Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Aisha Saeed

Authentication and Authorization Modules for Open Messaging Interface (O-MI)

Master's Thesis
Espoo, September 17, 2018

Supervisor: Professor Kary Främling
Advisor: Narges Yousefnezhad M.Sc. (Tech.)

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author: Aisha Saeed

Title:
Authentication and Authorization Modules for Open Messaging Interface (O-MI)

Date:	September 17, 2018	**Pages:**	82
Major:	Computer Science	**Code:**	SCI3042
Supervisor:	Professor Kary Främling		
Advisor:	Narges Yousefnezhad M.Sc. (Tech.)		

With the constant rise of new technology, developments in the fields of computer science, wireless networks, storage capabilities and sensing possibilities along with the demand for continuous connectivity have lead to the formation of the Internet of Things (IoT) concept.

Today, there are numerous organizations working on the IoT technology aimed at developing smart products and services. Each company proposes its own methods directed for a particular field of industry thus, it ends up with having several protocols. This has poorly followed the concept of a unified system. The Open Group attempted to address this issue by proposing Open Messaging Interface (O-MI) and Open Data Format (O-DF) protocols and claimed O-MI to be an IoT messaging standard as that of HTTP for world-wide-web (WWW).

The proposed protocols have been designed to ensure robust development, data standardization, and required security level. However, the security model needs to be upgraded with the recent security techniques. This thesis attempts to specify appropriate authentication and authorization (access control) mechanisms that manage various consumers and provide functionalities that fit into O-MI/O-DF standards. The thesis first discusses several challenges regarding IoT security and then different authentication and authorization techniques available today. It then describes in detail the design decisions and implementation technicalities of the autonomous services created for the reference implementation of O-MI and O-DF.

Keywords: Authentication, Authorization, Internet of Things, O-MI, O-DF, Security rules, Smart Devices

Language: English

Acknowledgements

This Master's thesis research fulfills one of the requirements for Master's in Science degree program in Computer, Communication and Information Sciences (CCIS).

I am thankful to the Allah Almighty Who provided me with the strength and capability to learn, comprehend and accomplish this thesis work. I owe my deepest gratitude to my supervisor, Professor Kary Främling for trusting me and providing me the opportunity to work on this research project, for his continuous feedback and constructive guidelines. This thesis would not have been possible without the passionate participation and input from my advisor, Narges Yousefnezhad and ASIA team members: Tuomas Kinnunen, Asad Javed, and Manik Madhikermi.

Lastly, I would like to thanks my parents and my aunt Tayyaba for their love, prayers, continuous support and motivation. I would also like to express my special gratitude to my dear friends and colleagues who have been a constant source of encouragement and guidance.

Espoo, September 17, 2018

Aisha Saeed

Abbreviations and Acronyms

ACL	Access Control List
AD	Active Directory
API	Application Program Interface
ARPANET	Advanced Research Projects Agency Network
BLE	Bluetooth Low Energy
CA	Certificate Authority
CBA	Context Based Authentication
CSV	Comma-Separated Values
DAP	Directory Access Protocol
DDoS	Distributed Denial of Service
DIT	Directory Information Tree
DN	Distinguished Name
DNS	Domain Name System
FDE	(Full-Disk Encryption
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IdP	Identity Provider
IIS	Internet Information Services
IP	Internet Protocol
IoT	Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation
JWT	JSON Web Token
LDAP	Lightweight Directory Access Protocol
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
GPO	Group Policy Objects
LTE	Long Term Evolution
MPTFA	Mobile Phone Two-Factor Authentication
MVP	Minimum Viable Product

NAT	Network Address Translation
NFC	Near Field Communication
O-DF	Open Data Format
O-MI	Open Messaging Interface
OASIS	Organization for the Advancement of Structured Information Standards
OS	Operating System
OTP	One Time Password
OU	Organizational Unit
P2P	Peer-2-Peer
PII	Personally Identifiable Information
PKI	Public Key Infrastructure
QR	Quick Response
REST	Representational State Transfer
RDN	Relative Distinguished Name
RFID	Radio-Frequency IDentification
SAML	Security Assertion Markup Language
SIM	Subscriber Identity Module
SP	Service Provider
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
SSO	Single Sign On
TCP	Transmission Control Protocol
TTL	Time To Live
URL	Uniform Resource Locator
WiFi	Wireless Fidelity
WWW	World Wide Web
XML	Extensible Markup Language

Contents

Chapter 1

Introduction

This chapter provides an introduction to the thesis research. It begins by discussing the motivations for performing this work, then some background information regarding the research topic and the relevant framework is presented. Finally, thesis structure is described.

1.1 Motivation

The beginning of the Internet started with the establishment of electronic computers in 1950. ARPANET (Advanced Research Projects Agency Networks), developed in 1969, is considered the basic foundation of the Internet and the first network that utilized the TCP/IP (Transmission Control Protocol/Internet Protocol) protocol [38]. With the evolution in computer technology, several standards were adopted such as HTTP (Hypertext Transfer Protocol) and HTML (Hypertext Markup language). This resulted in the development of the World Wide Web (WWW) that linked HTML files with the information systems which can be accessed by any device on the network.

Development of the Internet made the world smaller and the connection easier. Over the past 23 years, the Internet has evolved to a great extent with around 4000 million users [48]. Today, the mobile technology and the web has changed the way in which people utilize the Internet. It is the primary path that people follow to access all kinds of information, quick buying/selling, fast communication, and instant entertainment. The technology is advancing every day and is taking control of every small object in our society. As we add more devices and information/data to the systems or on global networks, more security risks are involved. For instance, using online stores for shopping presents effortless opportunity to purchase any product in any part of the world but at the same time, such online payment and banking systems are

constantly utilizing many techniques to prevent the cyber attacks [31]. Not only the bigger service provider firms but ordinary service users are also being affected. Various methods utilized by attackers are SQL injection attack, phishing attack, insider threat, and misuse of privileges and physical attacks. Although education regarding cyber attacks is being imparted, currently, cyber attackers have become very advanced and are adapting their ways to the new security techniques and are attempting to make them inoperative.

In today's world, researchers and practitioners around the globe have given much attention to the IoT (Internet of things) technology. Internet of things refers to the networked communication between everyday objects and their users and as well as with other devices. Due to rapid progress in technical research, IoT is aiming to provide exceptional opportunities to all the apps that promise to achieve the quality of life [58].

IoT products will also be increasing in the coming years and thus, we need to develop highly secure architectures. The security of such products is of significant importance as they may contain personal, private and confidential user/customer data. Therefore, more machines connected to the Internet, more cyber threats will be involved. However, businesses today have a fear of loss and consequently, provides as soon as possible the Minimum Viable Product (MVP) to the market without taking into account the product's security aspects. Ransomware (a type of malware attack that locks or encrypts the user files or systems and demands money in return) is also one of the major threats for IoT devices.

1.2 Research Framework and Goals

IoT security is a broad and a complex topic. Several protocols and standards have been developed so far as shown in figure 1.1 depicting the IoT protocol landscape. It is becoming messier with every new device and is much more dynamic when it is compared to the traditional web/Internet.

A number of protocols have been created and designed for particular use cases. All these protocols have different requirements and characteristics and are aimed to provide the best quality service in a specific field of industry. This enables to develop efficient and one-shot applications. However, it does not meet the vital requirement to have a unified IoT standard. Standardization is the intelligible solution to the growth of the vertical market and customized APIs. The goal is to achieve a seamless and perfect IoT demands protocols to be efficient enough to be independent of the domain and adapt themselves to various frameworks and models. An attempt was being made by The Open Group to provide the solution for this problem (partly

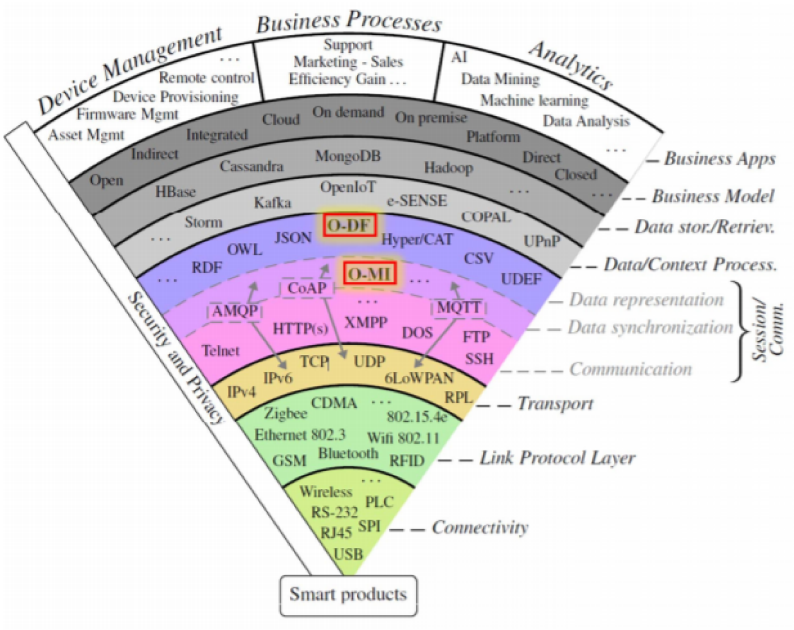

Figure 1.1: IoT protocols/standards landscape. [27]

at application stage). It issued Open Messaging Interface (O-MI) and Open Data Format (O-DF) as official standards in 2014 which probably makes them relatively the first standards developed explicitly for the IoT (which is not the case for other messaging protocols such as MQTT, XMPP, CoAP and AMQP) [11]. This solution aims to achieve the same impact as the one gained by HTML and HTTP for the World Wide Web.

The security model [60] developed for such protocol has been outdated and does not satisfy the current security requirements. The aim of this thesis is to implement the second version of existed security model compliance with all necessary features needed. Further details regarding older and newer versions are discussed in chapter 6. In accordance with framework and background discussed, this thesis attempts to provide the solution to the following queries related to the IoT research.

1. How do the devices in the Internet of Things interconnect with each other? What is a common messaging standard proposed and how does it assist in communication?

2. What is O-MI standard? What is O-DF standard? How well does it adapts each requirement related to the particular IoT messaging standard?

3. How to implement this solution and integrate it together in an application along with the security aspects (authorization and authentication modules)?

The above research queries were formulated due to the need for a general standard that is highly required to fulfill all the specifications of the Internet of Things. Additionally, a thorough analysis of O-MI and O-DF reference implementations is also essential. Therefore, a number of important instructions that are described in this work are as follows:

- **Analysis of the Background Topic:** It includes the literature review on the subject of the Internet of Things. It focuses on the introduction of the IoT industry, its important aspects, and related issues. This thesis formulates an analysis of various existing security mechanisms and describes the requirements of the work in a procedural and understandable method.
- **Study of the Reference Implementation:** Along with the examples of various use-cases, this study describes the O-MI and O-DF standards reference implementation. It also discusses the current security techniques being utilized and the urge to develop the authentication and authorization (access control) mechanisms for the reference implementation.
- **Design and development of Security Module Version 2:** This is the main area of the thesis which discusses the comprehensive procedure of gathering requirements for security features, its design and the implementation of the complete security model.

1.3 Thesis Structure

This thesis includes seven chapters. After the introduction chapter, the second one discusses the Internet of Things subject in detail, describing the requirements, related problems, and concerns. The third chapter analyzes multiple authentication and authorization techniques that are accounted appropriate in the Internet of Things scenario and pros/cons are being described. In Chapter 4, The Open Group messaging protocols are being discussed in detail. O-MI and O-DF standards essential principles and design concepts are presented. Chapter 5 describes the reference implementation of O-MI and O-DF and represents the working of such protocols in a specific app. Chapter 6 is the most important part of this thesis as it discusses various authentication techniques, the requirements for O-MI node and security

model, design, schemes, and implementation specifications. Finally, the last chapter concludes this thesis by providing the summary of the findings, the analysis of the research, and some future concerns in this work.

Chapter 2

Internet of Things (IoT)

2.1 Introduction to IoT

Internet of Things is also called as Internet of objects or Cyber-physical systems (CPS) [58] [59]. Binding the physical bodies with the Internet has shortened the distances and remote access to the sensors has been made possible. If the retrieved data is combined with the already available information on our web services, then we are entering a totally new era of technology where such services are provided to develop new kinds of products which are far beyond those offered by the conventional "isolated" systems. The new system called smart object is a CPS or an embedded system including a physical device and a computing object that is capable of managing the data from sensors and assists in establishing a wireless link with the Internet. For instance, a smart refrigerator that takes notice of the expiry date of the items and checks their availability periodically. Additionally, it places an order in nearby shops for new products if they have reached their ending limits [44].

The concept behind the Internet of Things has first emerged in 2002 in a publication [33] where this idea was named as intelligent products. Then in 2005 [59], this notion was referred to as the "Internet of Things" since then the world has visualized the usage of such smart objects. Communication and interaction can take place not only among the people but also between the humans and devices. Even device-to-device connectivity is also possible. Thus, this can take the world to another higher level of intelligent systems and distributed frameworks. Having abilities of action, connectivity, and sensory, network-enabled smart devices have assisted a number of applications in various areas of industries such as smart buildings, efficient transportation systems, social communications, personal domains, and health centers [59]. Everything that we see around is going to be connected together such as as-

sisted driving (cars, trams, buses and trains equipped with suitable sensors), mobile ticketing (access transportation services via NFC tags), remote tracking of environmental parameters (monitoring of farms across several kilometers) and intelligent maps (using NFC techniques). One can even consider various sensors in one's home and office, for instance, temperature, light, and humidity sensor devices. IoT utilities can also serve us in industrial areas by deploying RFID tags and enabling the automation of plants. IoT sensors can also help us to know about the possible thefts, for example, if an object is moved from its place without the user's permission to some restricted zone [22].

Several organizations have begun to launch various IoT services and devices. The world has witnessed numerous headlines of IoT products acquisitions, for example, Google bought Nest for about 3.2 billion dollars, SmartThings purchased by Samsung and Nest took over Dropcam [57]. Not only the practitioners but also the politicians are acknowledging the importance of IoT at an increasing extent in terms of business opportunities. IoT services are modifying the behavior of the companies and the corresponding consumers. According to some estimates, Internet of Things will progress worth 7.1 trillion dollars by the year 2020 [57] [49]. The figure 2.1 shows that the world population would be approximately 7.6 billion in 2020 whereas there will be a huge increase in IoT devices approximately 50 billion products, far more than the 25 billion IoT items produced in the year 2015 [32]. According to a survey data which represents the interests of the organizations for IoT products, around 15 trillion dollars will be invested in developing IoT services between the years 2017 to 2025 [49].

Given that there will be more IoT devices in the year 2018, we have seen a decline in terms of the cost of various IoT components. There are now low-priced sensors available, bandwidth cost decreased to 40x and processing power has reduced to about 60X in the past 10 years [39] where objects are not merely connected with each other but also have the ability to act smartly with the new generated or received information. Other drivers that helped in popularizing the IoT technology are: smartphones that acts as remote controllers of our devices (e.g., connected cars, homes, health care), extended Wi-Fi coverage that is mostly free or provided at very low price, developments in big data field as enormous data is expected to be generated with IoT and the emerging use of IPv6 which can support 128 bit addresses and translate to about 3.4 x1038 addresses far more than that of IPv4 (32 bit) that provides 4.3 billion addresses only [39].

Large-scale companies, such as Microsoft, IBM, Samsung, and Google, have an immense influence on our lives and now have impacted substantially the Internet of Things ecosystem. Such major businesses and tech giants pos-

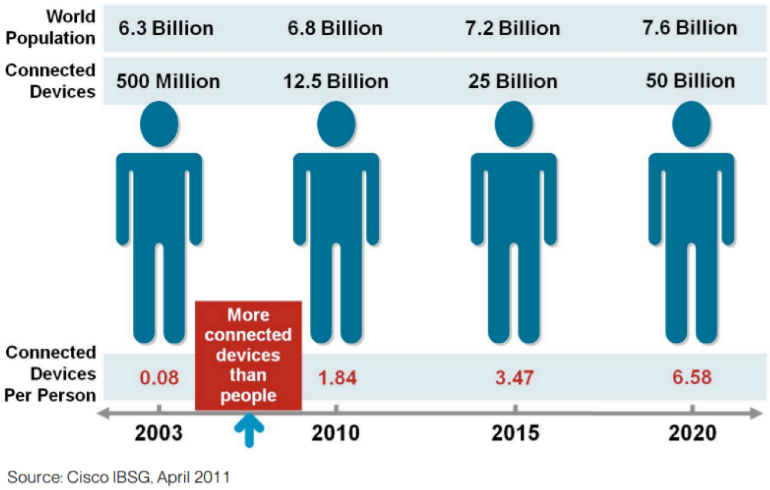

Figure 2.1: IoT connected devices [32]

sess all the required resources and funds and have a significant impact on the present and future development of technology. These enterprises are capable enough to create totally new products and services, integrate them with other models and provide support for the present techniques being utilized such as for Android OS and iOS.

There are many small and medium-sized enterprises (SMEs) and research-based communities that manufacture various protocols/devices and quite often produce new technical ideas and innovations. However, their domains are narrow and different protocols established by various producers have interoperability issues. It has been seen that a software developed by one service provider was not able to run on different operating systems or on different versions of hardware and software components. Various communication standards, distinct connection frameworks and incompatible connectors were observed in all objects of IoT. The evolution of technical solutions provided by SMEs is shadowed due to the lack of insights to how to immerse them with the existing products. The large enterprises discussed above also play a vital role in this interoperability issue. They have their own protocols, operating systems, and defined standards. They all attempt to be known as the most-efficient/best solution providers and in this race, popularize their services and makes them secure from the public, thus, establishing more problems of IoT incompatibility and consumerization.

On the other hand, we have the perfect example of standardization i.e. World Wide Web (WWW) which has adopted an open standard system of

technology. This is now considered essential to have the similar open source techniques for the Internet of Things, otherwise, the technical world would continue to see the big tech giants progressing day by day without realizing about the ecosystem fragmentation.

Today, several organizations realized the need for a standardized protocol for all IoT objects and hence, they are working on such projects. One such protocol is O-MI (Open Messaging Interface) and O-DF (Open Data Format) standards produced by The Open Group. Further chapters will discuss them in detail and will also describe the reference implementation working. Prior to this, the current chapter will highlight some of the challenges and related concerns in IoT and the importance of security in the Internet of Things.

2.2 Challenges and related concerns

All the objects that are bound with the Internet are expanding increasingly. This led to the introduction of the IoT concept. The ecosystem of Internet of Things is comprised of not only the smart devices (small physical objects capable of communicating and computing with memory and energy requirements) but also the identifying codes and labels that are utilized to distinguish devices globally and uniquely. There are multiple technologies that are currently serving these purposes such as LTE (Long Term Evolution), 6LoWPAN, WiFi Low power, RFID (Radio Frequency Identification), NFC(Near Field Communication) and QR (Quick Response). As Internet devices (appliances, tablets, laptops, and smartphones) are increasing, Internet protocol IPv6 is being utilized in order to achieve distinguish address of every object. Thus, increasing the possibilities of integration and end-to-end connected society for the Internet of Things [40].

The technological challenges that the IoT industry is facing now are related to connectivity, standardization, security, and compatibility. Connecting billions of devices in near future would be one of the difficult tasks and it will challenge the present implementations of IoT structures. For the connectivity of various objects, authentication, and authorization, many existing systems depend on the client/server or centralized models which are considered suitable for a small number, i.e., 10s, or 100s or 1000s of nodes. However, for larger numbers, these paradigms will become a bottleneck and would require massive cost investments to support servers to be able to handle a large amount of data exchange and in an unfortunate scenario, a server failure can turn down the whole system. A decentralized system of IoT can be one of the solutions where critical tasks are handled on the edges while information assembling process and computation to be done at the cloud servers level. A

P2P (peer-to-peer) system having no brokers where machines identify each other and deliver data; and utilizing mesh networks (which has reduced failures) can be some of the other solutions [30].

In order to process, store and maintain the information received from sensor devices, protocols such as communication/network protocols and data-gathering techniques are utilized. Relational databases are used for structured information and have a proper querying mechanism, for instance via SQL whereas, for unstructured information, non-relational databases are utilized that does not support a standardized way of querying. Another challenge faced for the standardization in IoT is to adopt the right technical expertise to take full advantage of new upcoming data aggregation tools and have the skills to layout, perform and handle the systems [23].

There are also the privacy challenges faced by IoT. Sometimes the information gathered by an IoT object can be very sensitive and usually secured by some legislation. However, it has been seen that appropriate steps are not taken to avoid sharing the protected data with the other service providers. This info should be scraped off or it should not associate any PII(Personal Identifiable Information). The producers or vendors should take extra precautions of the user's privacy. There can be a situation where insensitive data from an object (for example, a smart fridge) can merge with information from other appliances where there are chances of sensitive data leakage [30]. Many issues emerge due to a lack of consciously using the IoT devices. This is wide spreading in consumer appliances, for instance tracking gadgets in cars, mobile devices and sometimes in TVs. If we consider the example of a smart television, then there are chances that voice/vision recognition sensors are embedded in the TV which continuously hear the conversations around, can note down the user activities and then impart the info to the backend cloud services that can be any third party involved. The assembling of such data opens legal challenges against the information privacy and protection law. There can be many IoT situations that will include IoT objects deployments and information gathering services globally and thus, there is a need to cater such scenario by developing a broad IoT privacy model [23].

Internet of Things also suffers from a gateway problem. Portable devices such as mobile phones, laptops, and tablets have achieved exceptional success due to the wide wireless connectivity throughout the world. Cellular and WiFi networks have enabled global access to the Internet applications and cloud-based projects. Nowadays, we have seen small portable devices that currently do not have worldwide access to the Internet and consists of low-power batteries, having limited lifetimes and constrained wireless connectivity (for example, BLE: Bluetooth Low Energy). Recently, such wearable objects need to have an application specific to the device (iOS or Android

based) that has to be installed on a mobile smartphone or on a laptop. When we open a new browser or a new web tab, it is not needed to register a totally new application on a wireless router. However, in the case of such an IoT object, it requires to download and install a new application of a smart cell phone, a dongle device or another base station. A new network architecture was proposed that suggests to have an application layer gateway. This will enable the information translation from the connection of low-power to the Internet as a whole. This will provide application layer level connectivity in hardware/software to all the Internet of Things objects [61].

As the Internet of Things techniques are rising in various fields where several other technologies are being utilized and are attempting to be known as the most optimal solution, it is producing many difficulties and necessitates additional hardware/software deployment to communicate with different nodes. In today's world, IoT ecosystem is fragmented due to the reasons such as products from different producers are unable to integrate; numerous protocols for device-to-device communication and not a single one specified as a standard; no programmability which is required for the object's interconnection; variety of OS (operating systems) and firmware platforms; lack of API testing that utilizes similar mechanisms and techniques; non-uniform services of the cloud; objects using same interfaces are unable to push/pull data; lack of protection for products using security software from a third party and no availability of a common controlling and a monitoring tool to manage the IoT objects. In the coming future, there is a possibility that few of them may become outdated and this may cause an effect on the objects utilizing these technologies, making them of no use. This is a critical issue as IoT products, such as smart televisions or fridges, operate for a longer period of times and should be able to provide the service even if the production company and their techniques go obsolete [30]. In order to achieve a seamless and smooth programmability of IoT appliances or sensor devices in a connected world, interoperability is highly required. Horizontal platforms need to be enabled that are functional and communicative across various sensors. There is a requirement for standards. Interoperability is highly complicated which can be made possible by utilizing the application layer that bridges the technology to the layers beneath [52]. Side by side, the security techniques must also be implemented at each layer of the protocol stack that supports both the lower and upper layers. This feature is the focal point of the current thesis and particularly, the security of the application layer.

2.3 Importance of IoT security

Security is the prime focus of discussion in the Internet of Things market. IoT industrialists are continually connecting every object (from small monitors to automobiles) to the Internet and soon there will be more intense revenue progress in IoT business. The data shared by people with IoT devices is often confidential and they are bound to provide all the background information and context perceptions in case of any change in their services. Small startups strive for their share of the industry market and ignore the security aspects of their products. This leads to Internet breakdown, identity theft, privacy and data breaches. Security experts in the military consider IoT botnets as weapons which can also stimulate DDOS (distributed denial of service) attack [55]. In a large network of IoT devices, every single object of this system can be the initial point of a cyber attack which can then spread across the whole network. This is common in publicly deployed devices such as bus stops. Although security techniques are employed, there are chances of physical hacking attacks and as these objects are not the personal property of anyone, it is difficult to protect them. We have witnessed such incidents in the past, for example, the jeep hack where the jeep was hijacked over a Sprint network and the attacker was able to modify the speed of the car [18]. Hospital ventilation system was accessed by a hacker and he remotely controlled the temperature and air/heating systems [51].

Having IoT feature is significantly cost sensitive. For adding smartness to, for example, cars, buildings, and vending machines, the hardware part must be economical. To attain this, only a limited amount of memory and processing power is added that is enough to support the IoT property, with having no facilities left to incorporate the conventional security products, for instance, anti-virus or anti-spyware software. Such design structures have faced cyber attacks in the past, for example, several inspection cameras, digital video recorders, and other IoT supported objects were compromised and a huge DDOS attack was executed [45]. Practitioners are also working to add networking and other IT technology to such devices that originally have not had them. Consequently, users have to use security patches (if they even exist) periodically to keep them updated with new cyber techniques. Those objects that are used at least a decade, have their security patches released on and off. However, those devices (developed years ago) whose vendors stop creating secure patches due to less financial incentives become a serious threat to security. They provide backdoors to wrongdoers to their applications and to the entire system as a whole [28].

The security risks are excessive in autonomous systems of IoT which con-

siders no human interaction involved and only the devices connecting directly with each other. Today there exist several such applications, for example, Tesla introduced the autopilot feature which when turned on makes a vehicle a self-driving car and works without any human touch behind, taking assistance from the roadside implemented sensors [42]. If humans are not keeping the track, there are security risks associated with it. Unmonitored sensors can access sensitive information and if a malware gets installed then it can spread across the entire networked system and may cause blackouts. Even if a system involves human in the loop, unanticipated security breaches occur. For instance, smart watches or fitness tracker wearables can be used by a hacker to detect the hand movements of a person while he is typing a password or a secret code to another device and recreate them. This is also the example of backdoor access to conventional systems [53] [28]. Enormous data is generated using such fitness apps such as they track a person's heartbeat rate, his gestures and overall performance. Such volume of data and its scope can be of much value to the hackers as IoT sensors not only produce them but also they interact with other devices and thus, transmit plenty of information and data. Those companies who have produced a great number of products that are constantly transferring and accepting data must be more responsive towards the measures taken to incorporate security features. The most traumatic situation would be when attackers hack the medically related tools which can cause significant consequences on the health of the patient [17].

It is commonly said to not trust the third parties. The reasons are mostly that such an enterprise may misuse or sell a person's information with the wrong intention or there are also possibilities that a company may be fair to a person but due to poor security measures gets hacked and important information is lost. There are examples when even a government abused individual's confidential information or became the victim of such security hack [17]. One kind of such incident happened recently in March 2018 in Atlanta where the government systems became the victim of this attack. A hacker group known as SamSam attacked city files, locked down the online systems and services (that may be used to pay bills), and blocked Atlanta to do any kind of court-related processing or warrants. SamSam demanded $51,000 ransom in Bitcoin. This affected almost 6 million people and the city's IT department asked the employees to unplug their devices in case of anything suspicious. Atlanta government called out Dell SecureWorks security team and the city was able to recover some services [36]

According to the Gartner report, global IoT security spending will reach 1.5 billion dollars in 2018 which is 28 percent more of that experienced in 2017 (1.2 billion dollars). As IoT is becoming an integral part of our everyday

life, it is estimated that by the year 2021 the money spending on security will reach around 3.1 billion dollars [19]. By 2020, each person will possess five devices and thus, have to divert our attention more towards minimizing the vulnerabilities to information thefts and security attacks. The current security mechanisms need to get updated. About 33% of the enterprises confirm that their generated devices are immensely resilient and are able to protect themselves against any future cyber attack. Around 48% of the organizations have started incorporating secure mechanisms from the very start of the product development stages [17]. Security by design concept is utilized where security is introduced at the start of the developing stage of a product and not after the security breach incident. This has become important as tech firms are increasingly creating new IoT products for consumers and as each one of them has the ability to connect to the Internet, multiple vulnerabilities have to be taken care of. Tech companies have to find ways to secure their systems from unwanted access, should inform their users about their data collection and sharing schemes and should guide them about the usage of a particular IoT device to avoid any kind of risky act [20].

Incidents discussed above might frighten people from utilizing IoT products and this might lead to the termination of the development in IoT industry. Therefore, it is very important to divert our attention towards the security aspects of IoT features. Authentication, authorization, access control, data storing schemes and privacy issues must be carefully designed according to the latest security techniques available which are capable of avoiding any kind of current hacking methods.

To summarize, there exist diverse Internet of Things security concerned challenges. Few of them can be resolved to utilize the current security methodologies. Otherwise, there is a need for thorough discussion and elaboration towards IoT secure technologies with close cooperation with users, producers, software designers, and standardization organizations.

Chapter 3

Authentication and Authorization

Security of the devices basically depends on two goals: a) to protect the access to resources against the unaccredited person, and b) to make sure that the authorized user has the appropriate rights and is able to access the system. Authentication and authorization play a vital role to identify a person and allow him specific access rights to a website or to a particular application.

Authentication: This methodology is adopted by a server where the server analyzes who the person really is and exactly who is interested in accessing its resources. This process is also utilized by a client where it needs to examine the system as it pretends to be. Furthermore, it is used by users to identify themselves with such clients and servers. Authentication methods of server necessitate username and password whereas other techniques could be via cards, fingerprints, face/voice recognition and scans of the retina. Quite often the authentication process by the client is through certificates which are provided by the server to the client. In this process, a credible third party claims that the particular server is a legitimate one and is in accord with the entity it represents to the client. The authentication process, however, does not provide any information regarding what activities a user can perform and what exactly the files on a system he can see. It solely takes the responsibility of identifying an individual or a system [12].

Authorization: This technique is performed by the server where it examines whether the client possesses required rules to get to its specific resources or to some documents. Authentication and authorization processes go hand in hand so that the authorizing server has the idea of the requesting client. For the authorization, there may be different ways to authenticate a client;

username/password can be one of the methodologies in some cases. There are also scenarios where no authorization method is implemented. An individual may directly use the documents or can simply ask for the permissions to access particular resources. An example would be those websites on the Internet that does not ask for any kind of authentication or authorization process and a user can open them anywhere, anytime [12].

3.1 Authentication-Authorization Approaches

With the growth of a large amount of confidential data over the Internet, the need to keep the hackers away accessing such information has risen. Impersonation of identity has become an easy task to achieve. Therefore, special authentication methods to assess a requesting individual's identity are required. Today there are several verification methods and protocols that help to achieve this goal. Such techniques are described as follows.

3.1.1 One-Time Passwords (OTP)

Those enterprises that provide online facilities, also offer various-factor authentication services to strengthen the security of their products. In such scenarios, the system before authenticating a user, verify him using different unions of various components. As a result, in case a component is invalid, faulty or missing, that particular user will be rejected by the verification system and will not be registered. There are several data hiding methodologies introduced that secure the data during their transfer. Those techniques can be adopted by tech organization to escalate security to avoid intrusions or information thefts.

The one-time password is a password or pin code that is created automatically and consists of the numeric or alphanumeric character string. It validates an individual for one login session or transaction. To acquire OTP technique by several projects, they make sure that it requires access to something an individual possesses (for example, a smartphone) and something an individual has the knowledge of (for example, a pin code).

Mobile phone two-factor authentication (MPTFA) is one of the notable multi-factor verification service [56]. In such systems, an individual is asked to enter the data that he has registered in their service. After this, he receives an SMS, email or a message via any sort of application designed for this purpose and is requested to enter a dynamically created one-time valid password that is created by the server out of any combination of digits. It also has limited or short interval expiry time associated with it. When it is

expired, the individual has to enter his credentials again and a new one-time valid password is requested. It is then created by the host machine and is sent to that individual [25].
There can be some advantages and disadvantages of OTP and MPTFA which are described below.

Advantages:

- OTP technique is easier to adopt in the verification systems.
- As compared to the static passwords, OTP is capable of avoiding the replay attacks (an attack during the transmission of the data where the data is sent again and again or is delayed maliciously.) In a situation where an attacker gets hold of the one-time password, he will not be able to misuse it because that OTP will already be utilized by the user himself to log him in the system or to make any kind of transaction and thus, will no longer remain authentic.
- There are cases when people tend to use the same password for several applications. Therefore, when an intruder tries to get the password for one of such applications, this will not make other systems vulnerable as OTP is valid for one session only.
- In MPTFA, there is a limit on the number of wrong entries a user can enter. When that limit is achieved, the user is no longer able to access the application. This reduces the compromised data possibilities caused by hit-and-trail attacks [25].
- As the passwords are short-lived and expire quickly, attackers are unable to utilize them in future to get access to some application.

Disadvantages:

- There can be social engineering attacks on such passwords where a phishing (an illegal process to get access to the confidential information by impersonating as an honest identity) entity deceives the users and tricks them to impart some of their previous one-time passwords. The untrustworthy entity can then analyze the pattern and will be able to predict the type of OTP be generated in the future.
- One-time passwords can be in a way more secure than static ones. However, there can be man-in-the-middle attack and they are unprotected against it.

- The battery life of mobile phone and cell phone signals can affect the reception of the messages containing the pin code or OTP [25].
- If the SIM registered for receiving the OTP or the phone itself possessing that SIM is lost then the user will not be able to login to his specified applications due to incomplete login process [25].
- For MPTFA method, there are privacy concerns as the user has to provide his phone number to such applications to send them OTP or pin codes. There is always a risk that such applications can publicize his number or share to other services [25].

3.1.2 Context-based Authentication (CBA)

This kind of authentication mechanism utilizes contextual data to ensure an individual's authenticity. In addition to other secure authentication techniques, this mechanism is suggested to be utilized along. Enterprises looking for efficient and strong authentication mechanisms should take into account their customers, business proceedings, threats, and risks, and then choose an optimal security method that is flexible enough to modify according to the requirements. For instance, a company can utilize context-based authentication method to enable an authentication system that utilizes public key infrastructure (PKI) technique such as digital signatures, FDE (full-disk encryption) and a network login or plans to incorporate them in coming future [9].

Organizations, utilizing CBA in their authentication systems, are continuously increasing to support remote access to the workforces. CBA technique provides stronger control of their worker's machines where a thorough secure procedure is designed that develops secure connectivity between devices and the sensitive applications they utilize. CBA considers various factors to develop trust and enhance security. Some of them are described as follows.

- Every machine that a user utilizes, for example, mobile phone, laptop, tablet, and computer, is characterized, have a distinctive identity and is linked with the particular user. Thus, device-specific authentication is achieved [24].
- In order to implement two-factor secure method without any friction, both the user's login secrets and the device's identification number that is assigned to him are used by the context based authentication system. Thus, supporting behind-the-scenes and transparent authentication technique [4].

- Enterprises have defined trust tags for their devices and customers utilizing their products. The purpose of trust tags is to label them as they can be trusted always for any further interactions. In this way, an individual is labeled as a valid user and does not have to go through all the steps of authentication mechanism. Thus, he or the device will be granted access instantly [4].

- Context-based technique also tracks the user activities and locations and develops the legitimate data about it. Such information includes login activities, the speed and the number of times of user login efforts, languages used, number of devices and relevant configurations, IP addresses, geographical locations and uncovers unexpected places or points based on the recent data [4] [24].

- It highlights anomalies or viruses that have infected the registered device used for login or are someway involved in some fraudulent activity [24].

3.1.3 Certificate-based Authentication

Digital certificates are used for authentication of the user, server, organization or any entity and aim to achieve a secure connectivity. Certificates consist of an important information, i.e., a public key to an identity that is in accord with the private key. Similar to the personal identity card, passport or a driving license, a certificate provides acknowledgeable proof of a user's or any entity's ID. They can be kept on the client locally or on any kind of portable device, e.g., a smart card.

There are trusted entities that perform the validation of the identity and sign the certificates digitally by using its private key. Such a trusted body is known as the certificate authority (CA). Certificates operate by declaring the bearer's identity by attaining a certificate signed by CA. CA can be an organization introduced by the government; it can be a commercial company or independent third party that provides its own certificate-issuing services, and it can be internal to a specific enterprise. The techniques required to verify an entity varies and relies on the rules defined by certificate authorities. Usually, CAs use already formulated and published validation process, prior to certificate issuing, to assure that a user/service asking for a certificate is actually who it states to be [2].

A certificate provided by the CA associates the entity's name with its public key, for instance, the name of a server or an employee. Certificate reduces the chances of impersonation attacks by the usage of forged public

keys. Certificates can be transmitted freely anywhere as it has to be validated for a specific user only with its private key combination. The certified public key will only work with its corresponding private key that is linked to its owner defined by the certificate [6] [2].

Additionally, certificates consist of subject's name, CA's name, certificate validation period, serial number, revocation list, and other details. The most significant part is CA's signature that acts as "letter of introduction" for those individuals who are unaware of the entity certified by certificate but have put trust in the CA. Furthermore, when the expiry date of the certificate has reached, a new one must be requested. CA also has the authority to revoke the issued certificate [2].

A software application of the client or the server that supports authentication via certificates keeps a set of all trusted certificates from CA. These certificates can further determine other certificates which the application can verify, i.e., what other certificate issuers an application can trust. The simple case would be an application authenticating only those certificates that are issued by only one CA for which it possesses a certificate. There are many cases where a certificate is part of a CA certificates chain. In this chain hierarchy, every certificate is issued by the certificate authority above it. The certificate-based authentication facility is part of the PKI (public key infrastructure) and the standards are based on X.509 specifications for the frequently used certificates [54] [2].

As they are stored on the client machine, certificates are considered to be verifying the physical device and not the user himself. Today with the advancement in technology, smart cards have been created that facilitates the individuals to carry their electronic certificates and private keys along. This has parted the device and the certificate. Analogous to the one-time password method that utilizes a pin, using smart cards or certificates only, fails to achieve two-factor authentication. This has been solved by using a pin code to unlock the card and provides access to the specific user's credentials [6].

There can be pros and cons of using this method. Some of them are described below [1].

Pros:

- Level of security is equal to or higher than the public-key authentication method
- Using one location, an individual's access to various servers can be administered, an add-on to security in some cases.

- Credentials used for authentication purposes can be revoked centrally.
- Users rely on CAs that help them by analyzing how trustworthy the remote host is.
- It is a highly scalable process where there is no requirement for a separate entity rather only one CA or a limited number of them.
- Identity is verified only by the secret private keys.
- Certificates are utilized not only for login but also for the security of e-mails and file servers access.

Cons:

- To get an SSL certificate can be costly.
- Every time a certificate expires, an updated one is required.
- There are also risks that CA are compromised by attackers.
- If there is an involvement of encryption/decryption of messages then it becomes a slow process.

3.1.4 Biometric credential

Biometric does not focus on what a person possesses and what he has the knowledge about instead it relies on what he is [6]. It is more secure than using a smart card as it involves the usage of biological statistics which states that the possibility of two persons having same biological features (e.g. fingerprints) is very small or close to none. Therefore, such biological characteristics can be utilized to authenticate a person positively and is hard for attacher to falsify them.

Not only the fingerprints, iris, retinal, facial and voice patterns are unique from person to person but can also be utilized for identification. Implementation of such methodology requires to have costly tools and equipment. Unlike smart cards, a person does not have to carry it everywhere, his biological traits are never left at home [54]. However, if biometric credentials such as fingerprints are hacked by an attacker then it is not possible to modify the biometric authentication.

3.1.5 Single Sign-On (SSO)

Single sign-on is a session that is centralized and provides authentication services for users where only one set of login data is utilized to get access to various applications. In this way, the user is authenticated for all the applications to which he is authorized and further login prompts are eliminated when he switches to other apps in the same session. Considering the back end, single sign-on helps to track the user activities and logging accounts of the users. SSO has solved the problem of users who do not remember the passwords for various accounts where forgotten, unused accounts are at security risks; and slow service access and inconvenience. Single sign-on is usually implemented using lightweight directory protocol (LDAP) which is described later in this chapter [8].

According to Network Applications Consortium, in large organizations, people spend the average of 44 hours in a year on login activities for 4 apps. The number of times people called for resetting their passwords was also measured and it was 70% of the calls [29].

This SSO technique is beneficial for users and administrators as well. They need to possess only one set of credentials that can be used at regular intervals. This indirectly improves the productivity of the user. The authentication framework only needs to monitor the modifications of a single entry for each person in the database designated for credentials. One of the main advantages is the centralized system of authentication data. Moreover, the same methods and tools are utilized for access or any kind of change. However, there is also a risk involved, i.e., if an attacker gets access to the database and bypass the security system implemented, he can get hold of all the information instantly.

The unvarying authentication scheme is enforced by centralization throughout the organization which in turn is easier to protect as compared to the distributed scenario. In the absence of SSO, users in order to remember their passwords, write them down on their sticky notes or under their keyboards which poses security threats.

An appropriate solution for SSO would be application or platform neutral. This technique would cover/hide the implementation framework for authentication on multiple OS platforms from the SSO users and can assist to outsource the authentication method on application layer to authentication authority of centralized single sign-on. It is stated that SSO is a key to the kingdom which means that if someone gets access to single sign-on credentials then he can obtain all the protected resources. In order to minimize this threat, not knowledge based credentials should be used (where knowledge-based credentials can be passwords). It is recommended to utilize

possession based (e.g, smart cards, tokens) or biometric credentials. Using multi-factor authentication techniques with SSO can further minimize the security threats[29]. Some of the SSO protocols that are utilized today are described as follows:

3.1.5.1 Security Assertion Markup Language (SAML)

Security Assertion Markup Language is a standard that assists secure data exchange. Produced by Organization for the Advancement of Structured Information Standards (OASIS), it is a framework based on XML that is used for the authentication and authorization purposes. It is a single sign-on standard format. Electronically signed XML files are exchanged for authentication purposes. SAML is a complicated SSO implementation that ensures seamless and perfect user authentication between organizations and businesses.

There are three associated roles in SAML, i.e., end-user/principal, the service provider (SP) and identity provider (IdP). IdP assists by providing online resources that support authentication to the principals in the networked environment. They are sometimes known as identity assertion providers or identity service providers. SPs issue particular resources to the principals for SSO. SP first establish trust with IdP for the user authentication. IdP, on the other hand, creates an authentication assertion which represents that the specific user has been identified and authenticated [50].
There are two kinds of SAML SSO flow which are described as follows:

- **SP Initiated SSO:** In this type of flow, principal or end user starts performing the login process at the service provider. Service provider redirects the principal to identity provider using a SAML request. This SAML request will possess the required data needed by IdP to identify the user and send the response to the service provider with correct SAML assertion message, i.e., a SAML response. The SP initiated flow is shown in figure 3.1.

- **IdP Initiated SSO:** In this type of flow, principal or end user starts performing the login process at the identity provider. Identity provider has to be configured with the service provider's metadata (for instance, Issuer, Assertion consumer URL and audiences). Identity provider will dispatch response (SAML Assertion) to the service provider. The service provider will validate it using the configured information. The IdP initiated flow is shown in figure 3.2.

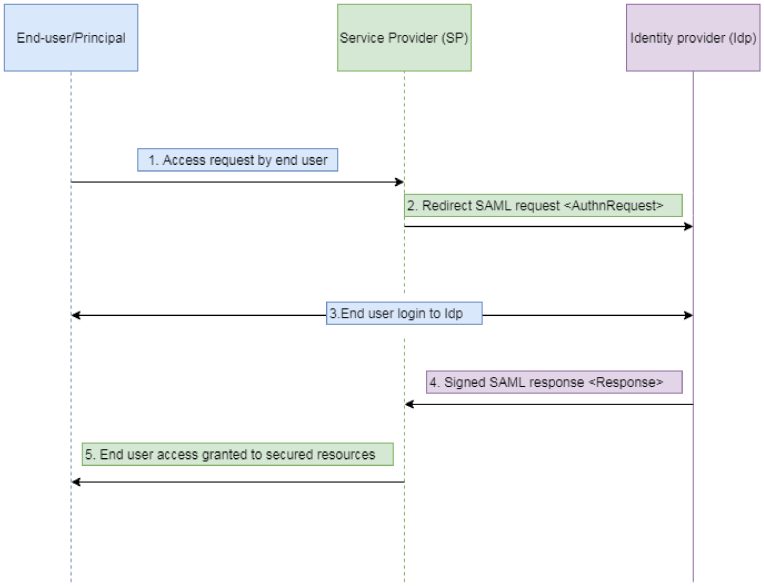

Figure 3.1: SP initiated flow. Adapted from [47]

SAML constitutes request-response XML exchange messages between users, SPs, and IdPs. Five main specifications include profiles, protocols, metadata, assertions and bindings [47].

- **Profiles:** It defines functional combinations of bindings, protocols, and assertions supporting a specific use case.
- **Protocols:** It defines packaging of SAML elements, i.e., the flow of messages for assertions requesting.
- **Metadata:** It defines the trust connections.
- **Assertions:** They possess a packet regarding security information or can be a decision information packet. They are the XML statements from identity provider about the end user.
- **Bindings:** It is the SAML message mapping to the standard messaging pattern or to the communication protocols such as SOAP and HTTP.

Some of the benefits that SAML offers are: a) The standard SAML format enables smooth compatibility between various parties and does not depend on the implementation architecture. It solves the problems of vendors and platform related infrastructure and its execution, b) By logging in only one time, individuals can access various SPs and does not have to provide any

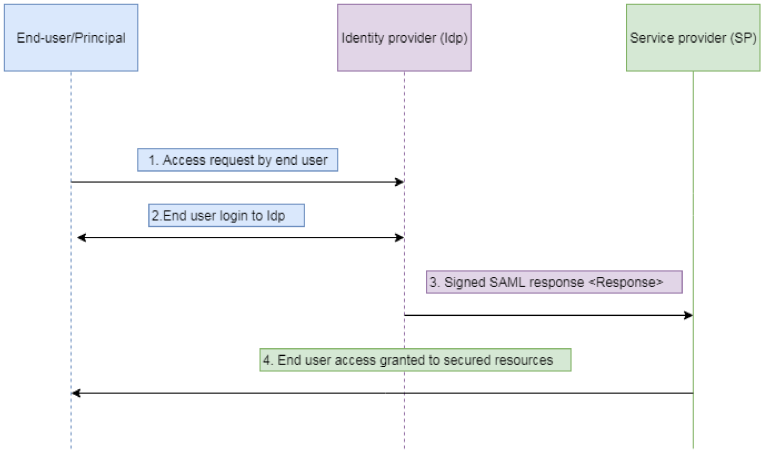

Figure 3.2: IdP initiated flow. Adapted from [47]

further authentication. This makes this procedure a faster one and a better user experience at every SP. Additionally, it prevents the problems of password resets or recovery, c) For every software product developed by an organization, security is mandatory. With SAML only one point of authentication is provided that is performed at a protected IdP. The identity of the individual is transmitted by SAML to SPs. Thus, ensuring that such authentication communication is within the firewall horizon, d) SAML supports loose coupling among the directories. It does not need to monitor, synchronize or maintain any kind of data regarding the principals between any files or directories, and e) No user account data needs to be maintained over various services, therefore only IdPs has to pay for this whereas SPs does not have to bear this burden [50].

3.1.5.2 OAuth 2.0

OAuth2 is an authorization model that allows third-party apps to get restricted access to user accounts on an HTTP service such as Google, Facebook, and Github. An individual's authentication is delegated to a service on which that individual has his account and enables authorization for other applications to access his account. Authorization mechanism flows are provided by OAuth2 for mobile, desktop and web applications.

Let us consider a conventional server-client structure. A client asks for a protected resource (that has restricted access) on the server by performing authentication with the server utilizing the credentials provided by the owner of the resource. In case of third-party apps that require access to the secured

resources, the owner shares the credentials with such apps that poses several threats and issues: a) Third party applications saves the resource owner credentials for using them in future. This is conventionally a clear text password, b) servers have to develop a password-based authentication system, although the security threats against password-based authentication are quite obvious, c) a broader access to protected resources provided to third-party applications as there are no limitations of time or any kind of access restrictions to any resource type, d) in order to revoke access of a specific third-party application, resource owners have to revoke access from all other third-party applications. It is done by modifying the password of the third party, and e) if in a case any third party secure mechanisms are hacked, this will result in the compromise of the password and all related information that is secured by using that password [37].

The solution to such problems is provided by OAuth that presented authorization layer and parted client role from the owner of the resource. Using OAuth, it introduced the different set of credentials (other than the ones used by resource owners) for third-party applications to access the secured resources maintained by owners and hosted by resources servers. Therefore, instead of using the same information as that of resource owner, a client is provided with access tokens. Access tokens are the strings that specify the scope, validity period and other related properties. These are issued by an authorization server only when approved by the resource owner. The client uses access tokens to get to the access-restricted resource handled by the resource server. For instance, a resource owner (end user) allows a client (let us say printing service) to access his secured photographs that are saved on a resource server (let it be some kind of a service for photo sharing) without providing the client with username and password. The end user authenticates himself with another server (authorization server) that is trusted by the resource server and issues access token, i.e., client delegation specific credentials [37].

There are four roles that are specified by OAuth [21]:

- **Resource owner:** An entity that allows access to the restricted resource. When this entity is a person, it is termed as an end user.
- **Resource server:** It is the server that hosts secured resources, has the ability to accept and send responses to requests by utilizing the access tokens.
- **Client:** It is an application that makes requests to the secured resources on behalf of resource owners and with its authorization specifications. A client can be any sort of application on a desktop, on a

server or can be on any other device.

- **Authorization server:** This is the server that creates access tokens and issues them to the clients only when the resource owner has authenticated himself and obtained the authorization. Authorization server can be a separate server or it can be the same one as the resource server. Only one authorization server can be used to issue multiple access tokens for several resource servers.

Figure 3.3 represents the interaction between the above-mentioned roles. Each step is described as follows [21]:

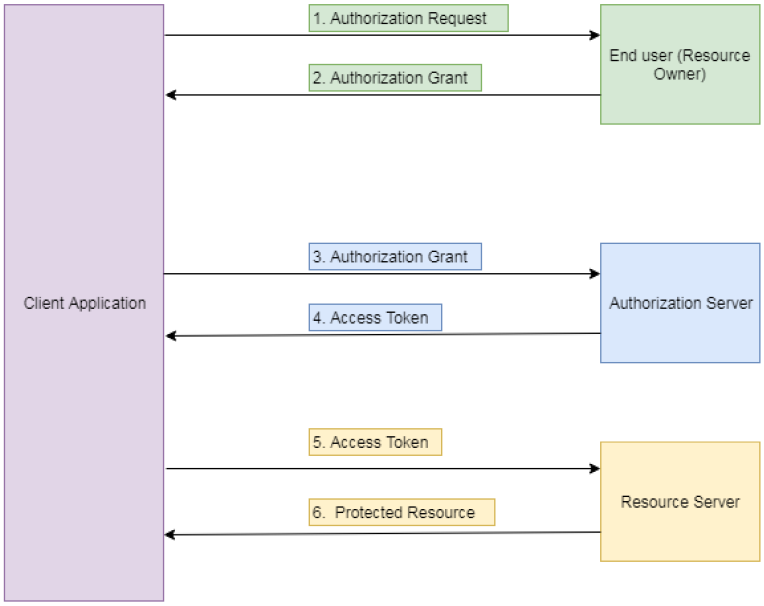

Figure 3.3: OAuth Abstract Protocol flow. Adapted from [21]

1. First of all, the client application asks resource owner for the authorization. This request can be a direct one as given in figure 3.3 or it can be made indirectly through the authorization server which acts as an intermediary.

2. The client gets authorization grant from the resource owner. It is a credential that shows the authorization of the resource owner and has four types. The types depend on the kind of method utilized by the client application to request and also on the support provided by the server.

3. Access token is requested by the client application by performing authentication with authorization server and providing the authorization grant.

4. If the client is authenticated by authorization server and if the authorization grant is valid, an access token is being issued by authorization server to the client application. Authorization is then complete.

5. Now access token is utilized by the client application to authenticate himself with resource server and asks for the secured resources.

6. Resource server then checks the validity of the access token. If it is valid then the client is granted with the resource he has requested for.

There can be pros and cons of using OAuth methodology. Some of them are discussed as follows [35]:
Pros:

- OAuth has made it easier for people to sign up to the websites without going into the hurdles of creating their own accounts from scratch. For instance, a person wants to edit a file, whose link is shared to him by his friend. He cannot edit the file until he has an account on that website. Here the OAuth functionality comes in. The website may support "Sign up with XYZ account". So considering that person already has XYZ account, can log in using its credentials, access the file and edit it.

- Consequently, OAuth is time-saving. If multiple sites support OAuth, then a person can access all of them using only one account.

- With OAuth, a user has the privacy of his confidential data. For instance, online shopping can be done with OAuth login to an online bank account, without letting the site know about the private bank details.

- It is a secure mechanism that utilizes secure socket layer (SSL) for protected communication.

- With OAuth, users have the ability to restrict the time frame to access their information. The expiry time of the authorization tokens can be chosen by the user.

- This methodology reduces the expenses that are spent due to licensing fees for businesses. It is an open standard and can utilize multiple online sites without any kind of association fee.

- Websites can retain their users using OAuth as the user does not have to go through all the efforts of creating an account and joining the community.

Cons:

- There can be phishing attacks where a reliable looking website may ask an individual for his credentials and will have his data phished.
- An attacker get access to all the applications that a user has logged in by using an application (e.g., Facebook) by hacking that user's Facebook account which is used to authenticate the user.
- There are chances that obtained data can be misused. Recently, there is an example of Facebook that has been guilty of such a procedure.

3.1.5.3 OpenID

OpenID protocol is a protocol that utilizes already existing account to log in to a service without any need to make new credentials. It is an HTTP based protocol and utilizes identity providers for user validation. It supports SP-initiated SSO flow. The word "open" means that any entity can be an identity provider, a user has the choice to choose an identity provider and is able to work on any sort of web browser without the need of patented software [13].

OpenID is the authentication protocol. End-user first chooses OpenID identity provider (e.g., Google) and gets an OpenID account. He then uses this account to register at any relying party (a website) that allows authentication via OpenID. The end-user then authenticates himself to the relying party with the assistance of an identity provider. It provides a communication model between the identity provider and the relying party [46].

Let us take an example of a person crossing a border. If a person (end user) is from Canada (identity provider) and wants to enter to USA (relying party), he needs to show his identity at passport control. As the government of the USA trusts the government of Canada to correctly provide identification for their people, the USA government will consider that person's passport a reliable one and will allow the person to cross. This scenario works as the end user has provided his identification from an entity that is trusted by the relying party [46].

OpenID is simple, easier to use and shifts trust from various parties to one. However, there are some threats included: a) phishing attack where compromised relying party directs user with a foul identity provider that

asks him to enter his credentials. In this way, relying party gets access to user account with identity provider and can use his account to sign in to other sites, b) a hacked OpenID account is more dangerous regarding the privacy breach than the one on a single website, c) this procedure lacks privacy as identity provider has the log of OpenID logins and can easily do cross-site tracking, and d) in case the communication is not secured, authentication can be hacked [7].

Today OpenID connect has been introduced (created in early 2014) and over a billion of OpenID connect accounts can be seen on the Internet. Enterprises such as Yahoo, Google, PayPal, and WordPress involves OpenID connect for user authentication. OAuth2 is the basis for OpenID connect where OpenID is used for authentication on the top of OAuth2 which is used for authorization. Thus, providing a complete security package [46].

3.1.6 Active Directory (AD)

Considering the computer network, a directory is utilized for two purposes: 1) To save information and organize objects data on the network, and 2) To discover the location and fetch the data regarding network objects from the storage of the information. Standards covering the entire industry to allow interoperability functionality between multiple computer OS were created [3].

Active directory is one such directory which was initiated by Microsoft Windows 2000 server. It was considered the fundamental part of server OS. It was upgraded in windows 2003 server. There are two main functionalities that Active Directory supports: a) AD saves information regarding computers, the users, their groups and other related data of the network, and b) AD allows clients to access data from the storage required for the authentication and authorization purposes [16].

The AD consists of a hierarchical organization of forests, sites, domains, OUs (organizational units), users, group of the users and accounts on the computers, thus storing the data regarding objects in the network. Not only the user, group and computer accounts, network objects also involve applications, printers, servers, network shared folders and security rules that defines allow/deny access of the users/computers or any kind of entity present in the network structure. Similar to other directory services, the AD also shares its stored data with the authorized entities such as admins, registered applications and allowed users.

One of the methods utilized by the Active directory to make secure the identity of the network entities is by handling the authentication during login and managing authorization rights to the resources of the network. The AD

is structured in a way that it has the ability to verify the identity of the user and allow/deny the access to the protected resources for computer users that is runnable on windows OS and also other operating systems [3].

The services provided by the Active directory to the entire network are briefly described as follows: a) Active directory runs on domain controllers for windows servers (2000 and 2003). A domain controller is the one that manages directory information (such as security rules and authentication info) domain-wide and handle interactions of the user domain, b) Active directory global catalog has the copy of all objects in an AD but an only small subset of attributes. It is used for searching functionality and locating copied objects, c) LDAP saves user accounts in a central database and supports authentication/authorization functionalities. AD supports LDAP v2 & v3 and for windows based network behaves as LDAP directory service, d) Kerberos (version 5 supported by the AD) is a highly secure method for authentication login and authenticated network service. It asks its users to login only one time and are not disturbed again for adding credentials, e) GPO (Group policy objects) contains group policy configurations that allow admins to apply policies, thus controlling all objects, f) AD can be integrated with DNS (Domain Name System) to locate domain controllers, with business applications such as SQL server and IIS (Internet Information Services and with other services such as remote access and certificate services, and g) AD enables clients from different OS and that use different directory services for interoperability [16] [3].

3.1.6.1 Lightweight Directory Access Protocol (LDAP)

LDAP presents a secure method for authenticating users and assess the authorized policies for that individual to access resources. A client/server protocol drafted for sending requests to directory services and changing required information in the directory storage. Platform independent LDAP is TCP/IP based IETF standard. It defines a communication protocol between LDAP servers and clients and shifts the details about the server implementation to the ones who create LDAP objects. It works in a way that client requests to connect to the LDAP server, when it is connected sends a query, gets the answer for his query and lastly, it ends the connection with the server.

LDAP specifies the query method to already present directories, ways to refer to the objects in the directory, description of the object's properties and the security characteristics that define access control for the objects in the directory. LDAP is the updated version of X.500 DAP (Directory Access Protocol) and saves data in a hierarchical manner where it forms a tree-like structure of different entities. Every entity has attributes and every attribute

has some specific values. Domain Name System (DNS) can be used by LDAP for the entities at the top level and can utilize entries depicting objects at lower levels such as printers, users and groups [3] [5].

LDAP has the power to extend to add new functionalities and maintaining compatibility at the backend smoothly. Many prime directory services have adopted LDAP standard. Two examples of LDAP directory are windows based AD and UNIX based iPlanet [3].

LDAP is comparable to a relational database, however, it does not possess some of its features. It deals with the reading of attribute-based descriptive information and is not usually modified. The ldapseacrh tool is utilized by UNIX systems for reading purposes. This tool can also be compiled for windows system where it is used for searching and displaying the objects of the AD [3].

There exists four fundamental component models of LDAP. Each one of them is described as follows [3]:

- **Information Model:** In order to represent an attribute of the entry, there needs some information regarding the data type and the data structure. The properties and features linked with the entry are specified in the object classes of the entry. The statements explaining the attributes and object classes are defined in the schema. LDAP demands an ability to use values of the attributes to search for the particular entries.

- **Naming Model:** The way reference to the entry should be made is defined by the naming model. The structure of LDAP consists of a hierarchical tree that is known as DIT (Directory Information Tree). DIT consists of nodes that represents an entry. Each node has the ability to save data and serve as a container for further entries. To provide a reference to entries, two methods are specified for this purpose. Either a DN (distinguished name) of the entry can be used or RDN (a relative distinguished name) can be taken into account. DN is considered distinctive globally whereas relative distinguished name is unique inside a directory.

- **Functional Model:** Functional model is considered as the LDAP protocol itself. It defines the ways in which the client of the directory can interact with the directory. It supports three operations: a) searching entries within the directory, b) adding, removing, modifying or upgrading directory entries, and c) authentication procedure to the directory. It is also known as bind operation.

- **Security Model:** The security model of LDAP defines authentication methods for the directory and specifies authorization procedures of the user access control towards directory. LDAP security model consists of two fundamental components
  - **Authentication Component:** Authentication in LDAP is performed by binding the client to the server of the LDAP. If the client's input credentials are correct then the client is accepted otherwise rejected. Successful bind leads to the successful authentication of the client and unsuccessful bind specifies unauthenticated client.
  - **Access Control Component** When the authentication of the entity is completed, the entity is allowed to access the LDAP directory only according to the specified ACL (access control list). Every object in the network comprises of an access control list that defines client/user accounts, various groups, and devices such as computers that are accessible to him or are denied using from him.

The details how we have utilized LDAP and OAuth2 are described in the further chapters.

Chapter 4

Open Group Messaging Protocols

4.1 Introduction

The Open Group is a global board that aims at achieving business goals of the organizations or individuals via IT standards. It manages over 600 enterprise memberships that involve users, academics, system and solution suppliers, consultants, integrators and tool vendors [10]. The IoT Work Group formed under The Open Group provided a comprehensible vision for the standardization of the protocols: The way web Internet utilizes HTTP protocol for transferring HTML-based data that is rendered in web browsers for user consumption, in the same manner, Internet of Things will utilize O-DF based information structure that is mainly developed for automated consumption by IT systems [11]. To put it in another way, O-MI/O-DF standards will offer connectivity functionalities between various things in the surroundings to the Internet and will allow quick integration of such links with the organization's systems and network, making it looks like it is produced by only one vendor. There exist techniques that manufacturers use to get the data from their systems at regular intervals to have the knowledge about their services and internal processes all along the life of a product. This is done for maintenance purposes and to make the systems more secure and reliable at the expense of low cost. It is also performed for keeping the note of a product's production phases, for instance, the condition of the machine or energy consumption measurement. The gathered information can be utilized for analysis and assists in managing further development costs, enhancing the production processes and quality of the end product. This chapter will discuss Open Messaging Interface (O-MI) and Open Data Format (O-DF)

standards in detail along with the examples from real-world.

4.2 Open Data Format (O-DF)

The Open Data Format specifies a standard representation of the data exchange in IoT devices. It is defined using an extensible XML Schema. Its formation is intentionally similar to the data structure in OOP (Object Oriented Programming). Its hierarchical structure consists of "objects" and sub-elements with "objects" being at the topmost level. Under the "objects", the nodes are "object" sub-elements and they can be of any number [14].

O-DF has solved the issue of data publishing from several nodes or sources. There can be various data resources such as server systems, machines, and devices which have to publish their information, support user access, provide filtered information to the user (identity of the entity requested, context and other parameters regarding the request) and incorporate security aspects for authentication and data confidentiality. O-DF has provided an uncomplicated way to publicize the information and it is through Uniform Resource Locator (URL) address. Furthermore, O-DF has enabled multiple systems to develop, control and maintain the data exchange about their devices or "things" in a comprehensible, globalized and standardized manner [14].

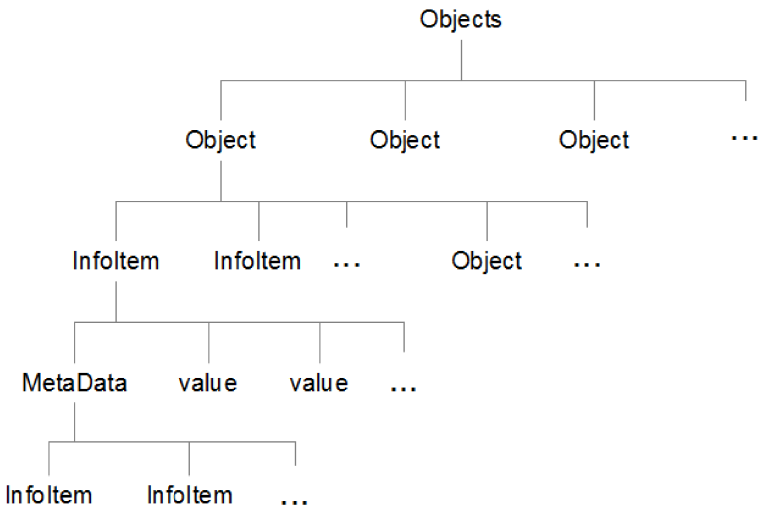

Figure 4.1: Open Data Format hierarchical structure

Figure 4.1 shows the generic version of the object tree for O-DF standard.

The element at the root is “objects” with several sub-elements as “object”. Every object is considered to possess an ID as its identifying element and may have “description” which is optional. Additionally, every object also consists of other features that are described in “Infoitem” sub-objects. Under Infoitems, there are “MetaData” sub-elements and the “values”. “MetaData” posses a similar formation as that of “Infoitem”, however, it has a different functionality. It defines the “Infoitem” sub-element and can be used in situations when there is unspecified “Infoitem” and systems needs to get information about it. The “values” contains the data values from the reference of “Infoitems” (for instance, sensor values of the humidity). There is also the possibility that “Object” contains further “object” sub-elements of the similar kind as itself. This contributes to the formation of a tree structure that possesses the random number of levels with various objects from the already specified set of kinds [14].

```
<omiEnvelope ttl="10" version="1.0"
xmlns="http://www.opengroup.org/xsd/omi/1.0/">
  <response>
    <result msgformat="odf">
      <return returnCode="200">
      </return>
      <msg>
        <Objects xmlns="http://www.opengroup.org/xsd/odf/1.0/">
          <Object>
            <id>SmartHouse</id>
            <Object>
              <id>BedRoom</id>
              <InfoItem name="Moisture">
                <value unixTime="1533475254" type="xs:double"
                dateTime="2018-08-05T16:20:54.665+03:00">1.0</value>
              </InfoItem>
              <InfoItem name="Temperature">
                <value unixTime="1533475254" type="xs:double"
                dateTime="2018-08-05T16:20:54.665+03:00">0.0</value>
              </InfoItem>
            </Object>
          </Object>
        </Objects>
      </msg>
    </result>
  </response>
</omiEnvelope>
```

Figure 4.2: Example XML structure of Infoitems and their values

Figure 4.2 represents an example of Infoitems and values in an XML format. As the structure of the O-DF is in a hierarchical manner, there exist ways to discover and request specific elements with the help of RESTful,

URL mapping scheme. This assists the user to make requests using particular URLs that involves the identity of the element or the name of the feature. For example, the XML response shown in figure 4.2 can be requested using the UNIX "wget" utility: *wget <URL>/REST/Objects*. If a user wants to request the information from SmartHouse only then: *wget <URL>/REST/Objects/SmartHouse* or if he wants to set or read the temperature of the bedroom then: *wget <URL>/REST/Objects/SmartHouse /BedRoom/Temperature*. The response must have all necessary attributes or sub-elements. It can also have additional sub-elements and attributes defined.

To summarize, O-DF was created with the aim of expressing the data regarding multiple different machines using a common method and that is not dependent on the context or the application. The transfer of the data in the network is not related to the O-DF technique. The data encoded using this standard can be communicated using several existing low-level protocols. The other option is to copy them in a USB storing device and manually transfer them to the other system. The fundamental objective of O-DF is to use it with the open messaging interface i.e., O-MI for requesting and as the structure for the response [14]. The Open Messaging Interface (O-MI) is explained as follows.

4.3 Open Messaging Interface (O-MI)

O-MI has the similar aim as that of HTTP for the web Internet. With the usage of Open Messaging Interface, multiple sensors and machines can communicate with one another. For the lifecycle concept in IoT, O-MI must assist in the interaction between various objects and other IT distributed systems that utilize/provide data instantaneously that is related to the object's lifecycle. As the concept of "things" introduced by IoT specifies it as anything in the surroundings, O-MI is designed to incorporate all of them. Therefore, O-MI has been defined in a most possible generic way. In spite of the focus on the lifecycle of the product, it should be possible to apply this IoT standard, i.e., O-MI to the lifecycles of everything (for instance, e-documents, services, and humans). The transportation of the payloads can be done using O-MI in almost any kind of format that will exchange the data among various O-MI nodes in a network. The data can be encoded using XML (which is considered the common way for text-based payload structure) and other formats such as CSV and JSON are also considered [15].

A fundamental property of O-MI is that it does not consider the predefined roles of its nodes and the communication is based on "peer-to-peer" in-

teraction model where each node can act both as a "client" and as a "server" with the other O-MI nodes or IT systems. Thus, interacting directly with one another and also with the servers at the back-end. The examples of information exchanged are new info availability notification, sensor readings, modifications in existing information, lifecycle or alarm events and queries for the past data [15].

Below are the eight fundamental functional requirements that are needed to have a scalable solution to all the IoT wide systems. The already present methodologies in the market are specified for a certain field of view and consequently, a considerable amount of amendments or extensions would be required to modify them for the other use cases. Therefore, a generalized solution was proposed that is scalable and tries to incorporate all IoT devices. The O-MI important features and requirements are explained below [15].

1. O-MI should have the capability to transfer the messages by utilizing the defined "low-level" protocols. Not only the network protocols such as SMTP, HTTP, and SOAP, O-MI can also transmit the data by copy/pasting them to and from USB sticks or any other kind of storage media. Sending the O-MI messages via texts on a mobile phone can also be done.

2. There are three defined basic functions, i.e., read, write and cancel. Read is used for querying the data and immediate retrieval of it. It is also used for deferred retrieval, i.e., for subscriptions. Write operation is used for writing the information from the IoT devices to the O-MI nodes. Cancel functionality is used to cancel out the subscriptions before their expiry time. The "Delete" function is under development phase which when applied will be able to delete O-DF objects and Infoitems.

    - **Immediate read:** When O-MI nodes request about a specific value of an Infoitem at a specific time or at the current time, they should be responded to immediately.
    - **Deferred read:** It is also called subscription where the read request is stated with interval rates and a callback URL that can be optional. When a user makes a deferred/subscription request, he will then receive the required information according to the intervals stated. The information will be sent to the callback URL as specified. In a case when no callback URL is defined, the information is retrieved i.e., "polled". This is done by issuing a read request in which the ID of the subscription is specified (that a user

gets as a response when he requests for subscription). Polling is beneficial in cases when the system uses NAT or firewalls that prohibit responses to link with callback URLs.

3. O-MI nodes need to send as quickly as possible any sort of data (whether current or historical) that becomes available to other O-MI nodes when they request them. This has to be done at any time.

4. It should enable different formats for payloads that are used in both requests and responses. Any kind of text-based formats such as proprietary and standardized can be utilized to transmit O-MI message information and such format can be embedded into XML as a payload.

5. There must be Time-to-Live (TTL) defined. TTL specifies the time interval in which a particular message is valid that can be transmitted, forwarded or is being replied back. If a message TTL has reached its expiry time, it needs to be removed and an error message should be generated that is returned back to the message initiator.

6. A synchronous interaction with O-MI nodes should be enabled. Any kind of response message can have a request message specified in it without having to send a completely new request message in an additional query. This creates possibilities to initiate a connection with the nodes that are situated behind NAT or firewalls.

7. Data services, information resources, and metadata discovery and publications can be done utilizing the "write" operation. Simple RESTFUL URL based queries such as HTTP GET or search engines can be used (along with "read" requests) to discover such information. There can be different formats from various resources and services. The format specification is the part of format standards such as O-DF and not the O-MI.

8. There can be a set of O-MI target nodes that are specified in the requests. The nodes that receive the messages have to forward the request to the defined target O-MI nodes or in case of any failure, must issue an error message to the node that has requested.

To summarize, Open Messaging Interface is a method for sending messages among the nodes in a general way possible. This mechanism can be utilized for various types of data such as files, document repositories, and even physical objects. It not only specifies read/write operations but also offers a subscription tool that enables information delivery when requested

when some modification is seen or after a particular defined time interval. Subscription is considered as the fundamental notion of O-MI.

Chapter 5

Reference Implementation Modules

5.1 Introduction

A proper documentation has become a vital part of the newly developed technology and standards. In order to have a complete understanding of the standards, it is beneficial to link them with a sandbox application and reference implementation. With reference implementation, every aspect of the standard is tried to be covered which includes request/response examples. This makes it easier for readers/developers to test protocol specifications that are written on the documentation and understand it thoroughly by actually executing them. The reference implementation for the O-MI consists of the following modules, developed by Aalto University, School of Science.

First considering the **O-MI node server**. This server holds a database that maintains all the information regarding O-DF based objects, object, Infoitems, and their values. This server is enabled to perform all the functionalities, read, write, cancel and subscription. The underlying transport protocols that are currently utilized are HTTP and web sockets. Any operation of O-MI is transported using HTTP POST request. In most of the scenarios, reference implementation acts as a simple REST node. However, not in the case of subscription operation where responses are based on intervals or on events to the list of users who opted for subscription mechanism. There is also an optional feature of specifying callback URL address. It differs from that URL which is used by the user to send the request. Callback URL is defined so that messages are received at this address when the subscription session is active. If no callback URL is specified then the information can be polled (as described earlier in 4.3) utilizing a new read request with the ID

of the subscription. The existing O-MI node server allows to cancel or delete the subscriptions.

The second module is the **Webclient** which is basically a graphical representation of an interface aimed for the users and also for the developers. The benefit of having such an interface is that the users do not have to write manually all the XML messages and HTTP queries, rather everything is performed by just clicking on the defined buttons representing the functionality behind them. Such feature creates correct O-DF/O-MI messages automatically.

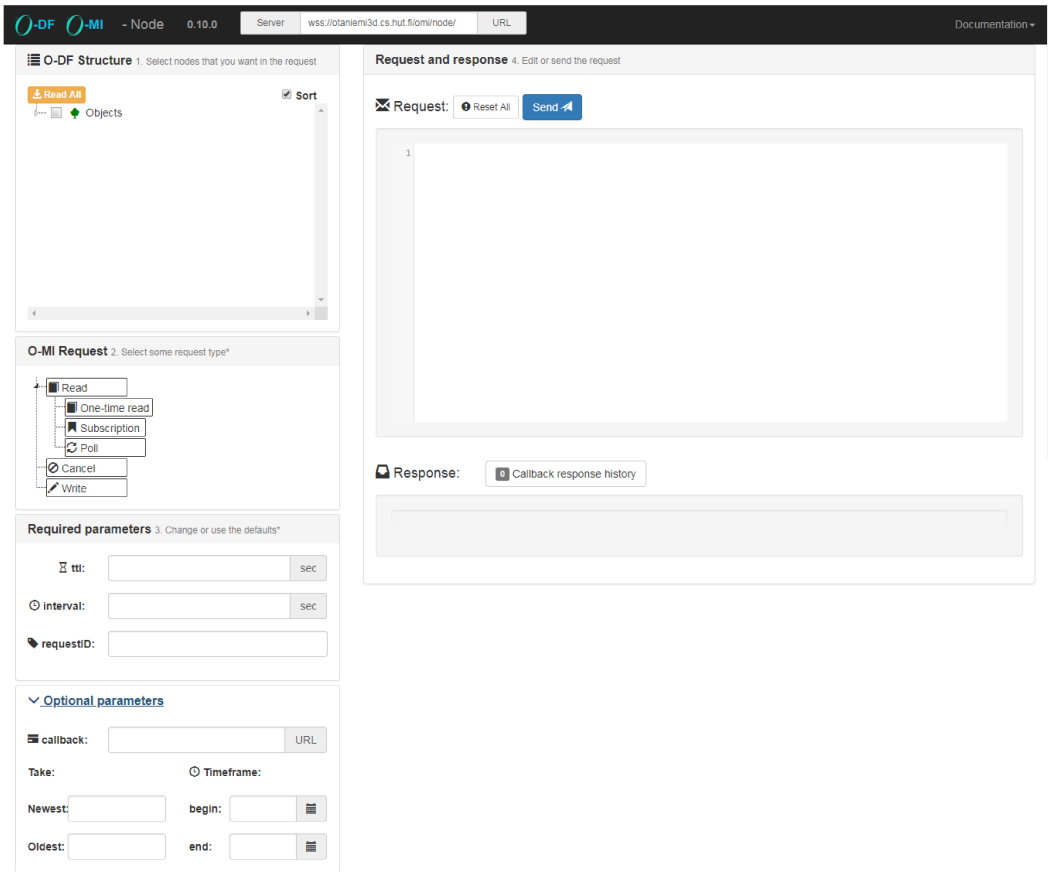

Figure 5.1: O-MI Webclient Interface

Figure 5.1 shows the O-MI webclient interface. All the required functionalities and buttons are placed in an order so that the user understands every next step to perform. The left panel of the webclient has three sections that are numbered and an optional parameters section. In the first section, nodes to be requested can be selected. If a user clicks on “Read All” button,

then a request is made to the server to read all the objects one time (One-time read in the second section is automatically selected by default and it can be changed). There is no exact number defined for the sub-elements or Inforitems under objects. One such tree structure can be seen in figure 5.2. Here, "FrontDoor" and "BackDoor" are the Infoitems while "Bedroom" and "Kitchen" are sub-objects. In order to select an item, one can do this by clicking on their check-boxes available.

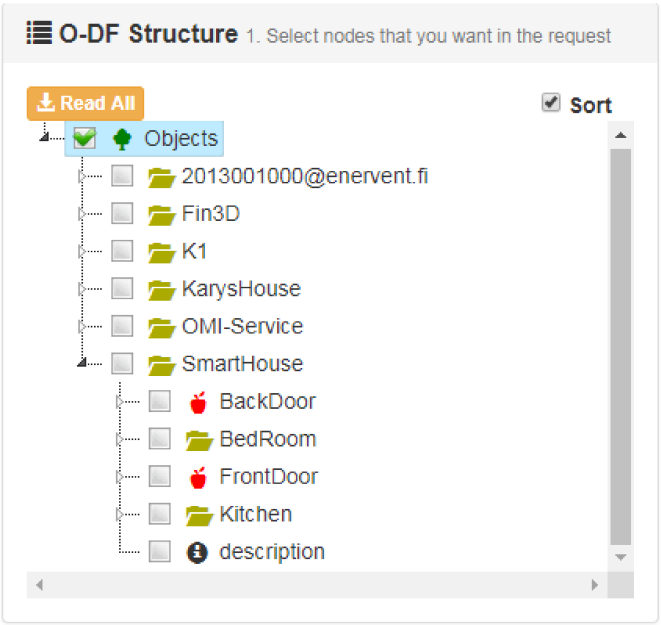

Figure 5.2: First section: O-DF Structure

Whenever an item is clicked on the left panel to be included in the request, webclient has the ability to translate the activity to the corresponding XML format for the request. Such XML message includes all the required parameters in accordance with the query type. The query type is specified in the second left panel. Figure 5.1 shows the types of queries a user can specify, i.e., Read, cancel and write. Under "Read" out of three (i.e, One-time read, subscription, and poll) one option can be selected. After choosing the type of request query, the user can then go to the third left panel to add the required parameters. There is another section of optional parameters list, which a user can also fill (see figure 5.1). The input parameters appear on the webclient depends on the type of request selected earlier in the second section. Optional parameters list includes a TTL parameter which is the shorter version of "Time to Live". It is the stated time in which the request

of the user is alive or in other words, remains valid. When TTL has reached i.e., when the specified time is over, the request is no longer working. TTL is basically the time to process a request and gets its response back. If a person opts for subscription, then the fields such as interval, request ID and callback becomes valuable. There is also an option available for the users to add "n" number of top newest and oldest vales to request. A "Timeframe" parameter with "begin" and "end" time specifications can also be seen under optional parameters section. By utilizing it, a user can specify the time between which he needs to request the values. When the user has filled in all the left panel values, it is translated into XML format on the right side of the webclient under the request UI section (see figure 5.3 where the user has selected "BedRoom" and "One-time read" options and corresponding XML message appeared on the section under request). It is considered as a raw XML which can be edited by the user. This makes the user learns the functionality of the protocol and he can visualize the correct way of requesting via O-MI/O-DF standard.

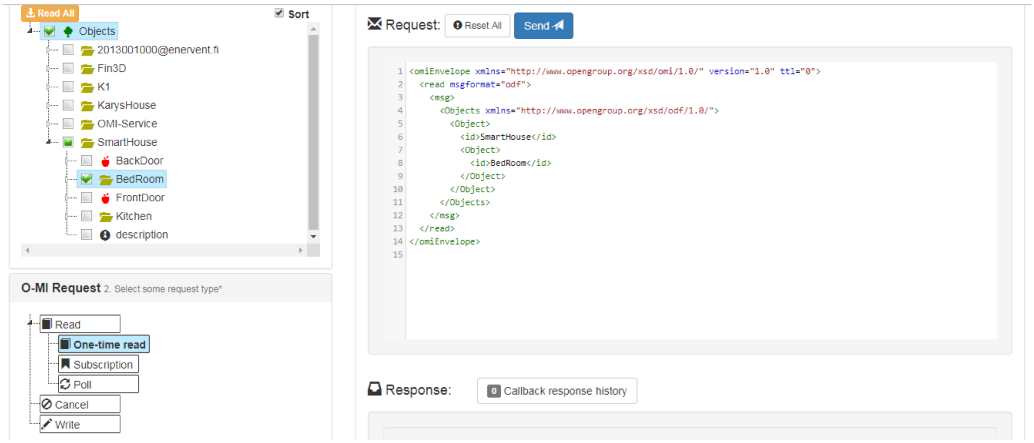

Figure 5.3: XML format of a request

When the desired request is generated, the user can then submit it to the server by pressing the "send" button. The user can receive the response on the below section on the right side of webclient and it cannot be edited by him.

Another module for reference implementation is "Agent". Agents are the actors that communicate with the core of O-MI node programmatically. An agent, generally, is a separate work thread that receives information from various data resources by utilizing specific protocols and changes them to objects that are easy to understand for the O-MI node core. The secure

techniques required for various agents are discussed in Chapter 6.

5.2 Reference Implementation Security Model

As all the product developments today require security incorporated in them against cyber-crimes, in the same way, security of the reference implementation is equally important. Reference implementation first had only IP whitelist mechanism for security. With IP whitelisting, trusted IP addresses or ranges of them are listed with which only trusted users can access the domain. In case of the reference implementation, IP whitelist was utilized for "write" functionality whereas "read" functionality was open for everyone. This method cannot be considered a proper security mechanism in its true sense. It did not even define access roles for the users/group of users and their associated rules for using the objects.

Therefore, in order to have a properly defined secure mechanism, a dedicated security model was created (which is now considered as version 1 of the security model). It developed the OAuth/certificate based authentication mechanism and access management rules based on the O-MI verbs and O-DF model. However, this version 1 has now become outdated and needs to be upgraded because of two reasons: 1) It does not provide any kind of support for O-MI read request permissions, and 2) It consisted of only specific authentication method implementations. The newer version of the security model has dealt with the above situation and has provided a better solution where Authentication and Authorization are separated into two individual services working together. The details of the Security Model version 2 are discussed in the following chapter which also compares the newer version with the older one and defines the new functionalities implemented.

Chapter 6

Security Model for O-MI Protocol

6.1 Introduction

In order to utilize O-MI node reference implementation in a practical environment, authentication and authorization are the basic functionalities. The administrators should be able to define policies and roles of all O-MI operations and object(s)/Infoitems of O-DF.

This chapter first discusses O-MI node security requirements in various cases and then explains in detail about the design architecture, interaction schemes and implementation of authentication/authorization modules.

6.2 O-MI Node security requirements

In the Internet of Things, security, privacy, and confidentiality are important aspects to be careful about. There is a lack of a standardized procedure for the security and its mechanisms in IoT. It is now an important fundamental need to define a valid secure system for the IoT environment. The first step would be to gather all the security requirements in different network scenarios same as in various use cases.

6.2.1 Network configuration based security

This section focuses on four main network configuration cases where clients/agents may belong to the same or different networks and interact with the O-MI node server. Each of these cases confronts a different security threat and has its specific requirements.

1. **Localhost or machine**: In this case, clients/agents are on the same host as that of the server. The security in such a scenario is considered trusted as the administrator would only accept the installation of that software that he believes to be authentic. In such situation, advanced security procedures such as digital certificates and request/response encryption/decryption processes are not applicable as they may overburden the server unnecessarily. The only security threat that can harm the server is from outside. A secure mechanism to safeguard the server is required to avoid the external threats.

2. **Unassociated hosts and same subnet**: In this case, O-MI node server and clients/agents are situated in the same subnet, however, on individual hosts/machines. The communication in the same subnet between the server and clients are carried out by utilizing the internal IP addresses. In this way, both parties do not have to go through the gateways and intermediary machines for the connectivity. Thus, creating a highly secured closed community. However, there can be such attacks where a machine in the network can become compromised by an attacker and that machine start sending the messages to that attacker or the third parties. Hence, in this case, major cyber threats can also occur from external potential attacks or from a hacker who can physically enter a network node.

3. **Identical Network and Non-identical subnets**: In order to understand this scenario, the structure of a big organization can be taken as an example. Big organizations can consist of various departments having various subnets where all these subnets are connected to a single main network of the organization. In this way, all nodes can share messages or resources by forwarding the requests/queries via gateways and utilizing the internal proxy machines. In this scenario as well, if a network is considered secluded from the Internet then requirements from the previous case set here. Furthermore, the nodes from different subnets usually do not have much knowledge about each other. Nodes present in one subnet only generally communicate with the nodes in another subnet via gateways where there is a possibility that gateways implement some secure mechanisms. However, there can be security threats where a hacker can sniff and spoof the information between the nodes (of different subnets) communication or can also override the rules. Therefore, it is considered to be the essential part of communication to have encrypted messages and to incorporate authentication mechanism in case if there is an even a chance that a node in a subnet

cannot be trusted completely.

4. **Internet (Non-identical networks)**: Internet consists of a number of various networks where there can be several subnets and nodes having different backgrounds. Such nodes are not very likely to be trusted and in such condition, machines need to incorporate the maximum security techniques they can. On the Internet, any node can be considered as a comprised one or a hacker, that is why clients or agents should get authenticated by the server and a secured encrypted iteration must take place between them.

6.2.2 Use-case based security

Not only on the network configuration, security requirements are also crafted according to the particular use case or scenario. Such use case usually consists of one's outlook in a specific environment and an individual's personal concerns, interests and needs. In the following, different cases are presented with different levels of security requirements, starting with no secure methodologies and ending with the highest security techniques [43].

1. **No or zero security requirements**: There exist various cases where data querying and information updating do not need to incorporate security techniques. For instance, a microwave oven may ask about the time to heat the bread that was placed inside it. As this information is not considered to be the sensitive one, utilizing passwords or digital certificates for such cases would be an extra burden. Similar is the example of a smart refrigerator that constantly after a defined period updates the user with the information regarding the expiry time of the products placed inside it. At first place, this data can also be considered not be a sensitive one. However, if the house is empty, there would be no food placed inside the refrigerator and this information can be a critical data. Generally, there exist many scenarios where no security is required and if secure mechanisms are added they make the system more complex, hard to utilize and might subject to issues related to privacy. There is also a probability that a system may not be used by people if it asks the user to identify or authenticate himself before use. However, in the IoT ecosystem, a data to be sensitive or not, assessing this might not be very straightforward. Hence, for IoT, this "no or zero security requirement" scenario is seldom used.

2. **Encryption of data and integrity check**: In applications where there is a possibility that an attacker can access the stored confidential

information or messages in between some communication, encryption of the data and integrity check becomes necessary. Data encryption also indicates the integrity of the data (however, not vice versa) which does not allow an external attacker to manipulate the information and the man-in-the-middle to read the data during communication without any notice. Today, information encryption and check of integrity can be performed simply because of the World Wide Web's widely supported HTTPS protocol that can be used straightly. This HTTPS protocol is almost supported by all the web browsers/agents. Even small enterprises have the required info for this and are able to set it up in low cost. As every company only requires to maintain their own certificate, there is no need for further effort for its management. All client applications (for instance, web browser) supports HTTPS and this level of security does not cause any kind of overhead. This also guarantees the identity of the data provider and also the integrity of the data provided.

3. **Authentication of every interacting body**: In the last scenario, HTTPS was utilized for data encryption and integrity. The server can also use signed digital certificates. These methods would make the client aware of a server's correct identity. In such a case, only one of the two communicating bodies is identified. There can be multiple situations where both interacting parties need to be authenticated. If the Internet is considered then authentication of the client is normally performed by entering his user-name and passwords. This is done over an encrypted medium. Furthermore, there can be a client-side certificate that can be used for authentication. HTTPS has the ability to assist this and it is considered a safe option as compared to the passwords that are usually termed as easy to guess. However, this authentication method can make the system more complicated and needs exceptional web server configuration modifications that might be difficult for some companies to incorporate. Moreover, saving the certificate on a client and the distribution of it can be a security risk. The usability of certificates is also not well known as many users do not accept this or simply ignore it when their browser pops the option of using it up. Thus, as this kind of authentication procedure involves all communicating bodies on an encrypted medium, this use case scenario is considered more challenging and demanding as compared to the previous one.

4. **Access control**: When a user is identified and authenticated by utilizing one of the techniques discussed above, it is then necessary to have access policies/restrictions to various data files and documents which a

user can retrieve. Like in an operating system where the user has permissions of particular operations to be performed on directories and files, similarly various network resources can only be accessed by the user if he has the specific rights.

In the security module v2, authentication and authorization modules have been implemented that authenticates the users, manages user and groups and their policies are defined to access objects. The next sections describe the requirements for such modules.

6.3 Security model requirements

One of the important requirements of the security model is to implement the desired features/functionalities without affecting the current O-MI implementation. That is why it was decided to have a separate security model that can be plugged into the current implementation. In the security model, two separate services (authentication and authorization) are introduced which makes the system more modular and can be changed with other implementations. All security requirements are formulated as follows:

- **O-MI node restrictions**: Unauthorized access to the resources are not allowed. Only the whitelisted consumers/users are given access to the O-MI node and have the right to perform the assigned functions over specific data objects. For example, an individual should not be provided with access controls to that O-MI node that publishes information intended for some specific group of users and not open to the public.

- **The type of operation restriction**: There are mainly two permissions: 1) read, and 2) write. It should allow only either to read or write or can be both for particular users. There should be "allow" and "deny" policies for the read and write operations and they should be able to be defined for any data object in the O-DF hierarchy.

- **Role (Group) based policies**: Every user has to be a part of some groups. The access policies are set for the groups and not for the individual users. A user can be in one or more groups. If a specific group is authorized, then a user belongs to it would be granted access based on his profile i.e., the group to which he is a part of. For example, a smart heater O-MI node must allow temperature reading for only those users from a group that is permitted to consume its data.

- **Recursive rules scheme**: The policies must be logical and perfectly detailed. Policies or rules should be inherited from parents to the child objects (similar to the OS file systems) and should not be vice versa.
- **Default rules**: There should be a default group that must implement default policies for any user. The default policies can consist of the allow/deny rules for the read and write permissions for any data object in the O-DF hierarchy.
- **Authentication procedure**: In order to configure rule-based permissions, there should be some mechanism to check the user or device identity that is trying to access the O-MI node. To minimize the server load, authentication can be performed with the help of the external services such as OAuth2.
- **Policies management console**: The administrator of the system should have a rules management interface (also called an administrator console in O-MI security model) where access rules of the groups are defined in a centralized way. The interface should have these functionalities: a) a display of all users in each group, b) a display of all groups in O-MI node, c) a display of access control tree for each group, d) option to change access control tree for each group, e) option to change data regarding any group (its name or its users), f)Add/Delete users, and g) Sign up new users.

From the above-mentioned requirements, it is pretty obvious that they are closely related to the commonly implemented security techniques. This is because it has been planned to make this security model similar to the well-known security mechanism of the Unix file system. It is decided to match its simplicity and effectiveness.

6.4 Design of the security architecture

In the context of the mentioned list of security requirements in the previous section, it is possible to setup requirement specific technologies, construct a comprehensive design and determine functionalities to be implemented. The fundamental approaches used for designing the security architecture are described as follows:

- **Autonomous authentication and authorization**: The security model has two parts: one is the Authentication module and the other

is the Authorization module. Authentication module manages users registration and their data. It handles authentication procedure and sessions. Authorization module has two sub-parts: the first one is the Access Control module that will process the requests made by users via O-MI Node and will authorize them. The second one is Administrator console that will be only accessible by super-user (or administrator). It will handle the adding of users to the groups and the related policies. These independent modules promote a modular approach which enables any use-case to integrate its own authentication or authorization modules.

- **Restful management of the permissions**: There is a RESTful method that permits O-MI administrators to add user rules and manage access control policies for the node overall. RESTful API must enable the administrator to (in a secure way) enter new users, remove or delete existing users and groups and also specify the access permissions.

- **New separate database**: In order to assure that the requirement to modify as less as possible the core O-MI node implementation, an individual new database should be proposed that handles new users and groups and their relationship. It also related the access policies with groups. This would have a drawback related to memory consumption and the performance of the system, however, it makes the whole system easier to maintain and code modularity is also achieved.

- **External OAuth2 provider**: In order to authenticate or register a user using the third party a feature called as external OAuth2 service provider should be used which includes login option with Facebook.

- **OAuth2 provider**: The authentication module should also act as OAuth2 Provider which offers OAuth2 services and capabilities that enables users to register their client applications on the authentication module and login with the obtained access tokens.

- **LDAP and local authentication**: More authentication options like Lightweight Directory Access Protocol (LDAP) and local username/password based methods can be added to ensure that the system (or authentication module) is operative in any scenario.

- **Certificates Extension**: Since the devices in IoT cannot have an account on Facebook, for the communication with the devices or machines, digital client-side certificates should be utilized. Certificates

should consist of mandatory credentials that are required to be verified by the server.

Database architecture: The security model contains a database that saves and manages consumers/users, their groups/roles, and related access controls. The database architecture can be seen in figure 6.1.

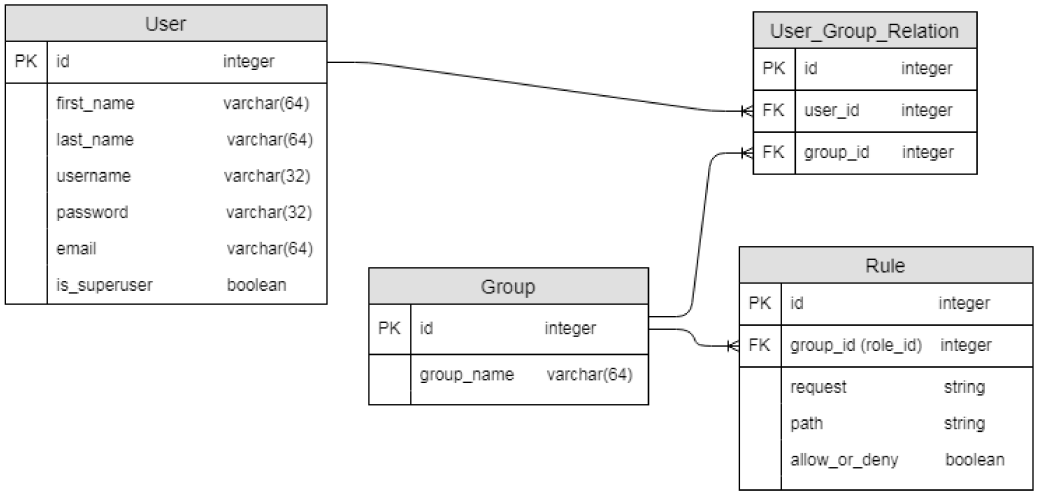

Figure 6.1: Database architecture for O-MI security model

It consists of four tables, i.e., *User, Rule, Group* and *User_Group_Relation*. When a user registers or logs in via Facebook or LDAP, his credentials will be saved in the *User* table. Password fields in the *User* table will only be used for local user registration. The field "is-superuser" is only visible to the administrator in the admin panel (webpage) to add superusers to the *User* table. The administrator also has the right to downgrade a superuser to the normal user and vice versa. The *Group* table contains the list of all groups. It also has a default group that is linked with default permissions for users who are not registered with O-MI authentication module (in other words, it is for anonymous users). The permissions of the default group can be modified only by the Administrator. Administrator or super-user can link users in *User* table with groups in *Group* table via *User_Group_Relation* table. *User_Group_Relation* table is a helper table that is used to achieve a many-to-many relationship between *User* and *Group*. Access policies will be stored in *Rule* table which allows or denies the access requests (i.e. read, write, delete, or call) to the specific path(s) in O-MI hierarchy. In *Rule* table,

the role_id/group_id column links with the ID in *Group* table. The field "Request" can have any request type from the read, write, delete, or call options. "path" is basically a path from root element in O-DF (i.e., Objects) till the requested element which includes all the middle nodes. "allow_or_deny" field is self-explanatory. The policies applied to an object, also be applied on its child objects and Infoitems recursively. There is a possibility to change the inherited policy on a specific node and override it with new rules manually.

Overall security model architecture: According to the list of requirements discussed above, the architecture of the overall security model is defined that consists of four fundamental components as shown in figure 6.2.

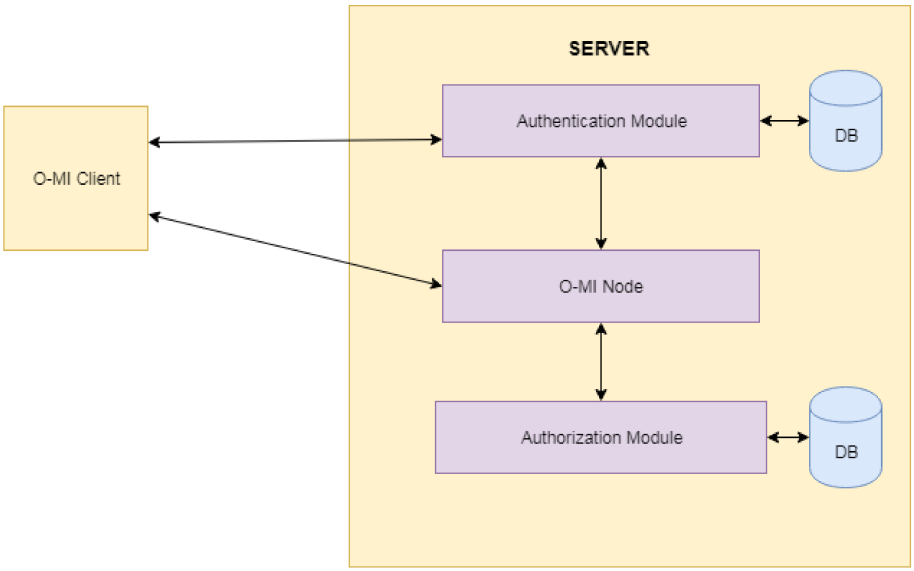

Figure 6.2: Overall O-MI security model architecture

- **Authentication**: It is the procedure of determining if a client/user is indeed who it declares to be.
- **Authorization**: It specifies the policies that assert who has the permission to perform which functionalities/operations.
- **O-MI Client**: It is a client application that enables the read and the write operations of the data utilizing O-MI node.

- **O-MI Node**: It is a server-side node that links multiple layers together to interact with O-MI client through O-MI/O-DF standards.

6.5 User Interaction scheme

The security model has been planned to have two separate autonomous submodules: authentication and authorization. Separating the authentication submodule establishes a modular structure and makes it easier to be modified into some other implementation platform (for instance, Fiware keyrock identity manager). Authorization is closely linked with the O-DF structure where there is a possibility to have partial authorization. In other existing authorization modules, one can either authorize or unauthorize completely with no middle way condition. In the proposed authorization structure, an administrator has the option to set access policies even for objects or Infoitems under objects, thus a partial authorize/unauthorize is possible. Separating the authentication and authorization functionalities is also a common practice in case of bigger systems. This assists in achieving a good control of scale and a single common security system requirement in the entire organization is also satisfied.
As it is decided to have two individual submodules of the security model for the O-MI reference implementation, there has to be an interaction scheme for each of these. They are described as follows.

6.5.1 Authentication mechanism

Authentication submodule has the responsibility to register and login users, manage authentication and handle sessions. There are three login options available for a user for authentication when he visits the O-MI client. These three options are 1) Local authentication login, 2) Login with Facebook, and 3) Login with LDAP credentials.

1. Local authentication login is a password based login where a user enters his username and password. The authentication module checks the user information in its database (in *User* table) and if it matches then the user is authenticated. If a user does not have an account then there is a sign-up form where he can register himself. The administrator also has an "admin panel" web page where he can register superusers.

2. OAuth2 based registration/authentication interaction scenario can be seen in figure 6.3. In this scenario, the user first interacts with O-MI

client (the web application) and selects to login with Facebook (registration with OAuth2 provider). O-MI client redirects the user to OAuth2 service provider's web page where he logs into his account and an authorization request is made that asks the user about his permission to let the O-MI client use his data. When the user accepts the authorization request, the access token is generated which is redirected to O-MI client. O-MI client then makes HTTP requests to OAuth2 service provider and gets access to the user's permitted information. After receiving the data, O-MI client looks for the user in the database. If this user is not already registered, his data will be entered into the database (in user table defined in figure 6.1). If no errors occur then session cookies (containing the JWT tokens) are set in the web browser of the user. Thus, the user is authenticated and has now the ability to request objects or Infoitems in O-MI node server (corresponding to his specified access rules).

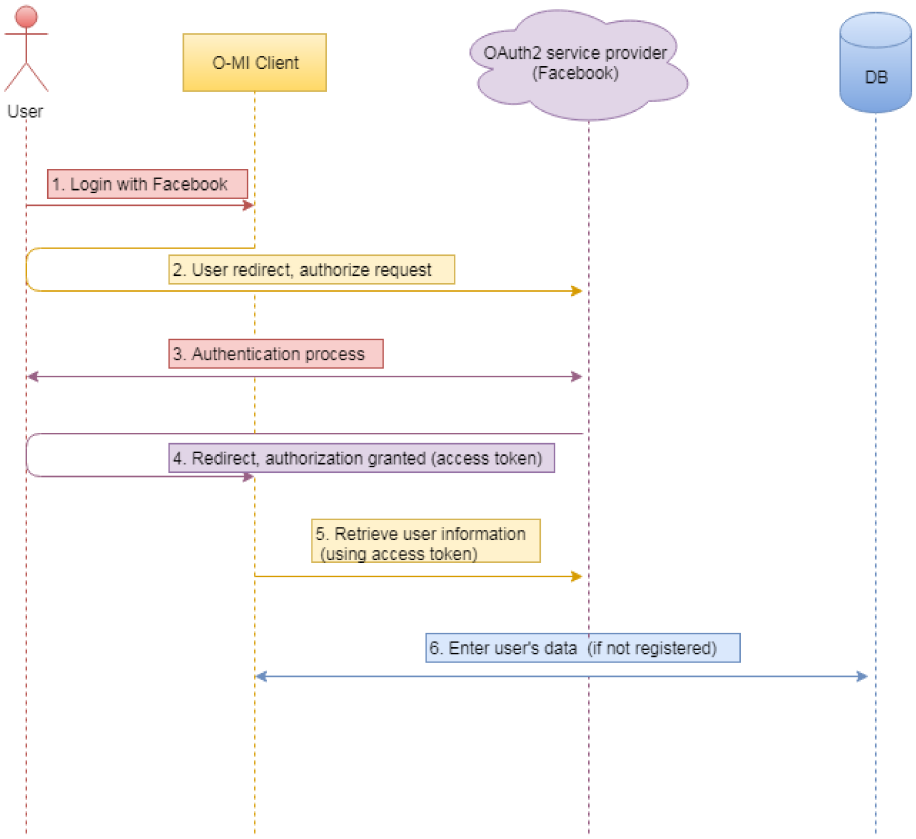

Figure 6.3: OAuth2 based authentication mechanism

3. LDAP based authentication scenario is shown in figure 6.4. In this case, a user enters his LDAP credentials to login to the authentication module. LDAP service should be running behind. The LDAP server is configured with a default MDB database that stores users and their groups. When a user enters his username and password, a search/bind mechanism is performed where a distinguished name (DN) of the authenticating user is searched in LDAP directory. After this, binding is again performed with the entered password by the user. This search has to provide only one result, otherwise, authentication will fail. Once the user is authenticated, the database (of the security model) is checked for the user. If his details are not present then the database is updated with this user's information.

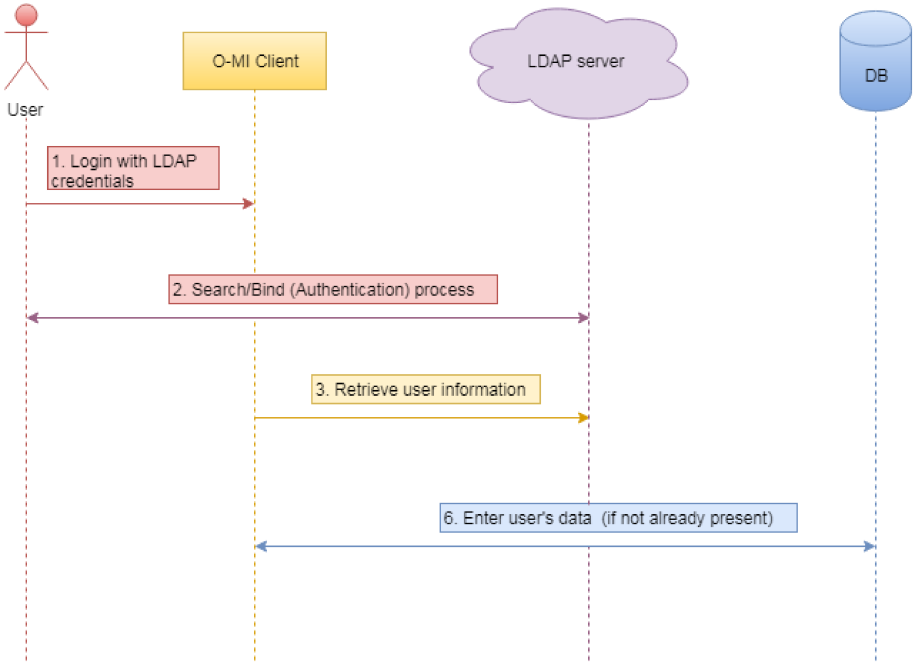

Figure 6.4: LDAP based authentication mechanism

6.5.1.1 Certificate based authentication

All the mentioned authentication techniques in above are applicable only if the interacting body is a human being using the web browser. If the user-agent is not the web browser i.e., a device running IoT framework then there should exist a mechanism for their authentication process.

According to [60], Smart home installation was performed where O-MI node along with the access control module was used. It had multiple sensors, all connected to one central gateway. The gateway connects the house with the Internet. ISP (Internet Service provider) assigns IPs to the devices that may modify with time. Using this methodology where dynamic IPs are consumed is a common process and justifies the reason behind abandoning the process of IP whitelisting. In this situation, obviously, IoT machines cannot possess social accounts such as Facebook or register via LDAP. To solve this issue, client-side SSL certificate mechanism is taken into consideration [1].

In digital certificate methodology, a certificate from the certificate authority (CA) is bought by the server. When the client interacts with the server via HTTPS, it gets the certificate from the server. The client then verifies the validity of the certificate with the CA [26]. The client also sends his certificate to the server, thus, performing a mutual authentication process where both parties exchange their certificates with each other. This ensures the O-MI node server as well that it is communicating with the correct client. A figure representing this process is shown below.

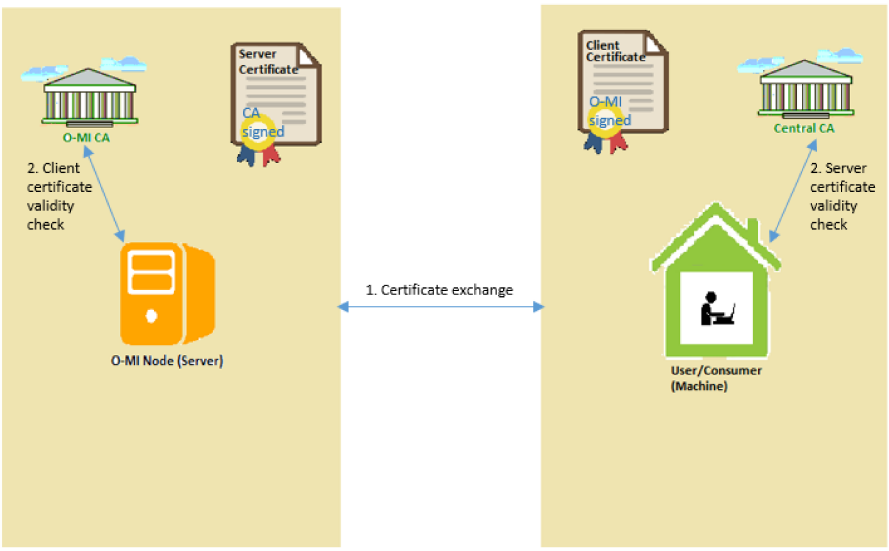

Figure 6.5: Certificate exchange mechanism

The certificate that the client sends is issued by the O-MI server. O-MI server generates a certificate and adds a digital signature by using the private key of the server. In order to transfer the certificate on the network,

[1] Adapted from security module version 1 by Tuomas Kinnunen

there should be a trusted software installed and running on the target device. The other option is to manually add the certificate to the client machine. At mutual authentication point, when HTTPS connection is formed between the home gateway and O-MI node, O-MI node also receives the client/consumer certificate. The certificate is then verified where O-MI node uses its public key and checks that the certificate is signed by it and finally claims the device to be authenticated [60].

Now, the consumer's identity must be stored and its access rights must also be defined. For this purpose, the consumer's e-mail address is extracted from the certificate by the O-MI node and is saved in the database with the access policies when the consumer/client registers. Upon the access request, the e-mail is derived from the certificate and the access request is matched with its allowed/denied paths. In a real-world environment, a device's serial number is taken into account rather than the e-mail address as its ID or can be a globally unique identifier as defined in [34] [60].

6.5.2 Authorization mechanism

Authorization submodule [2] has two parts: 1) Administrator Console, and 2) Access Control module. When a user is registered in the system, the administrator has the ability to add him in particular groups, each of them having their own defined access policies to retrieve specific objects. To perform this, a user interface was developed which was named as the access management tool (also known as administrator console)[3]. It is only accessible by the administrator or superuser and handles adding of users to groups and manages access policies on the database defined in section 6.4. The Access Control module will process the requests made by users via O-MI Node and will authorize them.

Access management tool: The interface of this tool is very similar to the one of O-MI node and has some extended features. It can only be accessed by administrator or superuser. They have the ability to manage user policies defined with respect to the groups. They can create new groups, change or remove them and can add or delete users from specific group(s). This web interface also links the left panel tree hierarchy (defined in the reference implementation webclient) that retrieves the object(s) presented on the O-MI node and access rules for all nodes of this O-DF structure are also specified. The rules are read, read/write, and no-access for every node of the tree.

[2] Authorization module is collectively developed with Tuomas Kinnunen

[3] The tool in this version 2 is adapted from [60].

Authorization submodule interacts with O-MI node via HTTP API to have a more fine-grained management of the read request results. O-MI node requests the authorization submodule for the policies of the user or/and group and the kind of request. The policies are defined as "allow <path>" and "deny <path>". Paths are basically from the root element until the requested element in the O-DF structure. For instance, there is a read functionality defined as "allow /objects" and "deny /Objects/myprivatelaptop/" which allows a particular user or group to read everything under objects except my private laptop.

An HTTP GET request generated will be formulated as:

```
GET/<o-mi request type>/<user or group>? <o-mi request parameters>
```

The response of this get request will have access rules (allow and deny) for the requested type and user/group/role. The real filtering which is based on the allowed path received from security model will be performed in a new O-MI Node AuthorizationExtension that implements this latest API.

In the context of the Authorization perspective, security model version 1 [60] did not possess the ability to filter out the requests, as O-MI permits the entire Objects "read" access without considering its sub-elements that might have some forbidden rules for specific elements. Accordingly, it was decided to have filtration process to be performed at O-MI node rather than communicating the entire request to the authorization module. Thus, the extension is named as internal authorization extension that applies the latest authorization API for interacting with Authentication and Authorization modules.Compared to the previous model, more adaptable configuration settings (for communication with both modules) are incorporated in the O-MI node of security model version 2. This has simplified the integrating, updating and developing (new modules) processes.

The overall interaction scheme of both the Authentication and Authorization modules with O-MI client and O-MI node is shown in figure 6.6. The steps involved in the interaction scheme are explained as follows:

1. User/consumer first logs in or registers himself (if not already) by choosing any of the four authentication methods described above.

2. When the user/consumer successfully logs in, the authentication module returns back a token.

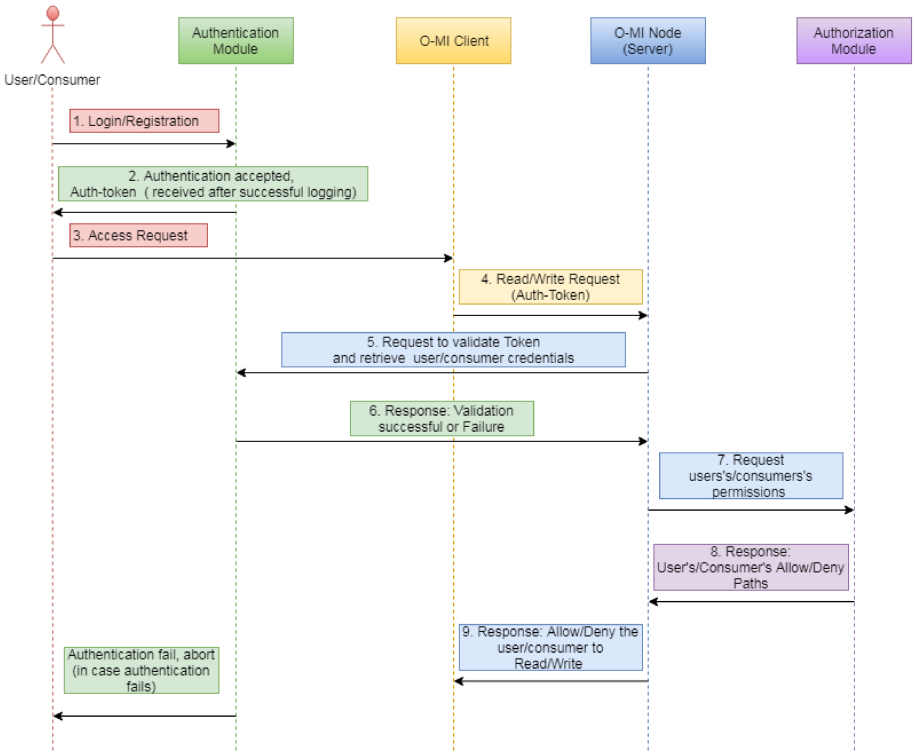

Figure 6.6: Overall interaction mechanism

3. The token is then redirected back to O-MI client application.

4. O-MI client can then forward the request to the O-MI node using the token in query parameter or HTTP header.

5. In order to validate the token, O-MI node checks it with Authentication module that whether the token is still valid or has become invalid.

6. Authentication module will send the response back to O-MI node. If the user/consumer is valid, it will forward that result to O-MI node along with user/consumer ID and optionally their policies (to be forwarded to the authorization module).

7. In case of successful validation, O-MI node interacts with Authorization module and send the request that includes the type of request (read, write, cancel), the ID of the user/consumer and optionally the policies of the requested user/consumer. In case of failure, O-MI node will request for the default rules instead.

8. Authorization module will send a response that consists of O-DF paths i.e., allow and deny (they should be subtracted from the allowed ones) paths specified for that user/consumer.

9. O-MI node will then filter or process the requests according to the policies it acquired from Authorization module and will return back the final result to the O-MI client.

6.6 Implementation of security model

In the previous sections, the design architecture and interaction flows were described. This section discusses the technologies utilized and implementation details. The security module is developed using Python and Scala languages. HTTP frameworks that were utilized in the construction of security model were Django web framework (for developing authentication module) and Akka HTTP. It supports SQL databases and LDAP directory services. Furthermore, JSON Web Tokens (JWTs) are also taken into account.

The new Auth (Authentication and Authorization) API feature utilized in O-MI node is used for setting up external services for authentication and authorization. External services refer to the separate processes that have the ability to run on the same or different device. Overall, the process goes like this: O-MI node first interacts with the authentication service, then communicates with the authorization service and finally filters the requests.

Authentication and Authorization APIs possess the feature of adaptability and flexibility. There exist configurable options for them which can be found in the configuration file of O-MI node. However, only the fixed format is when Authorization service sends a fixed response (as described in step 8 in figure 6.6).

There can be multiple configurable methods to extract the input for authentication service from the original O-MI request. Thus, the authentication session credentials can be sent in: omiEnvelope attributes, HTTP cookies, Authorization HTTP header, other HTTP headers, and URL query parameters. Utilizing omiEnvelope attribute is the recommended method that establishes the functionalities even if any other transport protocols are being used. One such example format of the omiEnvelope attribute is given in figure 6.7 which also includes a token.

The Authorization service input can also be deduced using the configurations in the same way. Utilizing the original request, values can be extracted. It is also possible to get them from the response of the earlier request to the authentication service. If the Authorization service receives a response mes-

```
<omiEnvelope xmlns="http://www.opengroup.org/xsd/omi/1.0/"
             version="1.0"
             ttl="5"
             token="eaJ9eYBiOiPMV1W...">
```

Figure 6.7: omiEnvelope example format

sage from the Authentication service that consists of the HTTP error code then the entire procedure results in an unauthorized response or if it is configured then default group rules (that contains a list of anonymous users) are applied.

Lastly, Authorization module sends a fixed JSON response that contains two O-DF path lists i.e., "Allow paths" and "Deny paths". These lists are utilized to process the O-DF formation in the incoming request using the filtering operation i.e., (O-DF intersect allow-policies) and deleting the deny-policies (see figure 6.8 [4]). The filtered or processed result is then added into the request rather than the original one and then further processing continues.

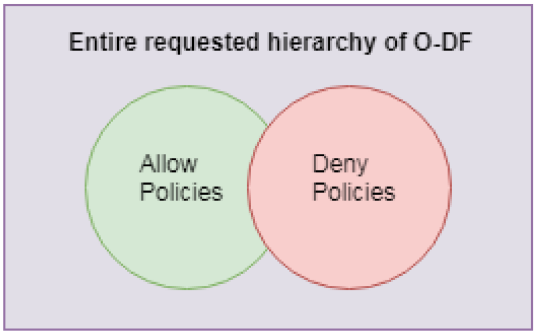

Figure 6.8: Allow and Deny rules Filtration

Anyhow, there is a possibility for the Authorization service to implement a complex structure for the policies instead of simply returning the allow/deny paths. In the reference implementation, group-based or role-based methodology was considered. This technique consists of three concepts: 1) Users or consumers, 2) Groups or roles, and 3) Policies or rules. Consumers can have multiple groups or roles and policies/rules are defined for each one of

[4]In figure 6.8, the visible green portion indicates the filtered O-DF for the response.

them. Providing the user/consumer ID, this methodology has the ability to determine the combined/overall permissions.

6.7 Comparison: Security Model v1 and Security Model v2

This section discusses the main features of security models version 1 and version 2. It highlights the important aspects and the reasons behind the decision of implementing the second version.

6.7.1 Recap of Security model version 1

This first version consists of two fundamental parts: 1) Authentication submodule, and 2) Access Control submodule. Authentication submodule manages the registration of the new customers/users and their data via external OAuth2 provider service (for example, Facebook). Furthermore, it handles authentication procedures and session management. Access Control submodule further divided into two parts: 1) Administrator console, and 2) Access control middleware. The first one is a tool developed for administrators to handle users, their groups and associated rules. The later part manages and authorizes the user access requests. It receives the incoming O-MI requests from the O-MI node server via HTTP to be accepted, rejected or processed/filtered [60].

6.7.2 Current Security model implementation

There was a need to upgrade the previous security model by providing a newer version for two reasons. Firstly, the version 1 did not support the read request permissions for O-MI. Secondly, the implementation was specific to the selected authentication techniques. The newer version has a new Auth API (that consists of Authentication and Authorization services) in O-MI node and new reference implementation for API.

In the newer version, there are separate autonomous Authentication and Authorization services. More authentication options are provided such as local authentication, OAuth2-based, LDAP-based and certificate based. Authentication module also behaves as identity manager to authenticate users/ consumers and verify that an authenticated consumer has sent the request. Furthermore, web technology also supports the separation of services. Hence, there is a possibility that the entire module for authentication can be replaced with the already developed identity manager implementations if required.

In the context of authorization services, the version 1 was not able to manage to filter the requests completely. In the previous implementation, O-MI protocol supported the read access for all entities that come under Objects without considering the denied access that might be applied to some child elements. That is why it was decided to have a filtration process to occur at O-MI node. This extension implementing a newer API was named as internal authorization extension. Thus, more flexible configuration settings are added to develop new modules, update the older ones or integrate them.

A detailed technical overview of version 1 and version 2 can be seen in the figures 6.9 and 6.10.

Auth API Properties	Version 1	Version 2
Request Type in O-MI	Write	Read, write, call, delete
O-DF structure is sent	Yes, in leaf node paths format	no, filtering happens in the O-MI Node
Authentication session credentials can be sent in		
- **Cookie**	Only JSESSIONID	Yes, configurable
- **HTTP Authorization header**	No	Yes, configurable
- **Other HTTP headers**	Only X-SSL-CLIENT	Yes, configurable
- **omiEnvelope attributes**	No	Yes, configurable
- **URL query parameters**	No	Yes, configurable
API Request type	One HTTP POST	One or two HTTP requests, configurable
API Request data format	JSON fixed format	configurable, data can be put into JSON, URL query and headers
API Response data format	JSON fixed format	Authentication: configurable (same way as session credentials), Authorization: JSON fixed format
API Response data returned	Authorized, Unauthorized or Partial (allowed paths list)	allowed paths list, denied paths list
O-DF Path level permission control	Limitations: - Only write request - fully block or allow Object or InfoItem	Benefits: - all O-DF requests - fully block or allow Objects, Object or InfoItem

Figure 6.9: Table for Auth APIs comparison between version 1 & version 2 [41]

Reference Implementation	O-MI Security Model	O-MI Authentication Module O-MI Authorization Module
Supported authentication methods		
- **local user and password registration**	no	yes
- **external OAuth 2.0 provider**	yes	yes
- **OAuth 2.0 (provider)**	no	yes
- **HTTPS client certificate**	yes	yes
Default session storage	cookie, In-memory session database	JWT token
User database supported	SQL	SQL, LDAP
Configuring and extending related		
- **Programming language**	Java	Python, Scala
- **Http Framework**	Jetty	Django, Akka-http

Figure 6.10: Table for security model comparison between version 1 and version 2 [41]

Chapter 7

Conclusion

7.1 Summary of research

The focal point of this thesis was to create an appropriate security model for the O-MI protocol. First of all, it introduced the IoT concepts, related challenges, the need for a unified IoT protocol standard, and security importance. It discussed the two basic security approaches for a system i.e., authentication and authorization and described various secure techniques in each approach with their pros and cons respectively. These included one-time passwords, context-based authentication, certificate-based authentication, biometric credentials, multiple single sign-on techniques (SAML, OAuth 2.0 and OpenID) and LDAP methodology.

Secondly, this thesis highlighted Open Group's O-MI/O-DF protocol features suitable to be known as a standardized protocol. The Open Group's vision to create a similar standard approach as that of HTTP for the Internet was aimed. It discussed the details of O-MI/O-DF standard which included some important attributes such as subscription feature, independent transport protocol, and support for various message formats. Next, it explained the reference implementation of this protocol which is currently being utilized and has the ability to incorporate any data of supplier/user scenario. The basic elements such as web client and O-MI node were described and related examples were presented. Then the need to secure O-MI/O-DF protocol was paid attention. Originally, the only IP-whitelisting mechanism was incorporated. After understanding the requirement of a fully developed security model, a detailed secure architecture was developed (known as security model version 1). Today, this version 1, however, needed to be updated with the more latest technology available. Therefore, a new security model known as version 2 was created. It was planned to build an external plug-in security

model that will provide authentication and authorization capabilities to the current reference implementation of O-MI/O-DF.

Thirdly, the thesis focuses on the security requirements, designing the architecture, interaction schemes and implementation details. This part of the thesis first discussed network based requirements and considered various use-case based security needs. Two fundamental functionalities were highlighted i.e., authentication and authorization. Primary design plans were formed based on the presented requirements such as database architecture, access permissions management and authentication methods. Interaction flows were described for OAuth 2.0, LDAP and certificate-based techniques. Administrator console and access module functionalities were presented and finally, a technical comparison with older security model was performed.

7.2 Study Analysis

The aim of this thesis was to develop a suitable and up-to-date security model for the O-MI/O-DF protocol. Autonomous authentication and authorization modules are created and it is possible to integrate them with any existing software system. Multiple authentication options are provided as compared to the previous version of the security model and fine-grained read permissions are specified. New Auth API (authentication and authorization) is developed for O-MI node and new reference implementations are considered for the API. The procedure to apply policies to the groups, users and various data entities was described, thus providing the way to set rules for real IoT devices generally. IoT related legal and privacy matters for the exchange of information were not included in this thesis.

7.3 Future concerns

This thesis concludes that O-MI/O-DF is a proposed unified standard protocol for IoT ecosystem and the security model of such system is highly essential that provides safe, reliable and secure communication between the consumers and devices.

The working of the developed security model has been verified by testing using the local systems and are further planned to incorporate them in various use case scenarios. One of the authentication methods was based on digital certificates and is considered more trusted method as that of conventional username/password technique. However, the distribution of such certificates via Internet connections might cause further threats. There is a possibility

that the producer of a product might incorporate the certificate into the device even before it is sold and hardware expert may create possibilities that no one with the possession of the device able to compromise these certificates or steal them in any way. However, certificates are valid for a specific period of time and expire after that. Consequently, this requires to update them on every machine manually which can be up to millions in number. Another option is to automate the creation and regeneration of client-side certificates. This can be done by generating a public-private key pair by the client. The client sends his public key to the server with the certificate request along with his additional data such as his e-mail address, company name, and location. A server creates certificates with the client's public key and sends it to the client that uses his private key to decrypt it. The server-client connectivity is protected utilizing HTTPS method. However, this is possible only if the server trusts that he is talking to the correct person/client.

Today, there can be multiple ways to implement authentication and authorization techniques. This research has put light on the importance of the latest secure methods that should be incorporated to deal with the rapidly emerging cyber attacks. This thesis supports the implementation of a unified IoT standard and security concepts that are independent of use-cases, and recommend further software systems a useful set of acclaimed authentication and authorization technologies.

Bibliography

[1] Advantages and disadvantages of certificate authentication. SSH.com. https://www.ssh.com/manuals/clientserver-product/53/ch06s03s04.html. Accessed 24 July 2018.

[2] Certificate-based authentication (sun java system directory server enterprise edition 6.3 reference. Oracle). https://docs.oracle.com/cd/E19575-01/820-2765/6nebir7eb/index.html. Accessed 12 July 2018.

[3] Chapter 1: Overview of authentication and authorization technologies and solution end states. Microsoft TechNet. https://technet.microsoft.com/en-us/library/bb463152.aspx#ECAA. Accessed 1 August 2018.

[4] Context-based authentication. ThreatMetrix. https://www.threatmetrix.com/digital-identity-insight/solution-brief/context-based-authentication/. Accessed 11 July 2018.

[5] Learn about ldap. LDAP.com. https://ldap.com/learn-about-ldap/. Accessed 2 August 2018.

[6] Network authentication, authorization, and accounting: Part one - the internet protocol journal - volume 10, no. 1. Cisco. https://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-35/101-aaa-part1.html. Accessed 12 July 2018.

[7] Security issues. OpenID Review. https://sites.google.com/site/openidreview/issues Accessed 31 July 2018.

[8] Single sign-on (sso). TechTarget. https://searchsecurity.techtarget.com/definition/single-sign-on. Accessed 25 July 2018.

[9] Strong authentication best practices — safenet authentication security. Gemalto. https://safenet.gemalto.com/multi-factor-authentication/strong-authentication-best-practices/?LangType=1046. Accessed 11 July 2018.

[10] The Open Group. http://www.opengroup.org/aboutus Accessed 4 August 2018.

[11] The Open Group. An introduction to Internet of Things (IoT) and lifecycle management.

[12] Understanding authentication, authorization, and encryption. BU TechWeb. www.bu.edu/tech/about/security-resources/bestpractice/auth/. Accessed 5 July 2018.

[13] What is openid?. OpenID. https://openid.net/what-is-openid/. Accessed 31 July 2018.

[14] The Open Group. Open data format (o-df), an open group internet of things (iot) standard, 2014. http://www.opengroup.org/iot/odf/ Accessed 5 August 2018.

[15] The Open Group. Open messaging interface (O-MI), an open group internet of things (IoT) standard, 2014. http://www.opengroup.org/iot/omi/index.htm Accessed 6 August 2018.

[16] Active directory domain services overview. Microsoft Docs, May 2017. https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview. Accessed 1 August 2018.

[17] Why iot security is so critical. The IOT Magazine, January 2017. https://theiotmagazine.com/why-iot-security-is-so-critical-381f4e7c29fc. Accessed 4 July 2018.

[18] The 5 worst examples of iot hacking and vulnerabilities in recorded history., February 2018. https://www.iotforall.com/5-worst-iot-hacking-vulnerabilities/. Accessed 3 July 2018.

[19] Gartner says worldwide iot security spending will reach $1.5 billion in 2018. Gartner, Inc, March 2018. https://www.gartner.com/newsroom/id/3869181. Accessed 4 July 2018.

[20] The importance of security by design for iot devices, February 2018. https://www.redalertlabs.com/blog/the-importance-of-security-by-design-for-iot-devices. Accessed 4 July 2018.

[21] Anicas, M. An introduction to oauth 2. DigitalOcean, July 2014. https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2 Accessed 30 July 2018.

[22] Atzori, L., Iera, A., and Morabito, G. The internet of things: A survey. *Computer networks 54*, 15 (2010), 2787–2805.

[23] Banafa, A. Three major challenges facing iot, March 2017. https://iot.ieee.org/newsletter/march-2017/three-major-challenges-facing-iot.html. Accessed 2 July 2018.

[24] Baumhof, A. Why now is the time to implement context-based authentication, February 2015. https://www.linkedin.com/pulse/why-now-time-implement-context-based-authentication-andreas-baumhof. Accessed 11 July 2018.

[25] Borgohain, T., Borgohain, A., Kumar, U., and Sanyal, S. Authentication systems in internet of things. *arXiv preprint arXiv:1502.00870* (2015).

[26] Brody, H. How HTTPS Secures Connections: What Every Web Dev Should Know. July 2013. https://blog.hartleybrody.com/https-certificates/ Accessed 27 August 2018.

[27] Buda, A., Främling, K., Borgman, J., Madhikermi, M., Mirzaeifar, S., and Kubler, S. Data supply chain in industrial internet. In *Factory Communication Systems (WFCS), 2015 IEEE World Conference on* (2015), IEEE, pp. 1–7.

[28] Corser, G. Why iot security is so important - and what to do about it., August 2017. https://www.iottechnews.com/news/2017/aug/04/why-iot-security-so-important-and-what-do-about-it/. Accessed 3 July 2018.

[29] De Clercq, J. Single sign-on architectures. In *Infrastructure Security*. Springer, 2002, pp. 40–58.

[30] Dickson, B. 4 major technical challenges facing iot developers., July 2016. https://www.sitepoint.com/4-major-technical-challenges-facing-iot-developers/. Accessed 2 July 2018.

[31] Europol. Mastermind behind eur 1 billion cyber bank robbery arrested in spain, March 26 2018. `https://www.europol.europa.eu/newsroom/news/mastermind-behind-eur-1-billion-cyber-bank-robbery-arrested-in-spain`. Accessed 12 June 2018.

[32] Evans, D. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper 1*, 2011 (2011), 1–11.

[33] Främling, K. Tracking of material flow by an internet-based product data management system. *Tieke EDISTY magazine*, 1 (2002).

[34] Främling, K., Holmström, J., Ala-Risku, T., and Kärkkäinen, M. Product agents for handling information about physical objects. *Report of Laboratory of information processing science series B, TKO-B 153*, 03 (2003).

[35] Fronczak, T. Oauth: Pros and cons of oauth. Social Technology Review, February 2011. `http://www.socialtechnologyreview.com/articles/oauth-pros-and-cons-oauth` Accessed 31 July 2018.

[36] Gallagher, S. Atlanta city government systems down due to ransomware attack. Ars Technica, March 2018. `https://arstechnica.com/information-technology/2018/03/atlanta-city-government-systems-down-due-to-ransomware-attack/`. Accessed April 2018.

[37] Hardt, D. The oauth 2.0 authorization framework. Tech. rep., 2012.

[38] Hauben, R. From the arpanet to the internet. *TCP Digest (UUCP)* (2001).

[39] Jankowski, S., Covello, J., Bellini, H., Ritchie, J., and Costa, D. The internet of things: Making sense of the next megatrend. *Goldman Sachs* (2014).

[40] Jara, A. J., Ladid, L., and Gómez-Skarmeta, A. F. The internet of everything through ipv6: An analysis of challenges, solutions and opportunities. *JoWua 4*, 3 (2013), 97–118.

[41] Jérémy ROBERT, Sankalp Ghatpande, N. T. K. Context-Sensitive Security, Privacy Management, Adaptation Framework v2. bIoTope. Planned to be published in 2018.

[42] Karpinski, M., Senart, A., and Cahill, V. Sensor networks for smart roads. In — (2006), IEEE, pp. 306–310.

[43] Kary Främling, David Potter, K. S. M. N. H.-H. DR13.0: PROMISE End-to-End Security White Paper.

[44] Kopetz, H. *Real-Time Systems - Design Principles for Distributed Embedded Applications*. Real-Time Systems Series. Springer, 2011.

[45] Krebs, B. Krebsonsecurity, September 2016. https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/. Accessed 4 July 2018.

[46] Lightfoot, J. Authentication and authorization: Openid vs oauth2 vs saml. Atomic Object, May 2016. https://spin.atomicobject.com/2016/05/30/openid-oauth-saml/ Accessed 31 July 2018.

[47] Mikolajczak, C. An introduction to saml (security assertion markup language), February 2017. https://www.secureauth.com/resources/blog/introduction-to-saml. Accessed 26 July 2018.

[48] Miniwatts Marketing Group. Internet world stats, June 2018. https://www.internetworldstats.com/emarketing.htm. Accessed 12 June 2018.

[49] Newman, P. The internet of things 2018 report: How the iot is evolving to reach the mainstream with businesses and consumers, February 26 2018. www.businessinsider.com/the-internet-of-things-2017-report-2018-2-26-1?r=US&IR=T.. Accessed 19 June 2018.

[50] Otemuyiwa, P. How saml authentication works. Auth0, December 2016. https://auth0.com/blog/how-saml-authentication-works/ Accessed 26 July 2018.

[51] Parker, S. Understanding the physical damage of cyber attacks., October 2017. https://www.infosecurity-magazine.com/opinions/physical-damage-cyber-attacks/. Accessed 3 July 2018.

[52] Rambo, S. Iot interoperability: Where it stands and what comes next., Nov. 2016. https://www.rcrwireless.com/20161031/internet-of-things/iot-interoperability-tag31-tag99. Accessed 3 July 2018.

[53] Sachgau, O. Smartwatches can be used to steal pin, researcher shows. TheRecord.com, March 2016. https://www.therecord.com/news-story/

6375960-smartwatches-can-be-used-to-steal-pin-researcher-shows/. Accessed 4 July 2018.

[54] Shinder, D. Understanding and selecting authentication methods. TechRepublic. https://www.techrepublic.com/article/understanding-and-selecting-authentication-methods/. Accessed 12 July 2018.

[55] Sutherland, L. The weaponization of iot: Rise of the thingbots, April 2017. https://securityintelligence.com/the-weaponization-of-iot-rise-of-the-thingbots/. Accessed 4 July 2018.

[56] Tiwari, A., Sanyal, S., Abraham, A., Knapskog, S. J., and Sanyal, S. A multi-factor security protocol for wireless payment-secure web authentication using mobile devices. *arXiv preprint arXiv:1111.3010* (2011).

[57] Wortmann, F., and Flüchter, K. Internet of things - technology and value added. *Business & Information Systems Engineering 57*, 3 (2015), 221–224.

[58] Xia, F., Yang, L. T., Wang, L., and Vinel, A. V. Internet of things. *Int. J. Communication Systems 25*, 9 (2012), 1101–1102.

[59] Yang, S.-H. Internet of things. In *Wireless Sensor Networks*. Springer, 2014, pp. 247–261.

[60] Yousefnezhad, N., Filippov, R., Javed, A., Buda, A., Madhikermi, M., and Främling, K. Authentication and access control for open messaging interface standard, 2017.

[61] Zachariah, T., Klugman, N., Campbell, B., Adkins, J., Jackson, N., and Dutta, P. The internet of things has a gateway problem. In *Proceedings of the 16th international workshop on mobile computing systems and applications* (2015), ACM, pp. 27–32.