Mark Harrison, Dr. Markus Eisenhauer, Dr. Pat Doody, "Internet of Things Strategic Research Roadmap," [Online]. Available: http://internet-of-things-research.eu/pdf/IoT_Cluster_Strategic_Research_Agenda_2011.pdf.

[12] Marsan, Carolyn, "IOT an Overview," [Online]. Available: https://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151014_0.pdf.

[13] García-Macías, Edgardo Avilés-López · J. Antonio, "TinySOA: a service-oriented architecture for wireless sensor networks," 2009. [Online]. Available: https://pdfs.semanticscholar.org/168a/2a2321463d4130de12c0d9d9ca0062e5fc4e.pdf.

[14] Song, Yuanjun, "Security in Internet of Things," May 2013. [Online]. Available: http://www.diva-portal.org/smash/get/diva2:702223/FULLTEXT01.pdf.

[15] Nader Mohamed, Jameela Al-Jaroodi, "A survey on service-oriented middleware for wireless sensor networks," April 2011. [Online]. Available: https://www.researchgate.net/profile/Jameela_Al-Jaroodi/publication/220621590_A_survey_on_service-oriented_middleware_for_wireless_sensor_networks/links/0c960517505fc7d7fc000000.pdf.

[16] HAOJIN ZHU, SHANGHAI JIAO TONG UNIVERSITY RONGXING LU AND XUEMIN (SHERMAN) SHEN, XIAODONG LIN XIAODONG LIN, "SECURITY IN SERVICE-ORIENTED VEHICULAR NETWORKS," [Online]. Available: http://bbcr.uwaterloo.ca/~rxlu/paper/WC-Service.pdf.

[17] Li Da Xu, Shancang Li, "Internet of Things in Industries a survey," November 2014. [Online]. Available: https://www.researchgate.net/profile/Wu_He2/publication/270742269_Internet_of_Things_in_Industries_A_Survey/links/55fc355a08aec948c4b189f6.pdf.

[18] Luigi Atzori, Antonio Iera , Giacomo Morabito, "The Internet of Things: A Survey," May 2014. [Online]. Available: 'https://www.researchgate.net/profile/Luigi_Atzori2/publication/222571757_The_Internet_of_Things_A_Survey/links/546b36df0cf2f5eb180914e5/The-Internet-of-Things-A-Survey.pdf'.

[19] Fernandez, F, George C. Pallis, ""Opportunities and Challenges of Internet of Things for healthcare:"," 2014. [Online]. Available: 'http://ieeexplore.ieee.org/abstract/document/7015961/'.

[20] M. K. Swaroop Kalasapur, Mohan Kumar, B. Shirazi, “Evaluating service Oriented Architecture,” 2014. [Online]. Available: ‘http://ieeexplore.ieee.org/abstract/document/7015961/’.

[21] W.T. Tsai, Xinyu Zhou, and Yinong Chen, Xiaoying Bai,, “On Testing and Evaluating Service-Oriented Software,” [Online]. Available: ‘http://repository.cmu.edu/cgi/viewcontent.cgi?article=1327&context=sei’.

[22] Yaser Jararweh, Mahmoud Al-Ayyoub, Ala’ Darabseh, Elhadj Benkhelifa, Mladen Vouk, Andy Rindos, ““SDIoT: a software defined based internet of things framework,”,” 2015. [Online]. Available: ‘http://s3.amazonaws.com/academia.edu.documents/41401553/SDIoT_A_Software_Defined_based_Internet_20160122-19325-li6dq3.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1489489289&Signature=cLcx8EudjNZxeifM8i3b8lcRCVA%3D&response-content-disposition=inline%.

[23] Stantchev, V, Services Research Group, “Performance Evaluation of Cloud Computing Offerings,” 2009. [Online]. Available: https://www.researchgate.net/profile/Vladimir_Stantchev/publication/232630085_Performance_Evaluation_of_Cloud_Computing_Offerings/links/566af3e408ae1a797e396af9.pdf.

[24] A. Arsanjani, “Service-oriented modeling and architecture,” 2004. [Online]. Available: https://pdfs.semanticscholar.org/c5b0/e0c46d33e85964240822b40ee513f3adb902.pdf.

[25] Mark Endrei, Jenny Ang, Sook Chua, Ali Arsanjani, Sook Chua, Philippe Comte, Pål Krogdahl, Min Luo, Tony Newling, “ServiceOriented Architecture and Web Services,” 2004. [Online]. Available: https://www.researchgate.net/profile/Ali_Arsanjani/publication/200167132_IBM_Patterns_service-oriented_architecture_and_web_services/links/5488c92e0cf289302e30b950/IBM-Patterns-service-oriented-architecture-and-web-services.pdf.

[26] Armin Haller, Emilia Cimpian Adrian, Mocan Eyal Oren, Christoph Bussler,, “WSMX - A Semantic Service-Oriented Architecture,” [Online]. Available: https://www.researchgate.net/profile/Armin_Haller/publication/4186806_WSMX_-_A_semantic_service-oriented_architecture/links/0deec530ed1e04b794000000/WSMX-A-semantic-service-oriented-architecture.pdf.

[27] Ronald Monzillo, Chris Kaler, Anthony Nadalin, Phillip Hallaem-Baker, “Web Services Security: SAML Token Profile 1.1,” 2005. [Online]. Available: https://www.oasis-

open.org/committees/download.php/15144/wss-v1.1-spec-draft-SAMLTokenProfile-09.pdf.

[28] Jack Beaton, Brad A. Myers, Jeffrey Stylos, Sae Young (Sophie) Jeong, Yingyu (Clare) Xie, “Usability Evaluation for Enterprise SOA APIs,” [Online]. Available: http://s3.amazonaws.com/academia.edu.documents/30665703/SDSOA-p29.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1489708315&Signature=L7lBiauWA3fs%2BXs14FFnwRxkrHM%3D&response-content-disposition=inline%3B%20filename%3DUsability_evaluation_for_enterprise.