USING BLOCKCHAIN TO SUPPORT PROVENANCE IN THE INTERNET OF THINGS

A Thesis Submitted to the

College of Graduate and Postdoctoral Studies

in Partial Fulfillment of the Requirements

for the degree of Master of Science

in the Department of Computer Science

University of Saskatchewan

Saskatoon

By

Emmanuel Kaku

PERMISSION TO USE

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science

176 Thorvaldson Building

110 Science Place

University of Saskatchewan

Saskatoon, Saskatchewan

Canada

S7N 5C9

ABSTRACT

The Internet of Things (IoT) has gained traction in all sectors and pervades all spheres of our lives. With statistics projecting an increase in the number of devices by 87% as well as increase in security concerns, traceability within this IoT will become a major problem. As more devices communicate with each other via the Internet, it will be crucial to determine the origins of requests and responses. Being able to store records related to the life cycle of requests and responses in an immutable form will provide documentary evidence that will help to establish transparency and accountability within the IoT. Previous works employed provenance techniques to address this problem but focuses on the request perspective. However, little or nothing has been done regarding the response perspective. Consequently, this thesis proposes and develops a blockchain-based provenance system to trace bi-directionally the sources of requests and responses in the IoT. This is achieved through the investigation of historical communication records. Furthermore, a performance evaluation of the system is provided. The results show that the developed system is scalable under real-world setting.

ACKNOWLEDMENTS

I wish to express my special thanks of gratitude to my supervisor, Dr. Ralph Deters for his advice, patience, support and financial assistance throughout my duration of study. Also, my genuine thanks to the members of my advisory committee: Dr. Julita Vassileva, Dr. Schneider Kevin and Dr. Chen Li for their exceptional contribution and feedback. Furthermore, I would like to thank Ms. Gwen Lancaster, at the Department of Computer Science, University of Saskatchewan, for her support and help throughout my study.

Finally, I would like to thank my mother, Ms. Hannah Nkrumah and my entire family for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST ABBREVATIONS

IoT	Internet of Things
M2M	Machine to Machine
IP	Internet Protocol
RFID	Radio-Frequency Identification
IIOT	Industrial IoT
NFC	Near Field Communication
CoAP	Constrained Application Protocol
MQTT	Message Queue Telemetry
IEFT	Internet Engineering Task Force
UDP	User Datagram Protocol
HTTP	Hypertext Transfer Protocol
ROM	Read-only memory
RAM	Random-access memory
XML	eXtensible Markup Language
JSON	JavaScript Object Notation
REST	Representational State Transfer
URI	Uniform Resource Identifier
TCP	Transport Control Protocol
W3C	World Wide Web Consortium
RDF	Resource Description Framework
SOA	Service-oriented Architecture
API	Application Programmable Interfaces

CHAPTER 1

1. INTRODUCTION

The Internet of Things (IoT) allow for everyday objects or devices (including books, coffee machine, washing machines, buildings, humans) in our environment to be equipped with sensors and actuators so that they can communicate with each other and to the Internet via wireless or wired connection. These everyday objects or devices gather data from the physical environment and then transmit the data over the Internet. The data is processed, analyzed and then insight is drawn for proactive decision making. The insight drawn enable us to reduce cost, change business operations and models, and ultimately make our human live experience simpler and richer. IoT has several real-world use cases that show its limitless possibilities and benefits. An example is the Nest Thermostat device [1] deployed in homes to allow home users to remotely change and modify their room temperature. This device is intelligent because over a period, the smart thermostat can learn the user's temperature preference and then adjust the settings accordingly without the user's assistance. This type of IoT service brings comfort to home owners. Also, in business, IoT is used in manufacturing industries for predictive maintenance. Thus, sensors and cameras are deployed in industries to gather data which are then analyzed in real-time to determine when a part of equipment will fail, so that pre-emptive measures can be taken to avoid such unforeseen events [2]. IoT has become the next evolution of the Internet, since it is allowing us to gather, analyse and share data from which knowledge is being extracted. Hence it is weaving itself into our lives and gaining lots of attention. A report by Gartner [3] indicates that employees can cut down health cost by 40%, by 2020. The report explains that, by wearing fitbit tracker, employees' gathered data can be made available to healthcare providers. Upon analysis of these employees' data, preventive measures can be taken to save their lives. Moreover, the report forecast that by 2020, IoT can reduce the cost of maintenance and consumables by 1 USD trillion a year [3]. In addition, Ericsson [4] projects that 28 million devices will be connected to the Internet by 2021 (see Figure 1-1), which will result in an increase in the compounded annual growth rate (CAGR) of 23% from 2015 to 2021.

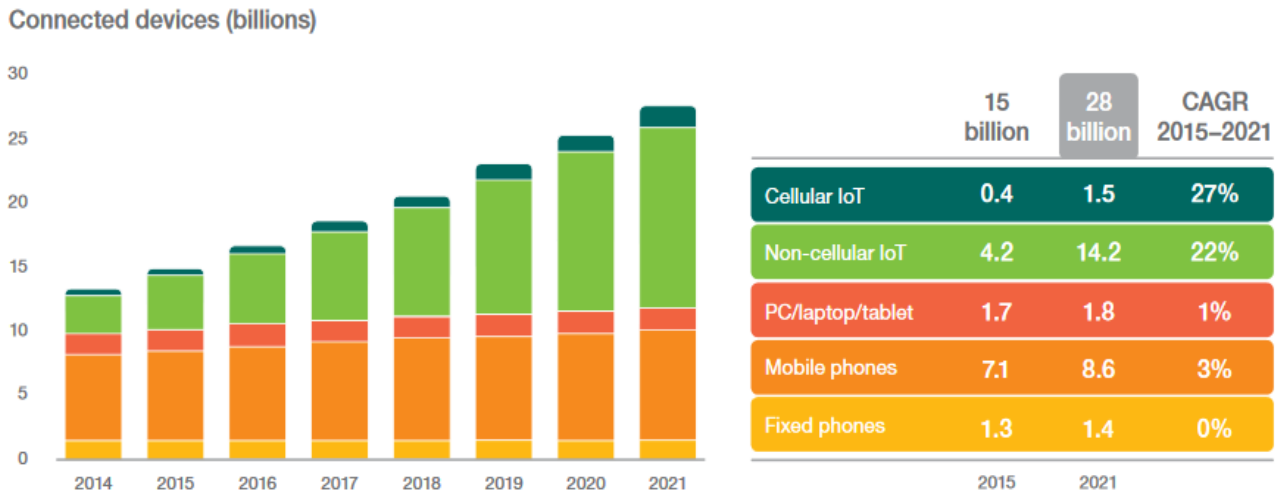

Figure 1-1: Ericsson's projected total connected IoT devices worldwide. [4]

Also Cisco [5] predicts that the total number of devices that will connect to the Internet will surpass the people in the world by a ratio of 3.4:1 up from 2.2 in 2015 to 2020. These numbers translate to a potential market for IoT in all sectors, but at the same time raise several issues when these devices communicate among themselves and over Internet. One of which is provenance tracking during IoT device communication. For example, you or your family member within your smart home can directly turn on/off your smart TV with a mobile device. In this case, it becomes easier to track who turned on/off the smart TV and the server that activated the state and responded with the data. Tracing in this context is easy because it is a direct connection and is also within the confinement of the home network. However, the problem arises when the cloud is introduced, making the connection indirect. Using the same above example, when someone aside your family members remotely turn on/off your smart TV via the cloud. This becomes dangerous and a worry to the home owner; Hence making it difficult to track and to determine from where the person sent request/command from. This is because you do not know who is behind the cloud as there are lots of devices within the cloud communicating with each other. Communication among these devices and amongst people will be prevalent. Hence, the above problem calls for attention and a need for a solution. Being able determine the origins and chronology of requests and responses during device communication is vital to providing a digital audit trail or digital footprint that will help establish transparency, auditability and accountability within the Internet of Things. In previous years, provenance techniques have been employed by many researchers to addresses the issue of data provenance within IoT; that is

where data came from. Provenance, also referred to as lineage is meta-data that determines the chronological history of a data item, from its original state to its final state [6]. Many researchers [7],[8],[9] have applied provenance in different domains for different reasons. For example, Bauer et al. 2013[7] investigated how the concept of provenance can be combined with Internet of Things to ensure trust, reliability and to help build security mechanisms for processing data emanating from Internet of Things. Since then, a large amount of research considering provenance integration in the Internet of Things has gained traction from industries, businesses, researchers and academicians.

Lots of research [10],[11],[12],[13],[14] have been done, with substantive application developed in various sectors ranging from smart cities [15], food supply [55], supply chain system, transportation to mention a few have been investigated. However, gathering extensive research from various sources on provenance in the Internet of Things reveals that most research work conducted so far looks at it from one perspective that is from the request perspective with the user making inference with a mobile device to establish the origin and chronology of the data (that is where did the data came from). This reveals that little or nothing has been done regarding the response perspective. That is enabling servers to determine the origin and chronology of requests (that is where did the request/command come from). On this basis, this thesis seeks to develop a provenance-based blockchain system that will help trace bi-directionally the provenance of requests and responses within the Internet of Things to answer the questions: where did the data come from and where did request/command come from? This system will store an immutable digital history of provenance information with regards to the life cycle of requests and responses within IoT communication for which inferences and decision can be made to establish transparency, auditability, accountability and truth. In addition, this thesis also seeks to evaluate the performance of the blockchain-based provenance system.

The remaining sections of the thesis are structured as follows: Chapter 2 defines the problem, state the research questions and the goals of the research. Chapter 3 explores related works with regards to Internet of Things, provenance application in the Internet of Things, provenance, REST and blockchain. Chapter 4 presents our proposed provenance decentralised system architecture that seeks to address the issue of traceability (tracing provenance of request/responses). The provenance-based blockchain system developed together with its performance analysis and result is described in Chapter 5.

CHAPTER 2

2. PROBLEM DEFINITION

In a resourced-constraint environment, resources including low powered devices, have limited computation ability, memory capacity, and less bandwidth to function adequately compared to high-end computing devices. These resources communicate with each other and to the cloud to send and receive data. However, traceability among these resources especially when the cloud is introduced becomes a problem. If a user sends a request directly to a resource, that becomes easy to track, because it is a direct connection. However, when you have an indirect connection due to the introduction of a cloud, tracing become difficult to achieve. This is because the cloud contains multiple resources within such constraint environment. For example, when a user sends a request to access services/data remotely from a server, it will be worth determining from the server's end the sender of the request and also considering the various paths that it traversed before arriving? Conversely, upon receiving response data from a server by a user, the users will also want to know where the data came from. It is important the data might have been falsified or tampered with whiles in transit via the cloud containing resources. This puts forward a research question within the IoT domain that requires a solution. To lay more emphasis on the problem domain, let us consider a scenario (see Figure 2-1, Figure 2-2), where you live with your family in a smart home. Whiles leaving for work, you turned off your Smart TV. But your brother who also left for school remotely turned on the TV to watch his favorite TV program but forgot to turn it off. After arriving home, you realised that your smart TV is on. You will be worried and would want to know who turned on the smart TV whiles you were away. If your brother connects to the smart TV directly as shown in Figure 2-1, we can easily trace back to him. However, if the connection is indirect as shown in Figure 2-2 it becomes difficult because there could be multiple IoT devices within the cloud/internet. Hence making it very hard to know who is behind the middleware/cloud.

Additionally, data is an important asset and very useful for decision making. Data is commonly stored in a centralized system by companies. In an event where the system crushes or the data is either compromised or deleted: how do we trace or retrieve the data that has been captured in the database to make decisions? Furthermore, data in a central database can be manipulated by

entities involved when they have adequate permission bringing about lack of control and transparency because of possible modification of data.

Moreover, with the introduction of the Internet of Things, the discussion determining where data came from (data provenance) has been a major concern not only for mobile users (consumers) but to industries in the automobile, manufacturing, retail and other sectors as well. It is likely that the occurrence of this problem will impact consumers since privacy to them is important. From the consumer's perspective, they will want to inquire about the data along with their origins and chronology. Likewise, enterprises or businesses including health, automobile, utilities, etc are likely to show concern. This puts forward a research question in the domain of the Internet of Things.

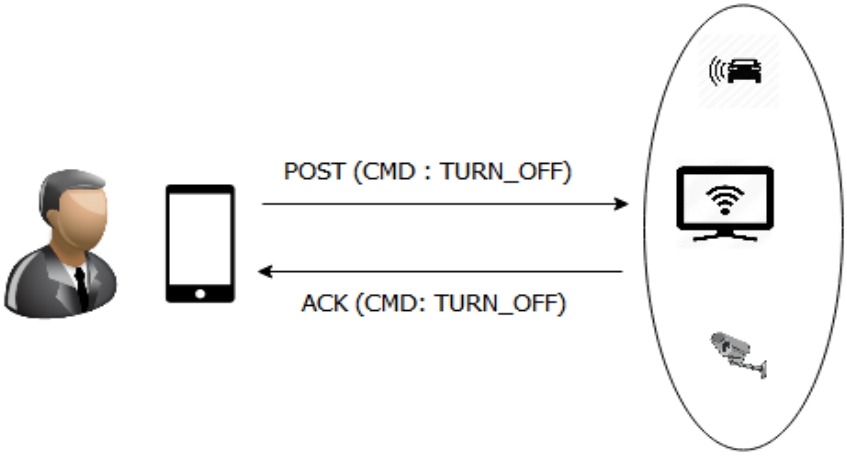

Figure 2-1 : Direct Communication between a user and sensors

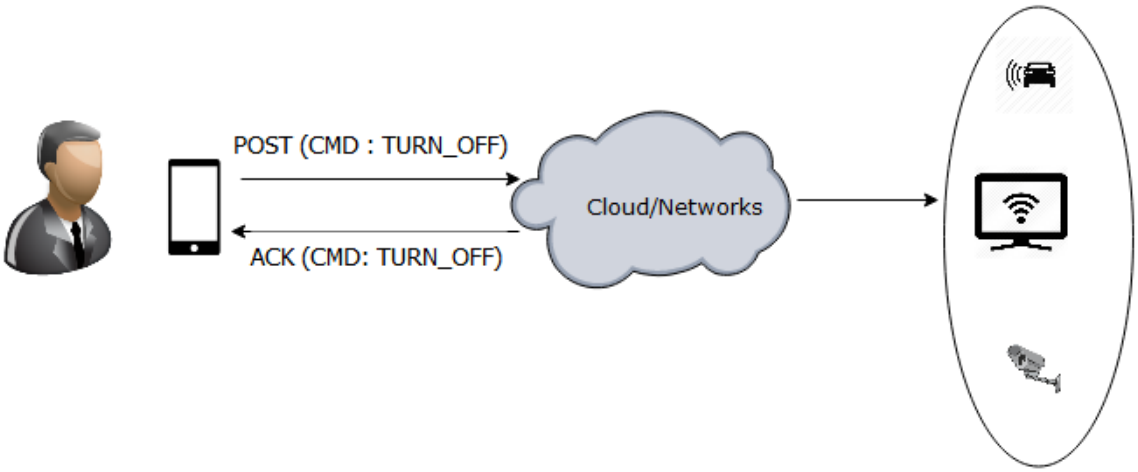

Figure 2-2 : Indirect Communication between a user and sensors

2.1 Research Questions

This problem defined and explained above, puts forward research questions in the domain of the Internet of Things.

The questions to be explored are:

1. How do we trace the provenance of request and responses across multiple networks (IoT) bi-directionally?
2. How do we store the provenance data in a distributed system?
3. What is the performance of the distributed provenance system?

2.2 Research Objectives

The objectives of this research are:

1. To develop a provenance-based distributed system.
2. To measure the performance of the provenance based distributed system with respect to the payload data and in terms of:
    a. Throughput
    b. Response time

In this thesis, the primary research focuses on using a decentralised solution to support provenance to answer the question; where data comes from and where request comes from. This will ensure traceability, leading to transparency, auditability and accountability within the Internet of Things. The reason behind storing the provenance record is to allow devices as well as users to make inferences based on historical records with regards to events/activities occurred within IoT communication. At any time when there is a modification of record, traceability can be ensured bringing about proper auditability. As a result, devices as well as users can be held accountable for their actions. Besides this, using blockchain to support provenance recording brings benefits as it prevents modification due to a hashing algorithm technique it utilizes and the fact that for a provenance record to be stored all the nodes on the network needs to agree other than that the record would not hold nor be stored within the shared database.

CHAPTER 3

3. LITERATURE REVIEW

The main objective of thesis is to use blockchain to support provenance in IoT to trace bi-directionally request and responses within IoT. In other words, develop a blockchain-based provenance system that will support provenance tracking to help answer the questions: where did request come from and from where did response come from. Provenance tracking issues have been raised in previous studies hence this chapter will review some works done in provenance tracking in IoT. A provenance-based system can enable us store provenance information including where, who, time, how; in a decentralised blockchain database. This can help answer our research questions stated above. Because provenance is append-only database and immutable, it is difficult to change a record; hence providing support to establish transparency and accountability within the IoT. From the perspective of both users and servers, having a digital chronological history of the life cycle of requests and responses within the IoT communication would go a long way to establish transparency; thus, enabling users as well as IoT device activities made open so that their actions can be held accountable. Lots of researches have been conducted in the areas of provenance, IoT and how both concepts have been combined for various reasons and to also establish that above mentioned requirements.

This chapter provides a background to our research and provides the foundation needed to understand our research. All the concepts that are closely related to our research are discussed and reviewed. Background to the Internet of Things (IoT), Provenance, REST, CoAP, Resource Description Framework (RDF) and Blockchain are discussed in details. Furthermore, the application of each concept is further discussed. The main focal point of our research which reviews how previous research have integrated provenance within the Internet of Things is analyzed, explained and discussed.

3.1 Internet of Things

Internet of Things is a new paradigm. However the concept of using computers and networks for monitoring and controlling devices has been in existence for years. For example, systems to monitor meters on electrical grid remotely using phone lines were in use commercially by the

late 1970s [18]. In 1990, advancement in wireless technology enabled machine to machine (M2M) as well as industrial solutions to gain popularity allowing equipment to be monitored and operated [19]. M2M relied on networks and industry–specific standards. Thus, lead to a push for the usage of the Internet Protocol (IP) standard to connect devices. This saw the first IP–enabled device (a toaster) capable of turning on and off over the Internet. This was also displayed in 1990 at an Internet conference. Over the past decades, IP addresses have been provided to "things". This enabled the soda machine at Carnegie Mellon University and the coffee pot in the Trojan Room at the University of Cambridge to have IP addresses [20].

The beginning of the research and development into the above technologies laid the foundation for the current IoT paradigm shift. Although, Internet of Things was conceived many decades ago, it was not until 1999 when the term was formally used by Ashton [21], to describe a system where everyday things equipped with sensors connect to the Internet. In his paper [21], he touched on Radio-Frequency Identification (RFID) usage in supply chain management and how it can be extended to other domains. Since then, the term, "Internet of Things", has gained popularity. Thus, IoT is being considered in a broader context. This will enable the Internet connectivity of everyday items with computing capability, e.g. smartphones, sensors, cameras, etc. The term, "Internet of Things", has now gained traction among researchers and industry players, with the likes of Cisco [11], Industrial IoT (IIOT) [22] and The European Commission [23], presenting different definitions focusing on different perspectives, such as connectivity and sensory, need of ubiquitous and autonomous connection of object and identity, and integration of services with regards to objects [24]. One such definition from Cisco, envisions IoT as comprising of people, things, places, which makes services available for other entities to consume [3]. Similarly, Giusto et al. [25] "describes the ubiquitous presence around us of different variety of things including sensors, Radio-Frequency Identification (RFID) tags, actuators, mobile embedded devices, etc. assigned with unique addresses that allows interaction and cooperation among each other to attain a common objective, resulting in a new class of services and applications". Buyya et al. [26], defines Internet of Things, "Interconnection of sensing and actuating devices providing the ability to share information across platforms through a unified framework, developing a common operating picture for enabling innovative applications". This, he explained can be achieved by smooth universal sensing, data analytics and representation of information with Cloud computing as the binding framework. Also, as

reported by [24], Internet of Things refers to “a world-wide network of interconnected objects with unique addressing scheme, relying on standard communication protocols.” Atzori et al. [27] claims that combing ‘‘Internet” and ‘‘Things,” brings about a disruptive level of evolution into today’s ICT world. In addition, [26], explains that a key characteristic of IoT, is smartness and further explains that there are two vital components with respect to IoT, namely, things and the Internet. The things, in this category constitutes a broad set of entities which includes sensors, smart devices, people and other objects aware of their environments and settings, and are also able to interact with each other without unrestricted access with regards to place or time. Similarly, [27], also noted that the internet of things is broken into two terms, of which the first is geared toward a network oriented vision, whereas the second focuses on generic objects. Likewise, Gubbi et al. [28] envision IoT from two perspectives : ‘Internet’ centric and ‘Thing’ centric. He explains that Internet-centric architecture comprise of Internet services whereas object-centric, comprises of intelligent objects and these objects generate the needed data.

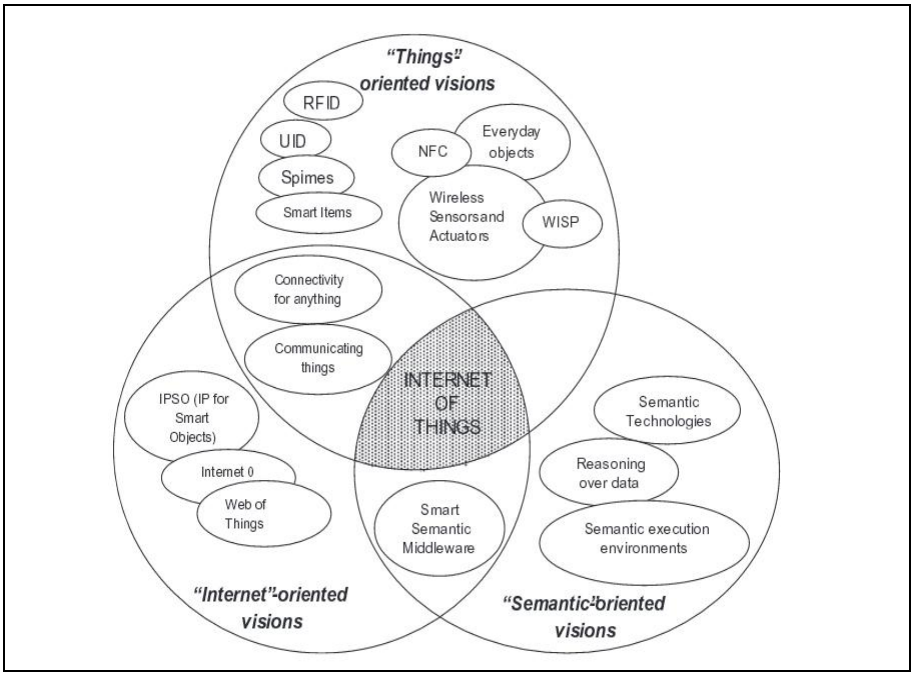

Figure 3-1 : Visions of the ‘‘Internet of Things” paradigm. Source [27]

Atzori et al. [27], took it further by proposing that, Internet of Things can be seen from three perspectives, namely “Things”- oriented visions, “Semantic”-oriented visions and the “Internet”-oriented visions. Atzori et al. [27] perspective of the Internet of Things is shown in Figure 3-1.

As explained in the diagram by [27], the author's combine the three perspectives of the Internet of Things vision and explain that it comprises of the Things oriented vision where Radio Frequency Identifications (RFIDs), as well as components such as Near Field Communication (NFC) and Wireless Sensors and Actuators [29] connect the physical world and the digital world. Beside this, existing global infrastructure and networks will be incorporated into the Things oriented vision. Following, the Things oriented vision is the Internet Oriented Vision which will simplify the Internet Protocol (IP) to allow anything to be connected anywhere. With respect to the semantic oriented vision, the author's state that it is expected that the things in the future will increase. As such information that will be generated by IoT will require insight so as to represent, store, connect, search and organise them. Therefore, semantic technologies will become very crucial in achieving this. Gubbi et al. [28], explain that three components are making up the Internet of Things. They include:

- Hardware: This category includes sensors, actuators, and embedded communication devices.
- Middleware: The middleware includes software's, on-demand and computing resources that provides data analysis.
- Presentation: This component provides tools that aid in visualising, giving insight and understanding of data which can be utilised of different platforms for various applications.

Gubbi et al. [28] further explains that, some enabling technologies under the above components are :

- Radio frequency identification (RFID) - this plays a vital role in Internet of Things. RFID, makes use radio waves which are used to locate and determine items. This will allow us gather relevant information.

- Wireless Sensor Networks: Using small tiny low powered devices and wireless communications will allow for sensor network with intelligence to capture data from different environments, process, analyze and share information easily.

- Addressing Scheme: Things need to be uniquely identified. Not only do we identify billions of devices but remotely control devices via the Internet. Therefore, every object t will connect in future will be uniquely identified in a form.

- Data storage and analytics: Lots of data will be generated; therefore, an important role will be the storage of such data. This enabler forms part of the middleware layer mentioned above. Furthermore, an infrastructure to support the storage of data and analytics using artificial intelligent algorithms, techniques and software applications for decision making will be of immerse important.

- Visualization: Visualisation is an important enabler of the Internet of Things. The reason being that a user will be required to interact with the environment. Due to the availability of devices including smartphones, tablets, iPad, data will be visualised in an easy to read and understand manner.

Similarly, [30] highlights three important enablers comprising of identification, sensing and communication technologies which fall under the explanation provided by [28] on RFID, wireless sensor networks and Data storage and analytics which is also linked to the idea of the middleware explained by [30]. In contrast, [31] highlights some of the factors that enable IoT. These are;

- Human beings - As reported by [31], people can either act as a consumer or a producer of data
- Smart devices – Due to the low cost of manufacturing devices, smart devices have gained popularity. [31]
- Communication networks – Communication among devices is crucial especially when connecting to the internet. Example of such communication networks includes Wi-Fi, GPRS, 3G, Wireless HART, Zigbee, Bluetooth, etc. This establishes a way for devices to talk to each other using standard protocols to allow for connectivity among IoT devices [31].

- Cloud computing- Cloud computing is another important enabler in a sense that they provide support and computing resources which will aid in scalability. This will be needed to support the high demand on storage and computing power required in IoT [31].

There have been lots of applicable domains where Internet of Things have been applied. These includes transportation, logistics, Healthcare, smart environment, personal and social domains. For example in the transportation domain, [32] identifies a problem with regards to the increase of vehicle, communication construction and city traffic (mass transit problem) as a result of rapid urbanisation. To resolve the issue, [32], develops an intelligent transportation system based on the Internet of Things. In the system, data is captured via vehicle terminals and then sent to the server with the aid of the network. This data is made available to consumers using an algorithm that runs on the server. Additionally, the system allows the consumer to query about public transit vehicle information via the web. On the other hand, the phone can be provided with public transit vehicle information through the station terminals. To test the efficacy of the system, an experiment was performed by the authors. They argue that their system can improve traffic resource utilization ratio, making travel much more convenient.

Also, with the Internet, connectivity among devices and support for communication protocols are established to enable interaction and for mining and analysis on data so as to elicit knowledge or gain insight. The things are low resource-constrained devices in terms of computation, memory and network capabilities. They are intelligent, enabling them to send/receive data, change to the dynamic environment, self-maintain, self-configure and self-repair to without human intervention [33][31].

Furthermore, due to the popularity of the Internet of Things, the Business Insider [34] predicts that 34 billion devices are expected to be connected to the internet by 2020, which translates to an increase of 10 billion from 2015. Quoting from the same research, it is expected that IoT devices will account for 24 billion, whereas computing devices such as smartphones, tablets, smart watches, etc. will cover 10 billion. Again, with the likes of major IoT players such as Google, IBM sharing information due to the potential market value, IoT is expected to grow. Additionally, collaborative and cooperative efforts by the likes of Cisco, Google, IBM, ARM, to mention a few will likely increase the adoption rate of IoT and increase the market space. To support investment and research in IoT, effort towards projects are being started by IoT

European Research Cluster (IERC), to elicit user and application requirement of which some of these projects are being used in developing a reference architecture meant for specific application types in IoT. [26]. As reported by [35], a 5 million support towards IoT innovation and research has been initiated, by the UK government as a starting point, whereas it is hinted that IBM [36], also intends to support IoT research and its industry applications by investing in billions of dollars. This translates to a potential benefit that IoT will offer in the future to both consumers and businesses alike. The potentials that IoT bring is exciting and spans across multiple business application domains including efficient traffic control systems, energy-efficient transportation systems, energy efficient management, smart health systems to efficient global delivery of systems and promises to improve quality and reduce cost drastically to its minimum level [37] [23].

3.1.1 IoT Protocols

IoT devices need to communicate with each other. They do so to exchange data among themselves and then send the data to the cloud for further analysis. However, for the communication to be established, they need to have a common language, called protocols. There are lots of IoT communication protocols, but some are not better suited for IoT applications. This is as a result of the extra header added to each request that is sent which then leads to an overhead cost with respect to resource consumptions such as battery power and network bandwidth. Thus, IoT protocols have been designed specifically for IoT devices and its applications. These protocols are lightweight and are targeted at devices that are resource-constrained in terms of battery power, network bandwidth, memory and computational capacity. Though, there are many IoT protocols, the two commonly used for IoT communication protocols are the Constrained Application Protocol (CoAP) and Message Queue Telemetry (MQTT). Overview of the IoT protocols is described in the next section.

Constrained Application Protocol - CoAP was proposed in June 2014, by Universitaet Bremen TZI and ARM. The protocol was created to enable communication among devices with limited resources with respect to computation, bandwidth, and energy. CoAP is a RESTful protocol, a standard protocol developed by the Internet Engineering Task Force (IEFT). CoAP runs on UDP but extends the features of HTTP and is based on a request/response model. Additionally, CoAP comes with a maximum packet size ranging from 4 to 1024 bytes [38] [39]. As shown in Figure

3-2, it has a compact message encoded in a binary format which is composed of the version, type, messageid, code, etc. in the header. Following the header, token and options is an optionally used payload [40].

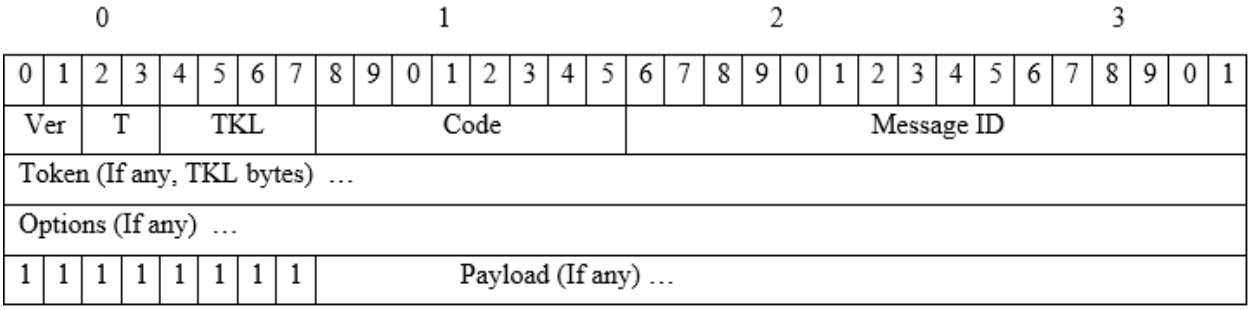

Figure 3-2 : CoAP Message Format as adapted from [39]

Additionally, the nodes have an inbuilt microcontroller of 8 bits as well as a small amount of ROM and RAM with constrained network providing a throughput of 10s of kb/s. Like HTTP, CoAP supports web URIs and provides built-in support for discovery of resource and allow for different data formats such as XML, JSON, plain text, html etc. The protocol is broken down into two-layers namely: Message layer and Request/Response layer. This is shown in Figure 3-3 [39]. The message layer handles asynchronous interactions whereas the request/response layer handles REST communication, making use of methods as well as Response Codes. CoAP has four types of messages: confirmable, non-confirmable, reset, acknowledgement. The types of messages are briefly explained.

Confirmable message – A confirmable message needs an acknowledgement. A client will continuously send a CON message till it gets an acknowledgement with the same message ID.

Non- Confirmable message – for this type of message there is no acknowledgement. An example is when a client sends a request with a GET method.

Acknowledgement message – An acknowledge message indicates that a confirmable message sent has been received.

Reset message - A reset message confirms that either a confirmable of Non-Confirmable message was received. However, due to systems or some other factors, the message couldn't be processed.

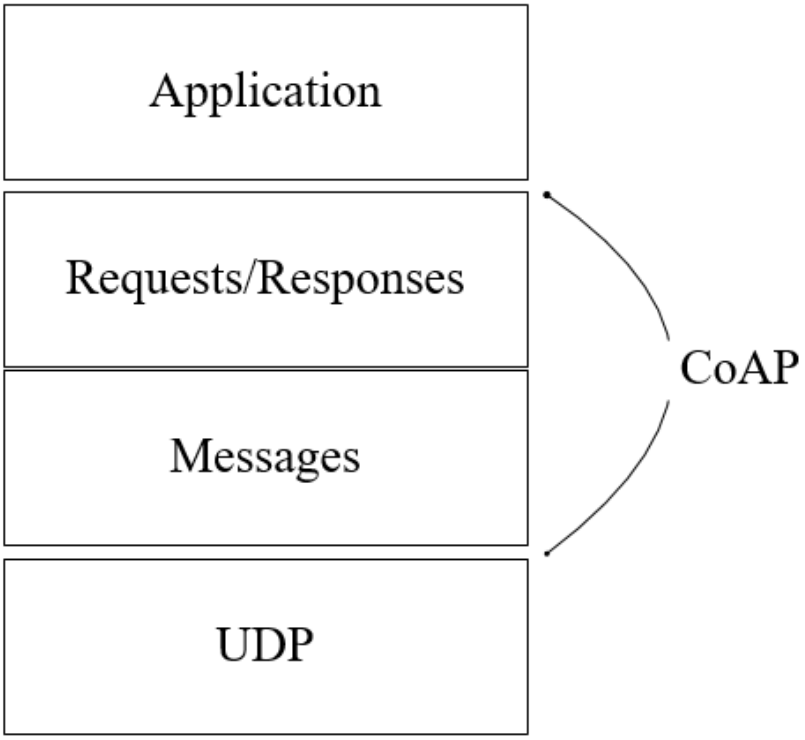

Figure 3-3 : Abstract Layer of CoAP as reproduced [39]

With respect to CoAP's request and response model of communication, clients which includes smart phones, laptops, and tablets send requests to servers (high end servers) whereas servers provide the client back with data as response. Methods such as GET, PUT, POST and DELETE are used by the client with the aim of retrieving resources with unique identities such as Uniform Resource Locator (URI) [40]. The protocol follows the principle of RESTful design and can interoperate with HTTP allowing integration with the web. Additionally, the protocol continuously maintains support for multicast, reduce overhead and at the same time provide ease for constrained environments. Some features that CoAP comes with are:

- Low header overhead and parsing complexity
- Asynchronous message exchanges.
- URI and content-type support.
- simple proxy and caching capabilities

In recent years, CoAP's application has been seen in the Transport Logistics sectors. For example, [41] highlights the use of CoAP to retrieve sensor data during land or sea transportation in machine (M2M) communication for logistic applications. In the paper, [41], a solution, called

libcoap was deployed and tested on two widely known embedded operating system such as TinyOS and Contiki. The metric considered were: Response Time, Total number of bytes and overhead of header. Based on their comparison between CoAP and HTTP with regards to resource retrieval within a constrained cargo environment containing embedded devices, they assert that CoAP performs better. In the context of our research, we choose CoAP as the communication protocol due to the fact that it is specifically geared towards IoT devices and also due to the features it comes with. The CoAP payload will be considered during our implementation.

Message Queue Telemetry (MQTT) - MQTT is a lightweight publish/subscribe message protocol designed by IBM and then standardized by OASIS [42]. It is aimed at connecting small devices to constrained networks. It runs on Transport Control Protocol (TCP) and is not designed to broadcast messages to receivers. MQTT uses the most favorable connection operation for its routing mechanism using either a (one-one, one-to-many, many-to-many). Furthermore, as a way of delivering its messages, MQTT makes use of three levels of quality of service explained below.

At most once: Messages are delivered at most once or not all. This means that message delivery is not acknowledged.

At least once: This type of quality service ensures that a message will be delivered at least once. However, it is likely that the message can be sent several times.

Exactly once- This is the safest way for message delivery. However, it takes a longer time for message delivery and acknowledgement between that sender and the receiver. This service ensures that messages are received only once by the sender.

MQTT, has three components namely, publisher, broker and the subscriber. Publishers publish their data to the broker. The broker serves as an intermediary between the publishers and the subscribers. The broker records all the topics that are published by the publishers. Subscribers subscribe to topics they are interested in. And as soon as there is an update of topics, the subscriber is notified based on its topic interest subscribed through the broker. Several MQTT uses in areas such as health monitoring, energy monitoring, environment sensing, Facebook notification makes it a good protocol fit for IoT and Machine to Machine (M2M) communication [43].

3.3 Provenance

Provenance, which is derived from a French word provenir, meaning “to originate” [44], is a well-known concept with respect to a piece of artifact in the field of arts, archeology and archives. The concept became very important in arts, because lots of people wanted to prevent forgery and to know the original source before purchasing an art work [6]. Recording the history of ownership, including owners, dates, transference, locations as well as time with respect to an art work is the provenance of the art. For example, [45] asserts that a piece of work sold in auction comes along with its chronological history from the time it was created to the time it is auction[45]. When this important information is captured, it validates and establishes the authenticity of the art work. Thus, increases its value [46]. Additionally, it lays the ground for assessing a digital object’s quality, reliability and its trustworthiness. For instance, it is applied to social web, science, and in the business domains as well [28]. Groth [47], states that the three things that characterizes provenance in the field of arts are:

- documenting precisely
- past, (where it was)
- process tracking

The revolution of the web wide web technology promises benefits but comes with risk due to the amass nature of digital information. It is therefore important that the provenance of this digital information needs to be recorded. With the introduction of provenance, many of definitions have been presented by [6]. Provenance is not limited to data or a piece of art but it goes beyond. Additionally, the term is dependant on the context of the question, and for the intention of it’s use. Provenance is a well-known area, and it is described by Cheney et al. [48] to play a key role in up and coming digital infrastructure. This means that provenance will allow for integrity, authenticity and repeatability when it comes to digital information. These features of provenance will allow users or devices to analyze and identify errors; prevent unwanted behavior as well as failure which will translate to an increase in transparency and accountability. As explained by [48], provenance solves problems in a scenario where one requires an understanding of how a system transformed a set of inputs to the outputs.

The basis of the description of provenance is applicable within a context and or within domain. For example, provenance, explained by Simmhan et al. [49], in the context of data, is as an audit

trail, (lineage or pedigree) of a data item including its recorded information and its associated processes. Similarly, [48], explains provenance from a data perspective. As defined by [48], provenance is information describing the context, origin or history of data. Also, Moreau [8], made a comparative analysis of the definition of provenance and viewed provenance with a focus on data as a concept, in computer systems. Moreover, [8], perceived provenance as a process and explains that "the provenance of a piece of data is the process that led to the piece of that data". W3C, World Wide Web Consortium [50] expands on this definition, by providing a recommended definition which describes provenance as "a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing". Provenance has over the years gained relevance in various sectors and in different use case scenarios and it's being applied to various domains for many reasons. For example, Qiannan et al. [51], claims that the idea behind tracing provenance is to know the context and to establish evidence which leads to its source of discovery, considering its lineage from owners and storage locations. Furthermore, Buneman et al. [49], states that, provenance is vital if one needs to understand data quality. Also, Buneman et al.[52], in the field of database and in the context of data provenance, points out that, database users are interested in understanding the provenance of a given piece of data. They explain that most data in databases are subsets (views) and curated, as such only few are source data. Therefore, understanding provenance determines that the data's accuracy and timeliness. They proposed a general framework in the context of scientific databases which is also applicable to other relational databases. The proposed framework provides an understanding of data provenance. Furthermore, they categorized data provenance into two aspect; namely why provenance (what influenced the existence of the data) and where provenance (the actual location where the data came from). Also, in the context of health domain, Álvarez et al.[53], applied provenance in a distributed medical system by proposing a provenance application based on service-oriented architecture. Their application made use of example which focused on human organ transplant management. As reported by [53], the captured provenance is used to trace the actors involved in the medical processes including data collected and equipment used. According to them, the provenance record was used in making decisions in a Distributed Health Care System scenario and in several institutions. Also, in Europe provenance has become vital to ensure the quality of food. For example provenance was used as a way of backtracing through the process involved in the food

supply chain [54]. In an event when there is a problem or an issue that occurred along the way, provenance becomes crucial to support the auditing of such processes involved among different parties [45]. Buneman et al.[55] Studied and analyzed various methods with respect to provenance. Based on their study, they put forward a characterization of provenance systems as part of the proposed solution. They explain that *why* indicates the reason for capturing provenance, "*what"* describes the processes and "*how",* describes the way provenance is represented, stored as well as how it is disseminated.

3.3.1 Application of Provenance in the Internet of Things (IoT)

In this section, the application scenarios of provenance in the Internet of Things in various domains are discussed and reviewed.

Extensive research on provenance has been done. Due to the importance of provenance its application have span multiple domains including databases [56], Cloud Computing [9], Scientific Workflows [57], Grid and Distributed computing [58]. In previous research, lots of emphasis has been giving to provenance, especially in the context of data and with respect to tracing the source of data. The importance of provenance in previous research has been linked to establishing authenticity, trust and quality, focusing on data. Tonjes, [15], asserts that there is wide spread research conducted in data provenance in the service area; however very less attention has been given to provenance in the Internet of Things. Although, studies have been done and research efforts applied in different domains and in different context, more research is being conducted especially in the Internet of Things which over the few years has gained traction. Indeed, for these recent years and considering the importance provenance, substantive amount of research is being conducted in applying the concept to the Internet of Things[16], [17] [51], [7],[59],[16], [17]. For example, Bauer et al. [7] applies data provenance in the Internet of Things. Bauer et al. [7] extends on Buneman et al [52] approach on why and where, and states that although provenance information such as why, where and when have been captured, an extension to encompass who, timestamp and time periods of processes on data is needed. In the research, they considered provenance, but from a data perspective. They further explain that objects with embedded sensors and actuators in and around us will communicate with each other. The authors' assert that the communication among the objects generates an enormous amount of

data who's processing require some degree of trust [7]. Additionally, they assert that requirements with regards to Integrity and Confidentiality will be eminent. To overcome the afore-mentioned issue, the authors propose a conceptual architectural model which establishes connection points between the two architecture models of IoT [60] and Data Provenance [61][62] respectively. They noted that the Information Model of the IoT infrastructure model serves as the infrastructural interface providing the integration of data provenance, capturing provenance information as data model of entities, agents, and activities. Furthermore, in combining the two concepts the IoT architecture is maintained with its component and is used as a common basis, while extending it to with a provenance Event Handling component. They explain that the provenance Event Handling component utilizes algorithms that handle the resulting information and controls the actions of the sensors and actuators. They explain that, the algorithm comprises of collection – (which collects and determines the data format of the needed provenance information), verification (which captures needed information and ensures that the operations performed on the data are seen), categorization (classification of information collection) and selection (ensures that needed sample of provenance information are selected). Additionally, they assert that, collection of information to retrieving phase needs to follow the requirement for provenance (Integrity, Availability, Confidentiality, Efficiency, Privacy, Likability and Un-linkability). Following the description of the common architecture, they explain that the web server retrieves collected information and connects to a database which is composed of two parts, namely device information, and Provenance information. Furthermore, the authors assert that an interface on the other hand enables provenance information to be visualized. At this phase, they claim that necessary settings and manipulation with regards to provenance information is allowed. The Browser Interface links with an Access Control that manages rights, which is connected to a Database and the Management Interface of the backup component to maintain privacy and confidentiality. The authors pointed out there were challenges during the combination of the two points however; their key findings show that introducing the Provenance Event handling establishes control for Link-ability and Un-linkability. That is the component which is based on the underlying algorithms ensures that information is collected and that it also establishes secure handling of the information.

In conclusion, the authors implemented a provenance Event handling component by linking data provenance and IoT connection points. They also state that more storage capability is still a

necessity and in addition, authentication mechanisms for security and trustworthiness on captured information and accessing users' needs to be integrated into the common architecture.

Similarly, Eduardo et al. [63] proposes a lightweight semantic model and a prototype mobile-enabled software to record, store and utilize metadata information (device, provenance, device capabilities and their usage). As reported by them, although IoT applications running on devices are beneficial, users are unable to understand the devices, how they are used as well as their capabilities. To lay more emphasis on the above, questions such as; "What kind of data does the thing collect? Is the data transmitted? How and to whom? For what purposes are the data used? What control do I have over any aspects related to the generation and use of this data?" They assert that, recording such information provides a capability of user inference to be made thus making IoT devices transparent. In the paper, two case studies regarding The Trusted Tiny Things project were investigated. The first case saw Aberdeenshire council implement a passive Near Field Communication tag which access time table information about bus stop utilizing a smartphone whereas the second case study, examines in-car black boxes which tracks and captures information with respect to diving behavior, vehicle location using a range of sensors to transmit this information to an insurance company. Key questions relating to user privacy such as "*what kind of data is being recorded? When and where is the data being sent? Who is using the data? Is the data being sent to other third party companies? For what purpose?"* These issues were raised in the second scenario. The authors explain that with the use of provenance in both scenarios it is possible for users to understand the lifecycle of the data and the purpose for which it will be used. A demonstrator application based on the two case studies was used in their approach for evaluation. After evaluation, their key findings show that, upon recording sensor data it is possible to reason about personal information, and moreover, connecting with sensor observations to data sources could enable inference of useful information on users performed actions.

Similarly, Kolozali et al. [64] proposed a stream tagging framework for real-time IoT data as a way of supporting dynamic incorporation into the web space. They suggested adding a meta-data to IoT data stream and explained that the framework has four main parts namely; virtualization, middleware, reliable processing and semantic labeling. Virtualization handles access to diverse data sources, middleware deals with the communication by using an Advanced Message Queue Protocol. In addition, the authors' state that a reliable information module handles the extraction

of dynamic and diverse sources of data and performs processing and aggregation but considers accuracy and trust. Another lightweight information model used for providing a summary and a reliable IoT data stream was proposed. The authors argued that this information model included stream annotation ontology, quality of service as well as quality of Information and provenance. To evaluate the performance of the framework, different data stream, raw and amassed were tested against their annotated data. The key findings of their research showed that the framework performance in all cases recorded increase in 99.4% and 96.2% with data size and average message exchange time used as performance metrics. Furthermore, they suggested adding a wide set of data streams and making use of computer network which would enable the capturing of real-time road traffic to perform analysis on large number of users. Additionally, they suggested further work on stream annotation ontology that will provide effective coverage of stream analysis techniques mostly used by researchers and to generalize the model to blend them with research tools in existence. In contrast to the above methods, Jie et al. [59] identifies resource scheduling issues in distributed and parallel computing environment and proposes a scheduling algorithm based on provenance for logistic chain in IoT. This provenance based algorithm takes into consideration user's appraisal and assigns jobs to effective providers with good quality of service.

To test the effectiveness of this provenance based algorithm a simulation was performed with three test cases which provided excellent results. They stated that, ongoing work comprises an information service that will provide detailed information on jobs and providers' data, a workflow engine capable of managing jobs assigned to several providers and a tool to establish reliability and coherence of diverse user feedback. They further touched on the need for efficient management and storage capabilities of provenance information which was point out by Bauer et al. [6].

Moreover, Tonjes et al. [15], asserts that IoT and smart city applications mostly concentrate on communication, connectivity and collecting data in order to make analysis. They further went on to explain that usually, the focus is centered on collecting and storing dataset, of which providing high performance computing and data mining are the major priority. Tonjes et al. [15] explains that, although most research has been conducted to provide innovative methods for smart applications, there is still an issue regarding providing methods that can scale and are efficient in providing real-time processing, inference of continuous sensor and social media data coming

from smart city settings. Because of the issue raised above, they proposed a smart city framework that allows large-scale IoT data streams to be processed, by tagging streams of data with meaning, providing the data to be processed dynamically, combined and merged. Again, the authors raised some challenges faced during the collection of data from the environment. Some of the challenges regarding data pointed out were multi-modal, quality, trust and reputation. Moreover, the authors explained that with respect to establishing a reliable information processing, testing and monitoring, issues such as data quality and provenance will be required to play a significant role in smart city scenarios. Thus, the authors identified some methods that smart city framework should integrate [15]:

- Accurancy and trustworthiness and ensure provenance with respect to IoT streams
- Resolution of issues when information contradicts
- Should be able monitor and test constantly so that changes to QoI and trustworthiness can be done regularly.

To resolve the problem with respect to efficiency, the proposed framework mentioned utilized three functional components namely: (i) Large-Scale IoT Stream Processing (ii) Reliable Information Processing (iii) Real-Time IoT Intelligence. With focus on achieving reliable data in the context of their paper, city framework utilises a component under the Reliable Information Processing function called QoI Datastore and Reputation Systems. The authors claim that, this component uses a method to rate information accuracy, trustworthiness and QoI. Furthermore, the Reputation System can assess technical reliability as well as provenance based trustworthiness of the data streams. This component as pointed out is used during the IoT data processing of streams to capture information and establish reliability of data in a controlled-iterative form.

Also, Qiannan et al. 2013 [51], similarly propose a smart sensor data collection strategy with algorithms to identify and trace infected food in IoT food supply chain. They explain that Self-Adaptive Dynamic Partition Sampling (SDPS) Strategy provides an efficient and intelligent way of collecting and managing data emanating from sensors. Infected sources are identified and potentially infected food in the IoT food supply chain are eliminated with the aid of their proposed tracing and backtracking algorithms as discussed in their work. To evaluate the propose system, Qiannan et al, 2013[51], uses a simulation which shows that SDPS could trace with an accuracy of 97.8% using small average of sample percentage relative to the traditional sampling

methods. Most works including the above related works focus on data and user perspective (response), where end users read and infer about data. However, little or nothing has been done regarding requests specifically on the part of the respondents.

Also, Yin et al, [16], raises an issue of food security in domestic situations and explain that due to such incidence the public have lost confidence in the kind of food that is supplied. The authors proposed and developed a system that uses IoT technologies in the life cycle of vegetable supply chain and combines the concept of provenance to record provenance information of the vegetables that is supplied to Hong Kong to ensure the security of food. In the proposed system, RFID Electronic vehicle cards and RFID tags is used to vehicles and vegetables supplied to Hong Kong and provenance information with respect to the whole vegetable supply chain is captured. They conclude that, the system developed is advanced in relation to other existing supply chain information systems and further assert that the system achieved good social benefits and that it is commended by all stakeholders.

3.4 REST

Roy fielding, in the year 2000, introduced Representational State Transfer (REST)[65]. As part of his thesis, Roy explains that World Wide Web, although seen as the world's largest distributed application it is necessary to understand the core architectural principles behind the Web. Furthermore, he asserts that if known will greatly translate to improving other distributed applications to avoid undesirable changes with respect to standards based on the web architecture. He introduces REST, a significant model or framework which is the underlying principle behind the modern Web's software architecture and describes that it can be used in software engineering principles as a guideline in designing and evaluating a real software system. The web represents a loosely coupled application framework, where resources are very crucial to the architecture. Resources are abstracted are made available on a server. These resources are uniquely identified and accessed via Uniform Resource Identifies (URI's) by clients including smart phones, tablets, laptops etc. The resources are accessed in a request/response model using methods like GET, PUT, POST and DELETE [66].

Moreover, as asserted by [65], system performance is usually reliant on the communication network when it comes to network-based applications. With respect to distributed hypermedia system, he asserts than the focus being on computation-intensive tasks, large data is rather sent

among components during their communication. Thus, brings in the idea of REST, to resolve these problems identified to get the needed functional, performance, and social properties required of an architecture [65].

3.4.1 REST Principles

Pautasso et al. [67], explains that REST relies of four key principles. These principles are listed and briefly explained.

- Resource identification
- Uniform Interface
- Self-descriptive messages.
- Stateful interactions

Resource identification - Abstract entities represented as resources should be identified with a uniform Resource identifier. These URI's are unique as such clients requesting for resources should interact with these resources using their URI's. for instance, a resource for a book can be assessed via http://book/1.

Uniform Interface - In interacting with resources, there are a four set of allowable operations that can be used. The set of allowable operations on resources are create, read, update and delete. These are represented with methods such as PUT for creating a new resource, GET, for retrieving a resource in any representation, DELETE, for deleting a resource and POST for updating a resource [67].

Self-descriptive messages - Representations are dissociated from resources therefore content can be retrieved via multiple of formats such as HTML, plain text, etc. In addition, attributes representing data about that resource, known as meta-data is used for content negotiation, error transmission and caching management and authentication[67].

Stateless - Statelessness is an important constraint in that, an HTTP request made is completely separated or are not dependent on each other and then the server does not rely on the previous request that have been made earlier to fulfill its request. If an HTTP request is made by a client, all is sent along information required by the server to fulfill that request. The previous request is not used as the basis for responding to the initial request. The client always needs to make the request repeatedly even if the information that it wanted to send was very important [66][67].

3.4.1 RESTful Web service

RESTful Web services are developed to work on the web. They are web services that follow the REST principles and thus derive benefits in terms of performance, scalability, and flexibility. Relying on these principles allow web services to communicate via Hypertext Transfer Protocol (HTTP) and integrate seamlessly with other services, making application development simple, lightweight and very fast [68]. Due to its flexibility and simplicity, Restful web service development has been encouraged in developing lots of systems. This is due to the popularity of REST discussed above which allows application developers to follow a set of guidelines whiles developing applications. Zhang et al [69] assert that RESTful web service is simple, lightweight, and is able to send data directly via HTTP as such it has been ideal choice when building services centered services. RESTful services follow the principles of REST. For instance,[63], uses RESTful web services as part of it technology to build a system called the Trusted Tiny Project, which included a mobile application. This mobile application was built to retrieve information about IoT devices to make informed decisions. Also, [70] explained that there a ton of vender-specific applications for health-monitoring. Thus, making such applications closed and proprietary, since they are built on the vendors' own infrastructure to collect and keep data from users. Moreover, they argue that the same functionalities are performed using the same approaches which results in inefficiency. Thus, in an E-health Oriented IoT application, [70], implements a common cloud-based infrastructure (open model) using Restful Web services, as a way of allowing different applications to be built and utilised by different health service providers. Using this Restful cloud-based common infrastructure, per their discussion, provide a market opportunity to different stakeholders. Guinard et al [71], similarly, raises concern with respect to issues regarding integration of physical objects with enterprise systems. They assert that most developed applications in the past have focused primarily on mashup architectures to enable flexibility with regards to composition of software within enterprises, but failed to address issues and requirements with respect to larger scale integration. Thus, [71], contributes to the application layer, by proposing and implementing an architecture, which allows to sensor nodes to accessed using the REST principles.

3.5 Ontologies

As defined by Gruber [72], an ontology “is a formal explicit specification of a shared conceptualization”. In other words, it provides a means or a model for which concepts within a domain are organized, grouped and how these concepts relate with each other. For a specific domain, things or concepts are represented as classes, which have properties and relationships. Based on this concept, meaning can be extracted through the class and subclass hierarchy along with properties, relationships, and restrictions that exist between and among them. The components of Ontologies that can be used to form a data model are [72]:

Classes - Classes represent concepts (things) that need to be described in a specific domain. The concept can either be concrete or abstract entity. Classes can have a relation to other classes. They can be described through their attributes. Classes are the focal point in ontology. For example: temperature sensor is a subclass of sensor [72].

Instances - Instances form the basis of ontology. Instances describe individual members that become a member of a class in ontology. For example, a temperature sensor is an instance of a sensor [72].

Property - The property describes the features or attributes that the class/concept has: and are commonly described using key-value pairs. For example, a sensor can have an identifier such as a URI. [72]

Restrictions - For relationship as well as attributes a set of restrictions can be placed or set to determine the set of allowable values. For example, a sensor controller (server) can store multiple sensor data (1: M). [72]

Ontologies can provide a common understanding of topics for which humans as well as applications can use for communication. Not only humans, but computers need to process and interpret data in a meaningful way so that knowledge can be shared independent of resources (applications). In this way, things are described for specific domains based on common standards of understanding. That is, agreement to that concepts or things should be described in this way. This role makes ontologies very crucial. Due to it’s importance, its usage have been seen in areas such as e-commerce, search, engines, scientific domains. Furthermore, Noy et al. [73] point out that ontology can generally be used for the following reasons:

- To provide a common way for which information can be structured and shared among people or software agents
- To provide knowledge about specific domains which can reused
- To provide clarity with regards to domain concepts or terms
- To make inference for concepts and their relations among each other, thereby extracting knowledge based specific domain knowledge analysis.

3.5.1 Resource Description Framework

There are different types of syntax that can be used to describe or model ontologies. Some of these are Web Ontology Language and Resource Description Framework. However, in this thesis we focus on using the Resource Description Framework for modelling our provenance data that will be derived during the bidirectional communication in system. Resource Description Framework is a syntax that can be used for modeling vocabulary and semantics in data models. Information primarily on the web is described for human consumption but describing data for human use is very challenging since it needs to be correlated, aggregated and interpreted. Data here refers to raw facts such as numbers, symbols, etc. and the combination of meta-data. Metadata enables discovery and access to information. For applications to make better use of meta-data and to share and reuse among themselves certain rules must be adhered to [74]. Again, documents found on the web are made to point to each other through hyperlinks but considering such web-resources as data in the form of database and spreadsheet they are unable to point to each other via links. Also, they don't provide descriptive meaning to humans. To provide an infrastructure that enables machines to represent and harness data the resource description framework was proposed and developed under the auspices of the World Wide Web Consortium (W3C). W3C's describes the Resource Description Framework as a framework for representing data on the web. It is an infrastructure for encoding, exchanging and making use of structured data also called meta-data. It lays the ground for meta-data processing [75]. In RDF, statements called triples describe resources and are represented as subject, predicate and object. The subject and objects represent real world things such as books, people, etc. or could be abstract in nature. These resources have unique identifier's which called Uniform Resource Identifier. The resources have values which are atomic in nature. The value can hold values literals such as

numbers, text and can also hold resources which in turn can hold properties of their own. Again, the resources are associated with properties which establishes relationships between resources [75]. To illustrate the conceptual framework of RDF and how data is modelled, a diagram is shown below in Figure 3-4.

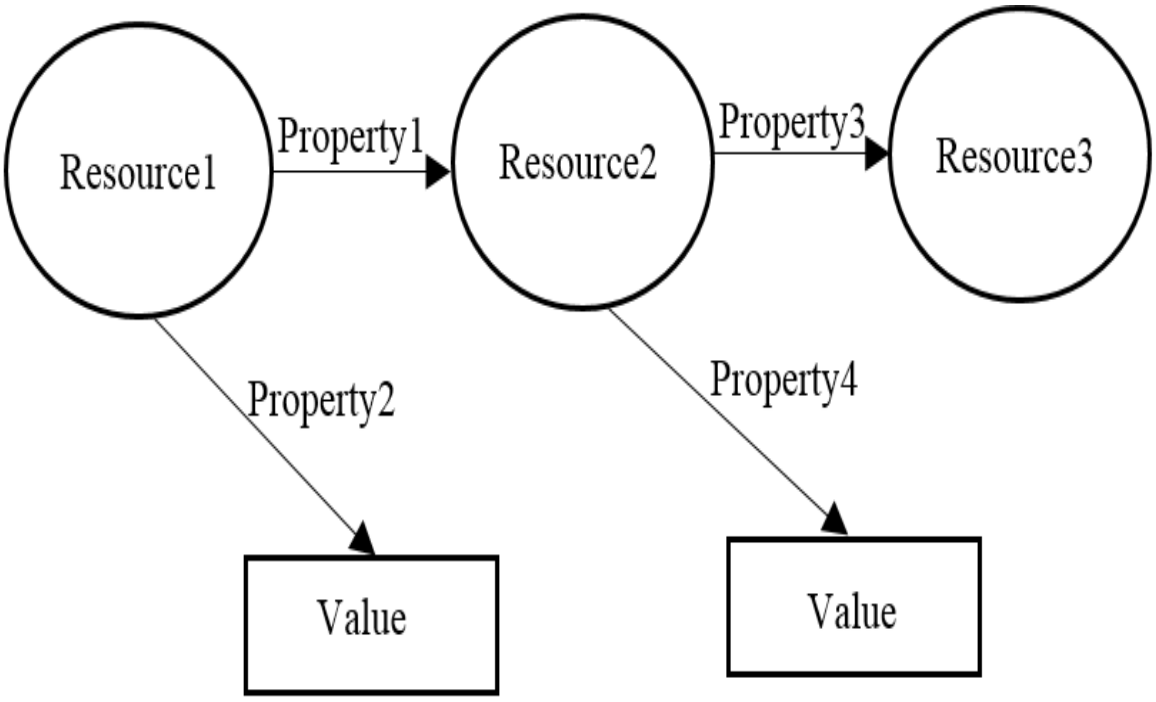

Figure 3-4 : Generic representation of RDF [74]

3.6 Blockchain

Blockchain is an essential component of our research as such it is crucial to present an overview and to discuss its importance and how the Blockchain technology has been used in various domains. The main reason for choosing blockchain in this thesis is because it is a good fit and support provenance by linking records in a linear chronological form which is very important to answering our research questions. In addition, it also provides core features such as transparency and immutability of which this thesis tries to establish as well.

The innovation behind blockchain was first conceived in 2008, by an anonymous scientist called Nakamoto Satoshi. In Satoshi published paper [76], he proposes a novel crypto currency based

on a complex mathematical formula and a robust distributed architecture. As a creator of the well-known bitcoin technology, which is a “purely peer-to-peer version of electronic cash”, he describes in his paper how the bitcoin allows online payments to be done between two willing entities without requiring a third-party, in this case, a financial institution such as the bank.

A blockchain is a distributed and a decentralised ledger that stores all transactions in the form of blocks with timestamps. In other words, blockchain enables nodes which necessarily do not know each other to conduct transaction in a verifiable form using cryptography, without the need for a central authority [77],[78].

3.6.1 How Blockchain Works

The underlying working principle of blockchain is shown in Figure 3-5. A blockchain begins with the first block known as the genesis which does not have a parent. All participating nodes within the network contain the genesis. A new block is then verified and then added to the block. Blocks are added in the blockchain in a linear fashion linking to the previous block. A block contains a list of records which may represent a state change of a transaction in the blockchain. Transactions are stored in the blockchain upon verification by all nodes in the network. Each block in the chain contains a list of transactions and a hash value. The signed hash value refers to the previous hash value in the previous block. The hash value is used as a means of preventing data from being modified. Furthermore, nodes can perform transaction using a public key paired to a private key. However, the public key is used as a unique address for identifying the owner of the account. The private key enables the owner to digitally sign their own transactions. The block chain network selects a node to create the next block in the chain by giving that privilege to the node that solves problems that required computational power. If a node solves the problem, it nominates the next block and broadcast it, which is then agreed upon and verified by participating nodes in the network. The node that wins is then rewarded and this process is what is called mining [77]. Aside its ability to maintain transaction security, blockchain also allows for a distributed consensus on the state of the database ensuring that transactions occurs once or nothing happened entirely.

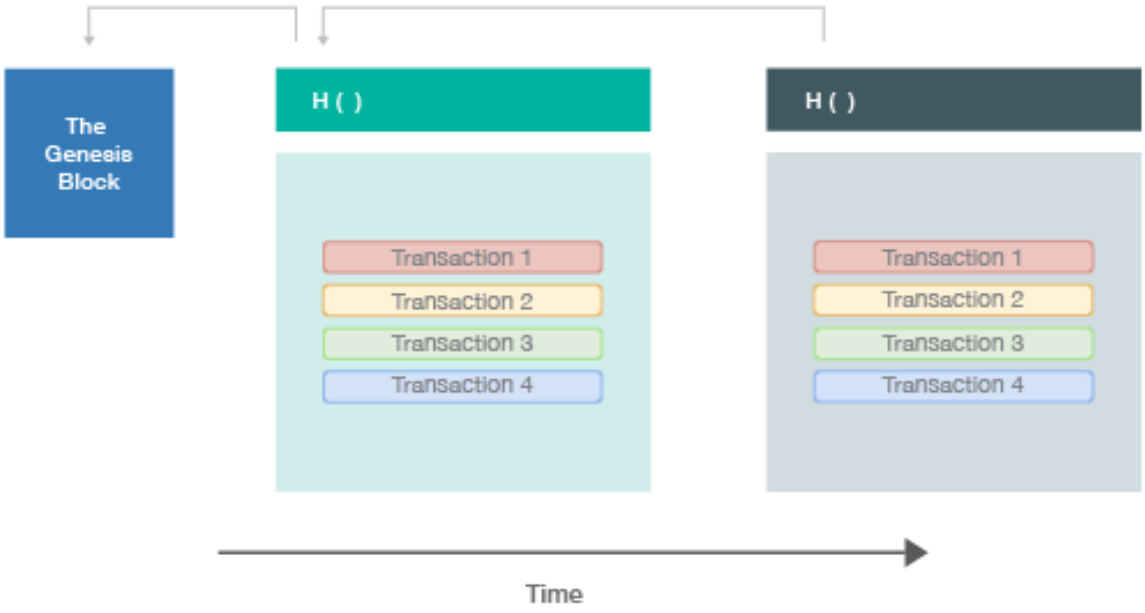

Figure 3-5 : Working principle behind blockchain as reproduced from [77]

3.6.2 Types of Blockchain

Blockchain can be broken down based on permission to the blockchain data. The two common types are the public blockchain and the private blockchain.

Public Blockchain - A public blockchain is a blockchain, which allows anyone (node) to access, the blockchain through either reading from or writing to the blockchain [79]. In other words, anyone (node) connected to the internet irrespective of place (in the world) can participate in the either reading or writing to the blockchain and validating the block. There is therefore no central authority in this type of blockchain making it a fully decentralised blockchain [80]. Some examples of popular public blockchains includes bitcoins, Ethereum, Factom, Blockstream etc [81].

Private Blockchain - A private blockchain allows for read and write to the blockchain data and but restricted to a limited to a predefined list of entities. In other words, the ability to read or write to the blockchain is permitted and managed centrally by an entity. The scope of access in private blockchain is that it is limited such as within a company or home etc. Example of these

applications include database management, auditing, etc which are under the control internally and for a single entity [79],[80]. Some private blockchain examples includes Multichain, Chain, Blockstack etc.[81]. In this thesis, multi-chain was chosen because it is an open source tool; meaning that it is a free tool. Also, the reason for choosing a private MultiChain tool is because we want to be able to identify devices and users within a confined domain and for us to be authenticate them because they are restricted within an environment which will make it easier to identify them based on credentials assigned. In addition, another reason is to simulate the environments that this solution targets. That is the home environment for which its provenance data can only accessed by authorized people.

3.6.3 Use cases of Blockchain

The blockchain technology has received lots of attention from different sectors including asset management [82], healthcare[83][84], finance[85], real estate [86][87] and in the government institutions[88] [89]. Moreover, blockchain serves as the underlying foundation for bitcoin [90]. Christidis et al [78], discuss that a combination of IoT and blockchain is powerful and can transform several industries, allowing for new business models and novel distributed applications. The cost of maintaining the centralized database is expensive. Also, from the user perspective there has been issues of trust with regards to Smart Tv's and its transparency of sending user viewing pattern. For example, an article [91] dubbed "VIZIO to Pay $2.2 Million to FTC, State of New Jersey to Settle Charges It Collected Viewing Histories on 11 Million Smart Televisions without Users' Consent" collect viewing data on 11 million consumer TVs without consumers' knowledge or consent. This shows a lack in trust since users have no idea that their personal viewing information is being gathered and sent. Christidis et al. [78] explains that such issues can be resolved with a blockchain which ultimately brings openness and security on share data. Wang et al. [92], argues that the authenticity of human information is an important factor. As such hinders the cost and efficiency of human resource management. To solve problem discovered above Wang et al. [92], proposes a blockchain model that tries to reduce the risk of authenticity of human resource information. Moreover, they assert that the model based on blockchain resolves a lack of discrimination of authenticity with regards to human information and enables an efficient and effective way of managing human information. Azaria et al. [93] argues that rules and regulations governing medical records as well as long administrative

procedures have brought about lots of inconsistencies and efficiencies when handling electronic medical records in the health domain. However, to resolve such issues the authors exploited the use of Blockchain technology in a large scale medical record system to aid in the handling medical data and to ensure that the data is not only auditable but accessible for easy retrievable via a detailed log.

3.7 Summary

In summary, we reviewed concepts relating to our research to have a background for better understanding our research. We understood that Internet of Things has gained popularity. Due to its popularity, lots of research have been conducted and its application used in various sectors. Furthermore, understood provenance and how it has been used in the Internet of Things. REST concepts including RESTful web services, ontologies and blockchain and some of their use cases were introduced and discussed. However, after gathering extensive research papers from various sources on provenance in Internet of Things, our research reveals that lots of the work conducted focuses one perspective. That is from the request perspective a user always makes inference with the help smart device to determine source of data or service. Table 1.1 shows the summary background works. Additionally, based on these papers that have been reviewed in this chapter, our research reveals that little or nothing has been done regarding the response perspective that is considering IoT devices who respond with data to determine the sources of a request. Consequently, identifying a gap in the research that motivates the need to "trace bi-directionally where data comes and where request comes from with the Internet of Things". A proposed blockchain-based provenance system to support provenance to answer where data comes from and where request comes from is discussed in the next chapter.

Internet of Things	IoT, IoT Protocols	• The things communicate with each other and to the cloud • Major protocols - CoAP, MQTT for IoT communication • CoAP is more suitable because it allows for discovery services via RESTful

Provenance	Provenance, Provenance in the Internet of Things	• Provenance describes the lineage of a digital object allowing for traceability. It achieves reliability, transparency, quality and auditability. • Applied in different domains including cloud, databases and IoT • Most research focus on the client perspective: where the data come from whereas little has been done on the server perspective: where the request come from.
Ontologies	Ontologies, RDF	• Formal and explicit representation of shared conceptualisation. • Used to describe concepts, attributes and to establish relationship among concepts. Ontologies have been used in different domains for various reasons. • Some syntax for modeling data are Resource Description Framework, Web Ontology Language but focus in this thesis will be RDF
REST	REST, RESTful web service and its applications	• REST is an architectural style that guides in the development of web services • Restful web services follow four key principles Resource identification, Uniform Interface, Self-Descriptive messages and Stateful interactions • Applied in different domain including cloud, SOA, IoT etc.

Blockchain	Blockchain and its applications	• Decentralized solution that allows for recording digital transactions in a secure way using cryptography. • Public and private blockchains are the two major types. • Ensures trust, transparency, immutability and auditability. • Used in different domains including financial sectors, real estate etc. • Blockchain technology can be used to support provenance because it keeps track historical and the chronology of data which allows for further decisions to be made.

Table 3-1 : Summary of Background works

CHAPTER 4

4. PROPOSED SYSTEM ARCHITECTURAL DESIGN

The main objective of this thesis is to explore "how we can use blockchain to support provenance in terms of tracing where data came from and where the request came from in IoT" as highlighted previously in chapter 2. To achieve this goal, a blockchain-based system is proposed to support provenance to trace the bi-directional communication among IoT devices and the cloud. The proposed blockchain-based provenance system will store provenance information during the lifecycle of a request and response model. In other words, when a request is sent by a mobile user for sensor data acquisition, any IoT devices in the cloud that intercepts or forwards the request also have its provenance information stored until it gets to its destination. Similarly, when the response from the server hosting the sensor data is sent to the mobile user, the same recording of provenance information is achieved by the proposed system. Having the blockchain-based provenance system to capture all the provenance information associated with the request and response lifecycle will allow mobile devices, users as well as servers to make inferences and make intelligent decisions. The overview, architecture, System components, key roles and System workflow, system interaction, architectural layers, data model and format is described in the following sections.

4.1 Overview

To lay emphasis on the problem discovered in this chapter, let us consider the same scenario as presented and described in chapter 2 (see Figure 2-1 and Figure 2-2) where you live with your family in a smart home. While leaving for work, you turned off your Smart TV with your phone via an application installed on your smartphone. Your brother who also left for school remotely turned on the smart TV via the internet to watch his favorite TV program but later forgot to turn it off. After you arrive home, you realise that your smart TV is on. Obviously, this might be strange and shocking to you in a sense that you are not able to tell how the smart TV was turned on. And in this scenario, you will be prompted to inquire who turned on the smart TV and how did the person achieve it. Assuming your brother connects directly to the smart TV as shown in Figure 4-1 and is confined within the same smart home, it becomes easy to trace and determine. However; if the connection is indirect, where you have the cloud containing multiple IoT devices

as shown in Figure 4-2, it becomes very difficult. Hence, making it extremely hard to know who or which devices are being used behind the cloud to achieve that purpose. This puts your smart home at risk.

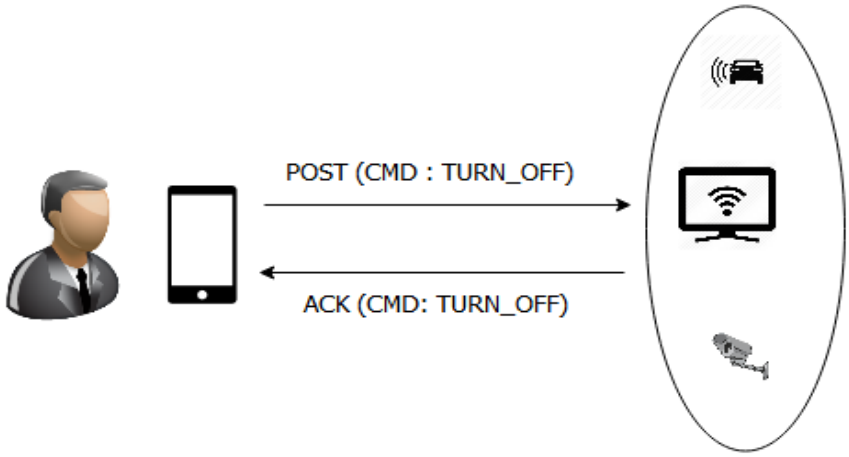

Figure 4-1 : Direct Communication between a mobile user and sensors

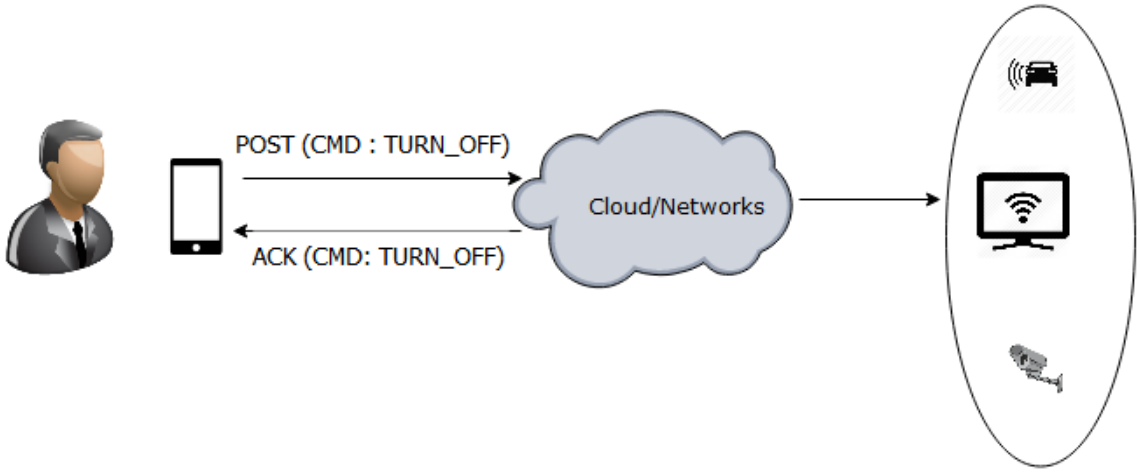

Figure 4-2 : Indirect Communication between a mobile user, cloud and sensors

Based on the above scenario, this work proposes a provenance based blockchain system aimed at tracing the bi-direction communication in IoT to answer the question; where data comes from and where the request comes from. In this regard, when a request is sent by a person using a mobile device, the context information including id, from, To, Timestamp, resource, method is

recorded into a decentralised blockchain database. When the request gets to the cloud which contain lots of resources (devices), whichever device intercepts the request, creates a new request matching it to the CoAP msgID of the initial request and then sends it to the sensor controller. In addition, the device attaches the original data from received from the initial source and then similarly its context information including id, from, To, Timestamp, resource, method is recorded in the blockchain. The sensor control then fetches that data and then also records its context information ID, from, To, Timestamp, resource, method into the immutable blockchain database. The same interaction occurs vice versa when the response data is being sent. Decisions based on the provenance record can then be made by users, devices, intelligent systems and businesses alike. Some benefits expected of this proposed solution is to ensure:

- Transparency
- Immutability
- Auditability
- Accountability

The diagram in Figure 4-3 shows the system interactions within the proposed solution.

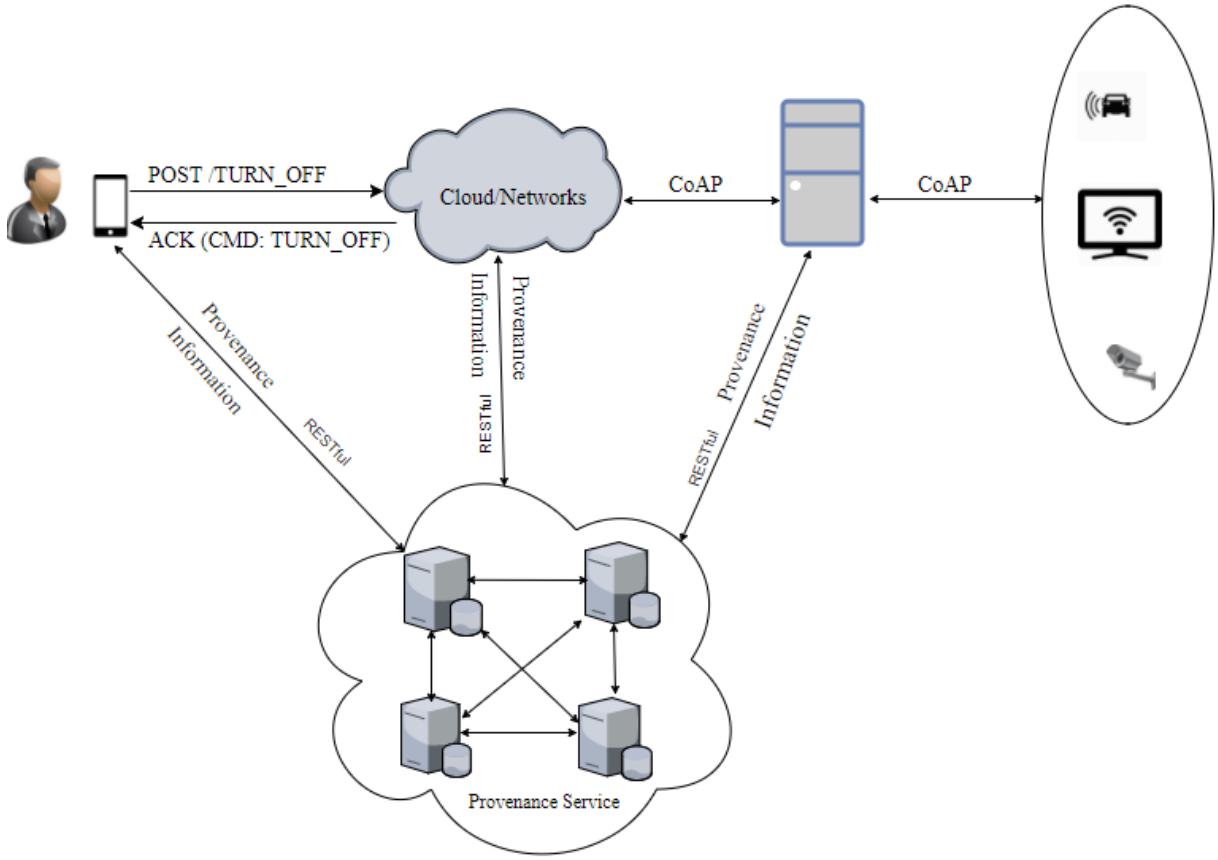

Figure 4-3 : The Proposed Solution Architecture

Following are the presented basic components of the system, and the key roles explained in detail.

4.2 System Components

The proposed System architecture is composed of client, agent, Server, Provenance Database and network. Furthermore, an explanation of the components is provided and it is highlighted as shown in Figure 4-4.

i. ***Client:*** A client is a computerised device that starts a communication by making a request to either a remote or local server for resources. Examples of smart-devices include: smart phones, tablets, laptops, desktops etc. Any of these can represent a client.

ii. ***Agent:*** An agent in this thesis context will serve as a proxy, an intermediary computerised device that will provide services on behalf of the clients. The agent will be

represented as an IoT device capable of processing and making intelligent devices. Multiple agents can be deployed in a resourced-constrained environment. Resourced-constrained, in the sense that these devices have less power, low computation and very limited memory capacity. These devices may include but not limited to Raspberry Pie, computers, Intel Edison etc. Raspberry pie 2 an IoT device will be considered in this work for simulating this role.

iii. ***Server****:* A server is a computerised device either personal or high end computer that provides resources to the client. A server usually, may have lots of memory (Random Access Memory), storage (Hard drive capacity) and network bandwidth capacity. However, any computerised devices including personal computer, workstation, low powered IoT devices including Raspberry Pie, Intel Edison or high end powered computer capable of processing can act as a server. In the context of our proposed system architecture, a work station will be considered for simulating this role.

iv. ***Provenance Database*****:** The provenance database will be used for storing all the provenance information which includes MsgID, timestamp, From (ClientURI), To (Server URI), resource, method. This will be the key component that will enable us to answer our research questions and to meet of our objectives stated in chapter 2. In our proposed architecture, we will use blockchain technology, a shared and a decentralised database that support provenance tracking. In this work and for performing the experiment, we choose a private-based multi-chain. The reason for choosing a private based MultiChain is that it is an open source and free. Also, it is a good choice for inter-organizational record capturing and auditing since all nodes are trusted and well-connected and at the same time maintain privacy. Any modification or changes can be easily tracked because they fall under the same organisation. This free tool will capture and store the provenance information. MultiChain is an already made platform that allows developers to create and deploy private blockchains within or between organisations. To provide more details on blockchain, MultiChain has support for windows, Linux and Mac, and provides an interface for Application Programmable

Interfaces (API) and command-line instructions. Furthermore, for privacy and control, it comes bundled with its package.

v. ***Network***: Communication is very essential to every system. Having computers or IoT devices without them talking is of no importance. Hence for computers IoT devices to communicate with each other and to share information across board, network is very crucial. Thus, a network will be required for effective communication among these devices. In this case, CoAP, and RESTful will be used for the communication among the devices in the context of this research.

4.3 Key Roles and System Attributes

Within the proposed system architecture, interaction will exist among three key roles namely; originator, agent and service provider. The role of each is explained. Also, the contextual information also known as the system attribute will be explained. Interaction or communication among the key roles is described in Figure 4-4.

i. **Originator (client)**: The client as explained above, will player a role in the system as the originator. The initial request will be issued by the originator. The originator will communicate with other devices via CoAP. However, communication to the blockchain will be via RESTful interface. Also, context information that will be captured are;
    - MsgID
    - Timestamp
    - From (ClientURI)
    - To (Server URI)

ii. **Agent**: The agent will play a role as a proxy. The agent could be many assuming with have lots of IoT devices with such constrained environment. The agent will also interact with the client and the service provider (server) via CoAP. However, to write to the block chain the agent will interact via RESTful. Also, context information that will be captured by the agent
    - MsgID

- Timestamp
- Receivedfrom (ClientURI)
- From (AgentURI)
- To (Server URI)

iii. **Service Provider (Server)**

The server will play the role of the service provider by providing the resource to the requestor. The interaction to the agent will be via CoAP. However, to write to the blockchain the service provider will interact via RESTful. Also, context information that will be captured by the server provider will be

- MsgID
- Timestamp
- Receivedfrom (AgentURI)
- To (Server URI)

4.4 Overall System Work flow

A very detailed description of the proposed system is provided in this subsection. The proposed system records all the provenance information as system components interacts with each other. Every request sent by a client, goes through an agent and finally to the server and vice versa. In other words, capturing information with regards to where data comes from and where request comes during communication between the mobile devices, cloud and IoT devices. Additionally, using a decentralised database will be vital to store and provide a mechanism that supports provenance in terms of tracing the bidirectional communication in IoT, which will ultimately bring transparency within IoT and for intelligent decisions to be made by users and IoT nodes based on provenance information.

As illustrated in Figure 4-4, the sequence diagram highlights the three key roles that are used in the proposed system. These are the client (originator), agent, and the server (service provider). The client connects to the blockchain and then store its provenance information such uniform resource identifier (URI), Server's URI and timestamp. In this thesis, we refer to provenance as the record of the lifecycle of requests/responses in IoT communication. Moreover, the client then

sends a CoAP request and adds its URI to it's payload. If the request is successful sent, the Agent in the cloud then intercepts the request and then log its context information into the provenance database, which is the blockchain. The Agent creates a new request chain in the blockchain and then log its information including (from, To, Timestamp, resource, method) into the blockchain. To make sure that message being is the same, the Agent uses a random generated msessageID that matches the original message sent by the client. The Agent then adds it URI to the payload information it receives from the client and then send the request to the server. It is worth pointing out that there is a possibility that request issued by the client will traverse through different routes considering the number of agents (proxies) within the constrained environment. Upon receipt of request by server, the server then connects to the blockchain and then it's information including (from, To, Timestamp, resource, method) is recorded as provenance information in the blockchain. The server further creates a new request, and then ensures that message received from Agent is the same. The server uses a random generated messageID that matches the original message sent by the Agent. This is done because requests transmitted across multiple devices needs to have the same messageID. In addition, the server creates a separate chain called response chain contained in the blockchain. The server then stores the record as provenance information which includes: from, To, Timestamp, resource and method. Furthermore, the server adds it URI to the response payload and then sends the processed response data along with the response status to the agent. As indicated earlier, within the cloud there is a possibility that request sent by the server might go through different paths and not necessarily the same path it took. Therefore, any agent within the cloud can intercept the message and then deliver the request. However, sticking to this same diagram, the agent receives the response, and then creates the response information which includes (from, To, Timestamp, resource, and method) inside the response chain and then logs its provenance information into the blockchain. A random generated messageID is used to match the original messageID. Furthermore, the agent appends its URI to the response payload received from the server. The agent further sends the response data to the client along with the status code. The client receives the response from the agent along with the response status. On receipt, the client again creates a response information within the response chain and inside the blockchain, which is a representation of the provenance information including (from, To, Timestamp, resource, method). Furthermore, to make sure that message ID stays the same, a random generated messageID is used to match the received

response. The client also acknowledges that it has received the message by appending its URI to the payload information received from the agent. All provenance information is captured and stored in the storage layer which is the decentralised blockchain (multichain). Due to the immutable nature of provenance blockchain database, and the fact that it keeps a long chronological log or history of information, it makes this technology a good fit for provenance tracking. This because it is an append only database, meaning that once provenance information is added it becomes extremely difficult to make changes because all activities conducted and in this case all communication among IoT devices from the very beginning to the end is captured. As result of provenance recording among the communication the system become transparent and auditable since all entities at any point in time can make inferences based on the historical data.

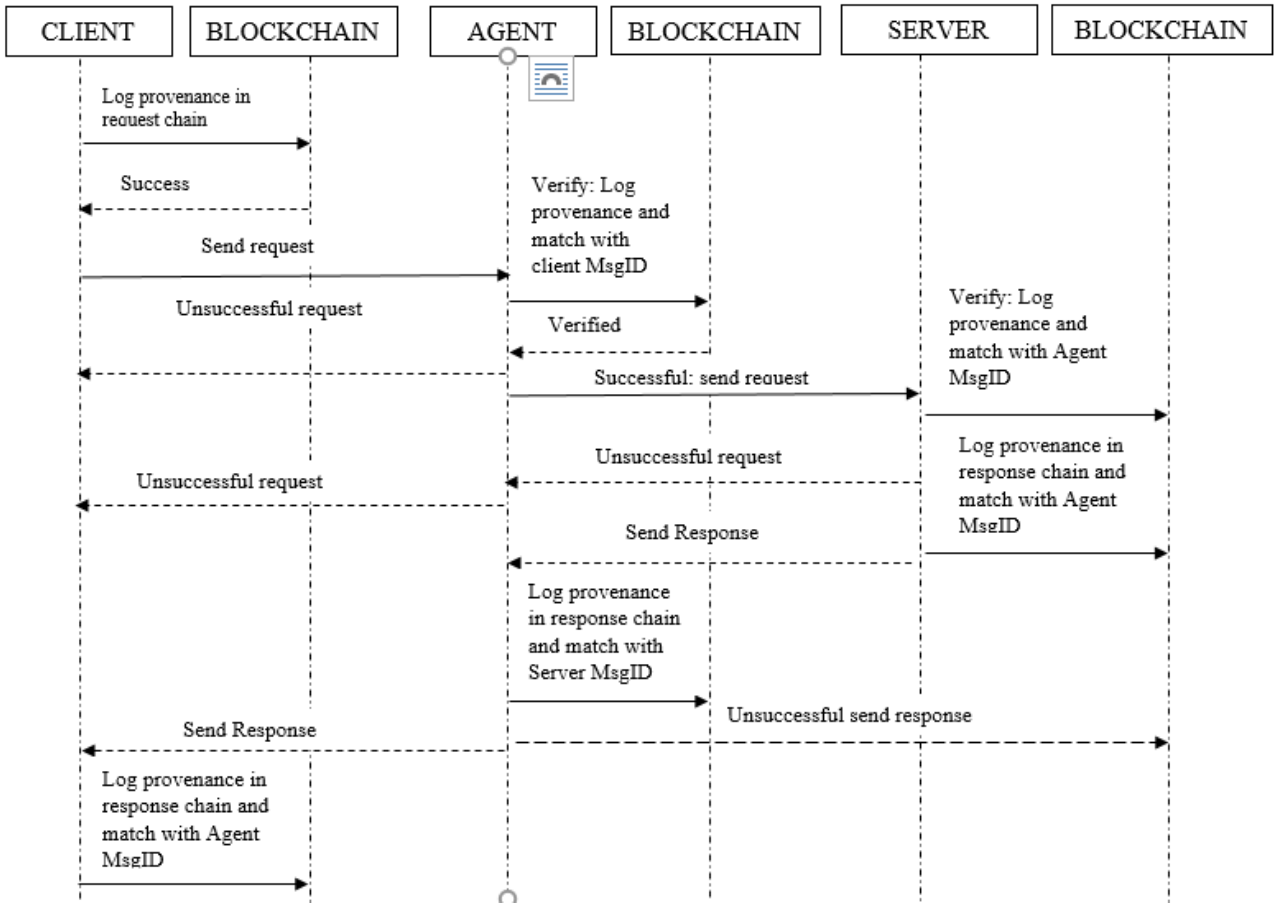

Figure 4-4 : A sequence diagram shows the interactions among the roles as adapted from [94]

4.4 System Interaction by Component

In this section, each individual system component is broken down and the interaction between them is described in detail. The diagram in Figure 4-5 shows the how the overall system works.

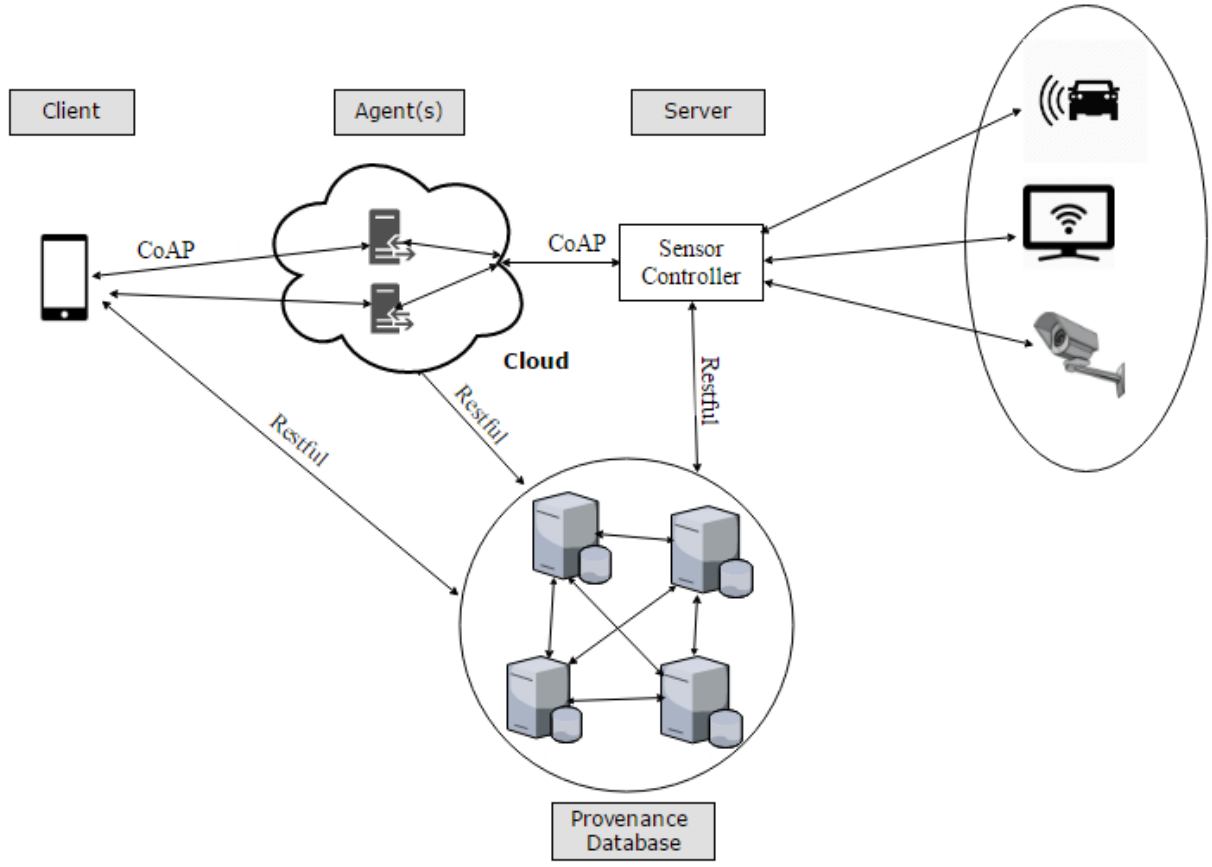

Figure 4-5 : Work flow of the Proposed System

In Figure 4-5, the mobile device (client) sends a CoAP request and then appends its uniform resource identifier, which uniquely identities the device. An example of the URI is /Emmanuel. In addition, at the same time, important context information is stored automatically by the provenance database. The extracted context information stored makes up the provenance information. The provenance database can either be stored locally or in the cloud. This is illustrated in Figure 4-6.

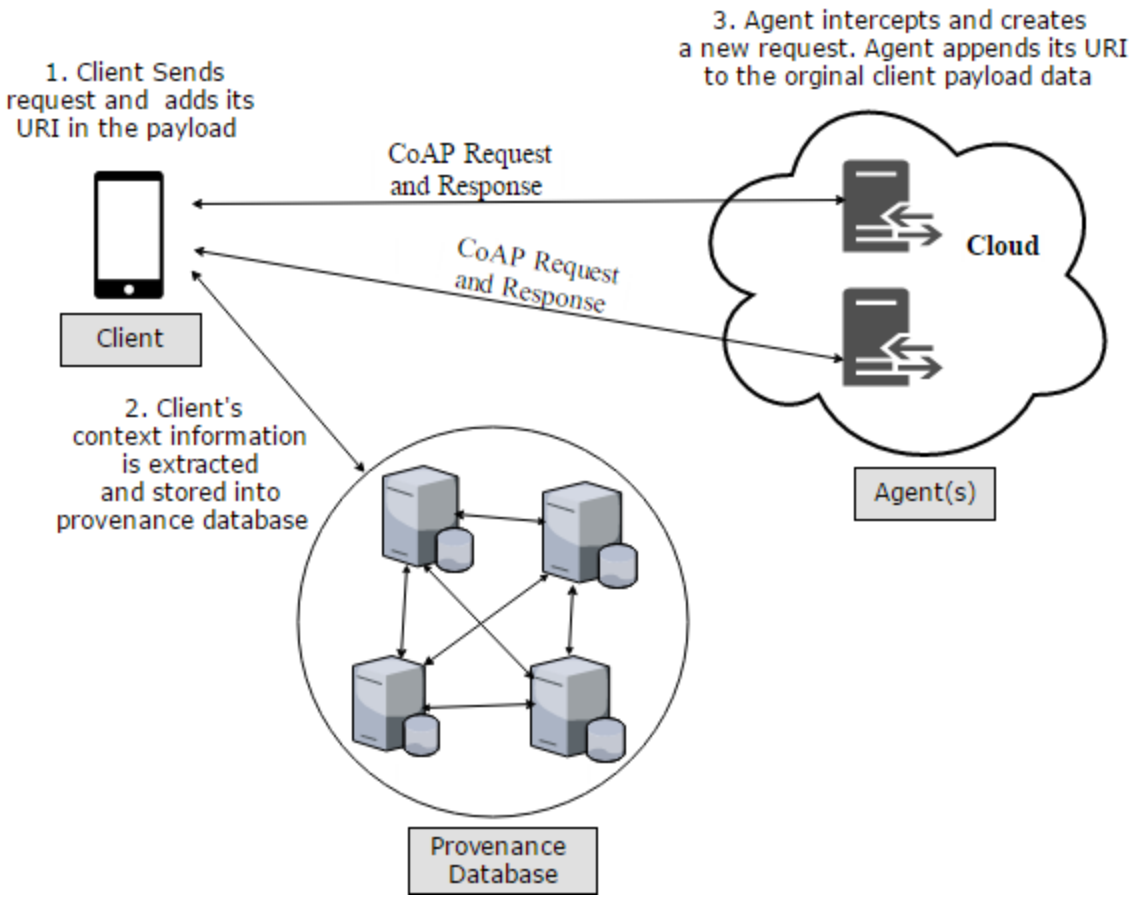

Figure 4-6 : Interaction between the mobile device, provenance database and the cloud

The cloud receives the request. The cloud may contain IoT devices (agent(s)), therefore any agent that intercepts the request, creates a new request matching it with the original request (MsgID), and then appends its URI to the payload (/Emmanuel/Agent). Concurrently, important context information is extracted from the agent and stored automatically by the provenance database. As explained previously, the stored context information forms the provenance information. This is illustrated in Figure 4-7.

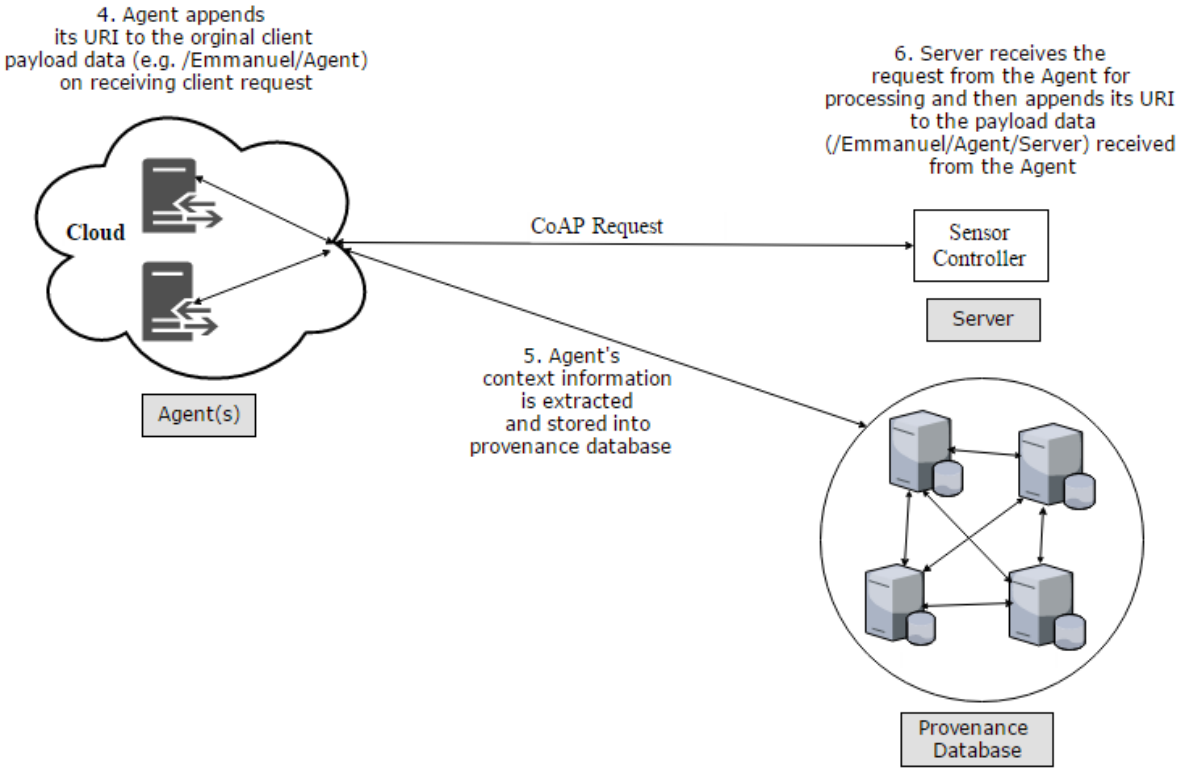

Figure 4-7 : Interaction between the cloud, provenance database and the server.

The sensor controller (server) finally receives the request and then appends its URI to the payload (/Emmanuel/Agent/Sever). Concurrently, its important context information is extracted and stored automatically by the provenance database. Additionally, if the request is successfully processed, the sensor data is acquired and then sent to back to the cloud, containing the agent(s). The processed request is sent along with the response data together with the response status. That is the cloud receives the response data and then logs its context information in the response chain and then further send it to the client (originator) Also, upon failure a response status is sent in a similar manner as described previously. The response is sent and logged into the provenance database in the same way as described in the request phase. For example, the traversal path of the response will be as follows. (/Sever/Agent/Emmanuel). This is shown in Figure 4-8.

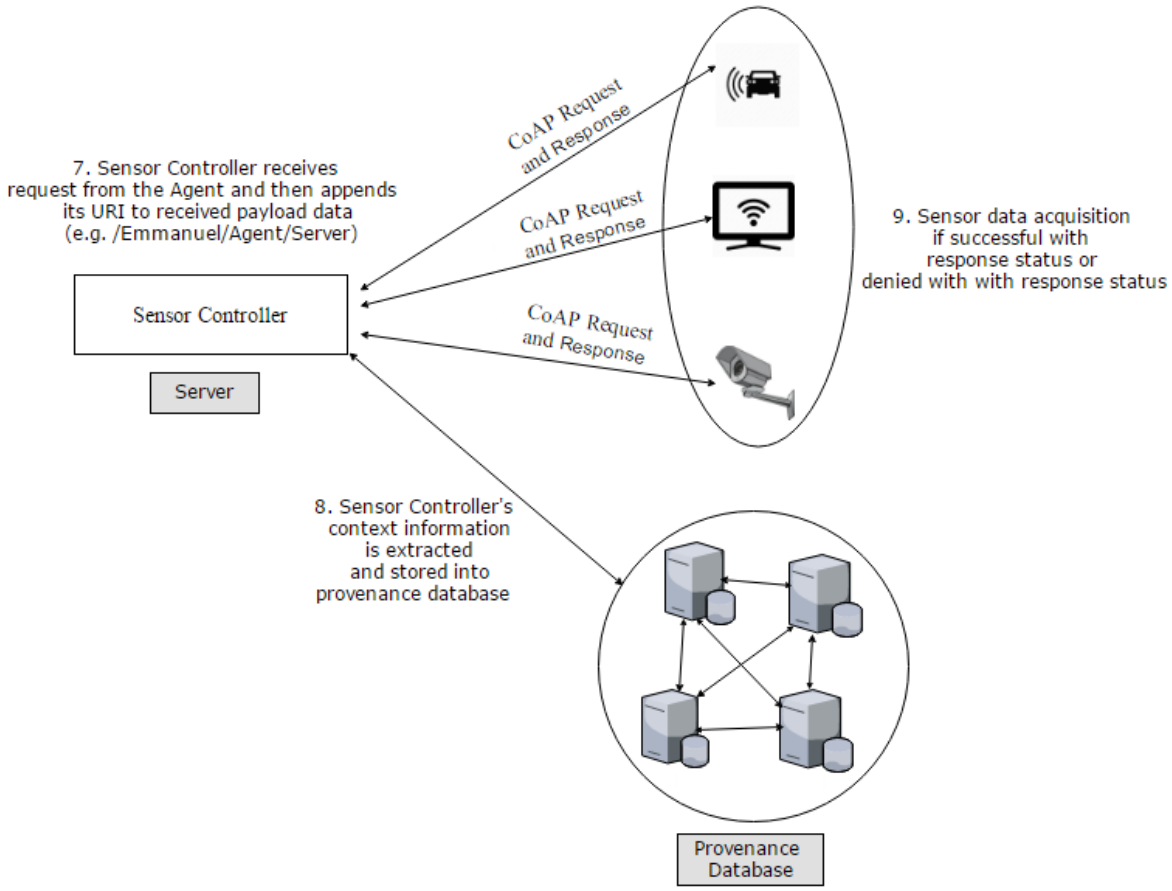

Figure 4-8 : Interaction between the server, provenance database and the sensors.

4.6 Abstract Architectural Layers

Following the idea of Trusted Things framework proposed by [63], the proposed architectural design will be broken down into three layers namely; Core service layer, ontological layer and the storage layer. Figure 4-9, shows the layers within the architecture.

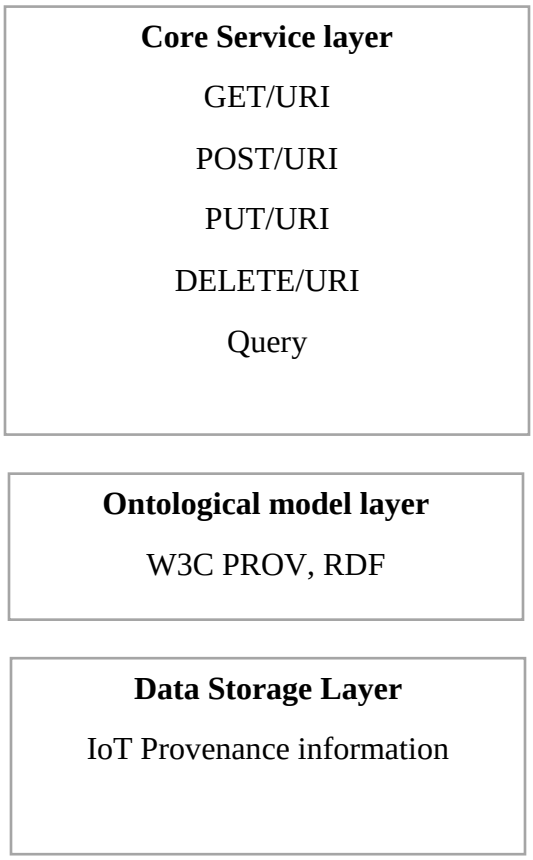

Figure 4-9 : Abstract Architectural Layers

It is worth noting in this thesis that the layers are dependent on each other. For instance, the core services layer, ontology layer and the storage are interdependent, and all work together seamlessly to achieve the intended purpose. The functionalities of each layer are explained below:

- **Core service layer -** The core service deals the interfacing of the system. Users with the aid of smart devices such as smartphones, laptops, IPADS, etc can interact with the system based the CRUD operation. Users can either, create a resource by using POST, read a resource by using the GET command, update a resource by using PUT, delete a

resource by using the DELETE command. Additionally, Users can query for provenance information to make decision.

- **Ontological model layer** - With this layer, the information regarding IoT devices, their relationships as well activities can be described using ontology such as Resource Description framework. Furthermore, provenance information captured can be modelled using RDF or W3C PROV-O. This will aid in describing the Entities and Agents involved in various Activities that occurred within the constrained environment and with regards to request and response cycle.

- **Storage layer** - Within the storage layer, provenance information such as timestamps, uri-identifying each entity as well as agents, resources and commands used within the request and response life cycle will be captured and stored within this layer.

4.7 Data model and Format

Provenance data can be modelled using ontological languages as explained in chapter 2. However, in the context of this work we modelled the interaction among the various roles using the Resource Description Framework (RDF). Because RDF, allow the resources to be described and to establish a relation among them, we model the how provenance of the roles as well as how they interacted or with each other. The diagram in Figure 4-10 shows how the provenance of request and response are modelled based on the payload naming scheme which is a representation of a URI.

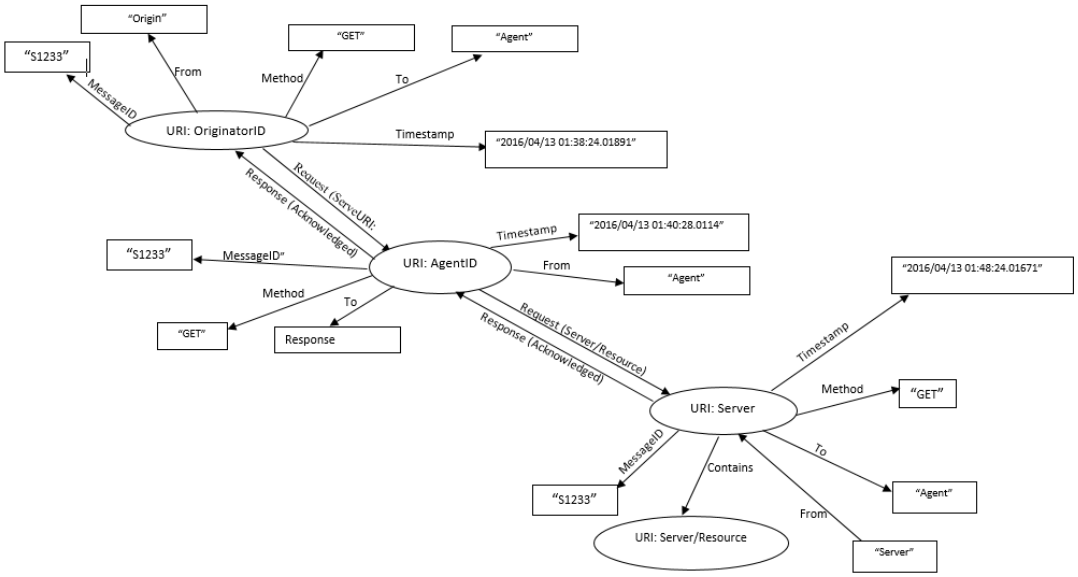

Figure 4-10 : RDF Data model of Request/ Response

The content of the CoAP request as well as the payload and sample provenance information is in JSON (JavaScript Object Notation) format. All the communication that occurs among these devices and the data format of exchanging data is in JSON. A sample provenance of the request and response in JSON format is shown in Figure 4-11.


```
{
  ▾ publishers: [
      "1MfqbnlBy5vXwrvwrA6LWXe542HrhtqSok49sB"
    ],
    key: "8392",
    data: "{\"messageid\":8392,\"from\":\"ralph\",\"to\":\"agent\",\"method\":\"\\u0001\",\"timestamp\":\"2017/08/11 05:37:12.025191\",\"uri\":\"/server/resource\"}",
    confirmations: 0,
    txid: "fd4c42892809b0d70401725a6ca2193d2f469cbfdd9f892a67671f810a7080ad"
},
{
  ▾ publishers: [
      "1MfqbnlBy5vXwrvwrA6LWXe542HrhtqSok49sB"
    ],
    key: "8392",
    data: "{\"messageid\":8392,\"from\":\"agent\",\"to\":\"svcprovider\",\"method\":\"\\u0001\",\"timestamp\":\"2017/08/11 11:37:12.468215\",\"uri\":\"server/resource\"}",
    confirmations: 0,
    txid: "238c3030ef5ce43afa9f0966a34406e5d9325c94f44384b204ecd57148a478e0"
}
```


Figure 4-11 : Sample Provenance for Request

```
{
  ▾ publishers: [
        "1YGmhfG4f4QRd5GGkXoYLMaw4aTqqY5VE48Z6U"
    ],
    key: "8392",
    data: "{\"messageid\":8392,\"from\":\"svcprovider\",\"to\":\"agent\",\"method\":\"\\u0001\",\"timestamp\":\"2017/08/11 05:37:13.880565\",\"uri\":\"server/resource\"}",
    confirmations: 5,
    blocktime: 1502451447,
    txid: "98a1c9f4cc0349f7588083d2039aecde51819f74920b447ccff72d7272ef9437"
},
{
  ▾ publishers: [
        "1YGmhfG4f4QRd5GGkXoYLMaw4aTqqY5VE48Z6U"
    ],
    key: "8392",
    data: "{\"messageid\":8392,\"from\":\"agent\",\"to\":\"ralph\",\"method\":\"\\u0001\",\"timestamp\":\"2017/08/11 11:37:12.620675\",\"uri\":\"server/resource\"}",
    confirmations: 5,
    blocktime: 1502451447,
    txid: "578575ea5588ca7148ae92da9d234eee79f8e552207232a6cb6dae775603dce7"
}
```


Figure 4-12 : Sample Provenance for Response

4.8 Summary

In summary, a bi-directional provenance blockchain-based system is proposed. The system architecture is explained together with System components, Roles and System attribute overall workflow and System interaction. Furthermore, the Abstract Architectural layers and data format was explained in this section. The proposed decentralized provenance-based system can help us answer our research question; where did data come from and where did request come from. Moreover, it can enable the trace of who, how and the timestamps from the client's perspective as well as the server's perspective. Thus, allowing requests/responses to be tracked a bi-direction as explained in the research question in section 2.1. During the interaction among IoT devices the system records all the provenance information with regards to requests/responses and then stores it in a blockchain called Multichain. Every provenance record is synchronized to all the nodes in the blockchain network for consensus to be made. This makes it difficult to modify. The proposed system provides features that ensure transparency, immutability, auditability and accountability.

CHAPTER 5

5. IMPLEMENTATION AND PERFORMANCE EVALUATION

This chapter describes the implementation of the proposed system initially explained in chapter 4 and the experiments conducted with regards to the system. In addition, the performance evaluation of the implemented decentralised provenance based system is conducted. It is very important to compute the system overhead cost whiles these IoT devices interact with each other. This will help us determine if the proposed system can meet up with the standard requirements in a real-world application in terms of performance. Earlier on, Chapter 4 discussed about the proposed system required to bidirectional trace requests and responses and to answer questions such as:

- Where did the response (data) come from? That is from the client's perspective.
- Where did the request come from? That is from the server's perspective.

The details for the implementation, experiments, performance metrics and data collection and performance analysis is described in the following sections.

5.1 Detailed Implementation

The proposed system is implemented in Golang, [95], a programming language designed by Google. It is an open source programming language that allows developers to design and develop simple but reliable and efficient software. To implement the prototype system, an existing package/library that provides both a CoAP client and server implementation written by Dustin Sallings was adopted to ease with rapid development. The implementation code of the proposed system is broken down into four parts namely; the client-side, agent-side and server-side and the database code-implementation. All codes are written in the same language specifically for the client, agent and server as highlighted above.

5.1.1 Client-Side Implementation Code

The client-side code was designed in Golang, and basically the client-side implementation allows for the client to connect to the blockchain, that is the multichain database. The code is implemented in such a way that the moment the client connects to the multichain, a WriteRecord

function logs the client's: MessageID, From, To, Method, URI, and Timestamp into it and then sends the request containing the payload which is the uniform resource identifier of the client. The implementation code performing this functionality is shown in Figure 5-1.

```
func main() {

  //CLIENT CONNECT AND MAKE REQUEST
  c, err := core.Connect(core.AgentAddr)
  if err != nil {
    log.Fatalf("Error: %v", err)
  }

  // create a record
  var actor core.MessageRecord
  actor.MessageID = core.CreateMessageID()
  actor.From = core.OriName
  actor.To = core.AgentName
  actor.Method = string(core.Method)
  actor.URI = core.GlobalURI
  actor.Timestamp = core.GetCurrentTime()
  err = core.WriteRecord(core.TableName_Request, actor)
  if err != nil {
    log.Printf("Error: %v\n", err)
    return
  }
```

Figure 5-1 : Client-side code implementation

5.1.2 **Agent-Side Implementation Code**

Similarly, the agent-side was built with Golang, and the main function of the agent was to simulate that of the real world IoT devices likely to be in the cloud. In the agent-side implementation, the code was designed to connect to the blockchain that is the multichain database. The code is implemented in such a way that the agent listens on a port and then accepts the request from the client. Upon receipt of client's request, the agent acknowledges and then connects to the server that contains the resource being requested for. Concurrently, the agent connects to the multichain, using the WriteRecord function and then also logs its provenance information: MessageID, From, To, Method, URI, and Timestamp into it. The agent then sends

the request containing the payload which is the uniform resource identifier of the agent. The implementation code for the agent described is shown in Figure 5-2.

```
coap.FuncHandler(func(l *net.UDPConn, a *net.UDPAddr, m *coap.Message) *coap.Message {

      // log.Printf("Received Message From %v: %#v\n\n", a, m)
      log.Printf("Received Message ID (%d)\n\n", m.MessageID)
      // var data core.Payload
      // json.Unmarshal(m.Payload, &data)
      if m.IsConfirmable() {

        //connect to remote connection with the clients request
        c, err := core.Connect(core.SvcproviderAddr)
        if err != nil {
          log.Fatalf("Error: %v\n", err)
        }

        // write a record to the multichain
        var actor core.MessageRecord
        actor.MessageID = m.MessageID
        actor.From = core.AgentName
        actor.To = core.SvcName
        actor.Method = string(m.Code)
        actor.URI = m.PathString()
        actor.Timestamp = core.GetCurrentTime()
        err = core.WriteRecord(core.TableName_Request, actor)
        if err != nil {
          log.Printf("Error: %v\n", err)
        }
        // data.Trace += "/" + core.AgtName
        req := core.BuildConReq(actor.MessageID, actor.URI, m.Payload)
```

Figure 5-2 : Client-side code implementation

5.1.3 Server-Side Implementation Code

The server side was also developed in Golang. The server hosts the resources that client's request for. To simulate a server in a real-world setting, the implementation code was designed to function in a way that the server listens on a port for incoming request from the agent. The server after receiving a request acknowledges and concurrently, connects to the multichain database, and then with the help of the WriteRecord function, logs its provenance information: MessageID, From, To, Method, URI, and Timestamp into it. The server appends its URI as its identifier. It goes on the to create response stream inside the blockchain database and further logs into the database it's contextual information as highlighted above. The same process is done with the agent and the client since the code is implemented in a way to trace bidirectionally from the client to the agent and from the agent to the server and vice versa. The response is sent along

with the payload which is the URI identifying each device that intercepts the response. The implementation code for the server as described above is shown in Figure 5-3. Each module coded during the implementation phase is tested separately to eliminate bugs within the code and to ensure that it meets the targeted functional requirements. In addition, all the modules are tested to ensure that they interact with each other to achieve the bidirectional trace, which is the intended purpose.

```
func main() {

  log.Printf("Service is running on %v\n", core.SvcproviderAddr)

  log.Fatal(coap.ListenAndServe("udp", core.SvcproviderAddr,
    coap.FuncHandler(func(l *net.UDPConn, a *net.UDPAddr, m *coap.Message) *coap.Message {

      // log.Printf("Received Message From %v: %#v\n", a, m)
      log.Printf("Received Message ID (%d)\n\n", m.MessageID)

      // var data core.Payload
      // json.Unmarshal(m.Payload, &data)

      // write a record to the database
      var actor core.MessageRecord
      actor.MessageID = m.MessageID
      actor.From = core.SvcName
      actor.To = core.AgentName
      actor.Method = string(m.Code)
      actor.URI = m.PathString()
      actor.Timestamp = core.GetCurrentTime()
      err := core.WriteRecord(core.TableName_Response, actor)
      if err != nil {
        log.Printf("Error: %v\n", err)
      }

      // response the request
      if m.IsConfirmable() {
        // data.Trace += "/" + core.SvcURI
        req := core.BuildAckReq(*m)
```

Figure 5-3 : Server-side code implementation

5.2 **Experimental Setup**

In this section, a description of the experimental setup is provided. The experimental setup component is composed of hardware and software. The hardware required for this experiment comprises of IMAC, workstation and raspberry pie 2. The IMAC will be used to simulate client devices that sends request in an IoT environment. The workstation and the raspberry pie 2 will also be used to simulate the behavior of the server and the agent(proxies) respectively. The detailed specification of each hardware component is listed below:

- Local

IMAC – Client Simulation hardware

- Brand - IMAC
- OS – Mac OS X Yosemite
- CPU - Intel Core i5 CPU @ 3.5GHz
- Memory - 16GB

Raspberry pi 2 – Agent Simulation hardware

- Brand – Raspberry Pi 2 Model B
- OS – Linux Raspbian
- CPU - 800 MHz
- CPU Speed - 3.40 GHz
- Memory - 1GB

Lenovo Workstation – Server Simulation hardware

- Brand - Lenovo M series
- OS – Windows OS 10
- CPU - Intel Core(TM) i7-3770 @ 3.40 GHz
- Memory - 32GB

Lenovo Workstation – Provenance Node 1 Simulation hardware

- OS: Ubuntu 16.04 LTS
- CPU: Intel(R) Core(TM) i7-6700 @ 3.40GHz
- Memory: 4 GB RAM

Lenovo Workstation – Provenance Node 2 Simulation hardware

- OS: Ubuntu 16.04 LTS
- CPU: Intel(R) Core(TM) i7-6700 @ 3.40GHz
- Memory: 4 GB RAM

Lenovo Workstation – Provenance Node 3 Simulation hardware

- OS: Ubuntu 16.04 LTS
- Processor: Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
- Memory: 4 GB RAM

➢ ***Cloud***

- OS: Ubuntu 16.04 LTS ((3 Nodes))
- CPU: Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
- Instance type: t2.medium
- Availability zone: us-east-1b
- Memory: 4 GB RAM

5.2.1 Physical Layout

This section, describes how the physical system including the machines were laid up. The layout was setup in the local and in the cloud (sees Figure 5-4 and Figure 5-5). The actual physical layout in the local is shown in Figure. In the local, communication among the devices was established with network connection using Ethernet technology. The server, raspberry pie (agent), provenance nodes (MultiChain) and the client connects via Ethernet cable to a switch. It worth mentioning that wireless technology can be used as well. We run each implementation code on top of the operating system on each machine before the actual experiment. In other words, client code is installed on the client node, the agent code and server codes also installed on the agent and server nodes respectively. This is done because each implementation code performs different function based on the role they play in the system. Because, the database used is based on blockchain technology, we needed three nodes for testing during the experiment.

These nodes run Ubuntu Linux 16.04 as their individual operating systems. The three provenance servers/nodes, host blockchain software called MultiChain which configured on each node to simulate replication across board and to ensure consensus at anytime during the bi-directional request and response cycle with regards IoT environment which is expected of blockchain. We point out that the system under consideration can be extended to support multiple blockchain databases as well as multiple agents. The experimental setup and design and is shown in figure. Performance metrics, Data collection and performance analysis is explained in the next section.

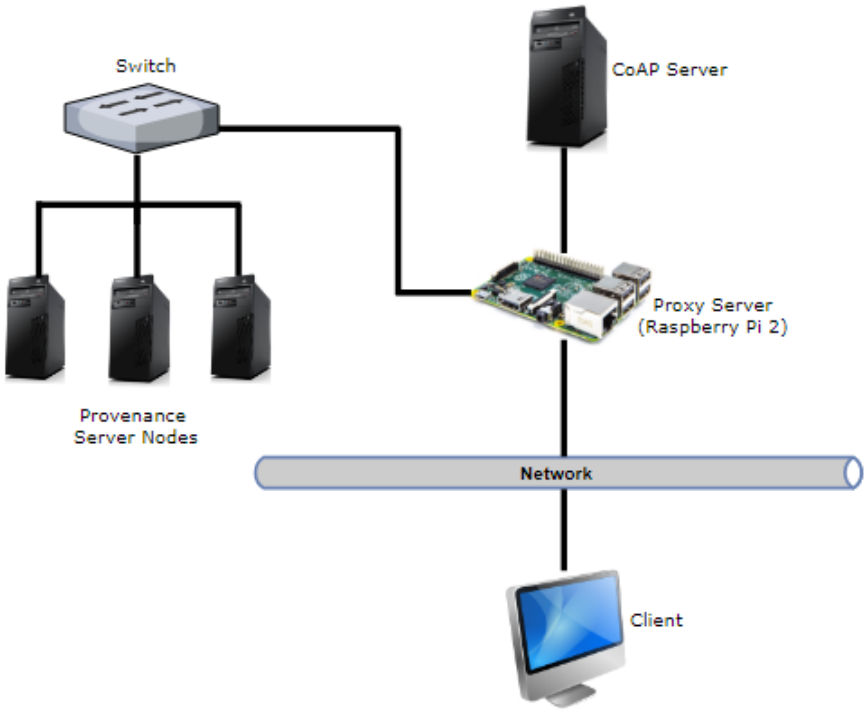

Figure 5-4 : Layout/Setup in local as adapted from [94]

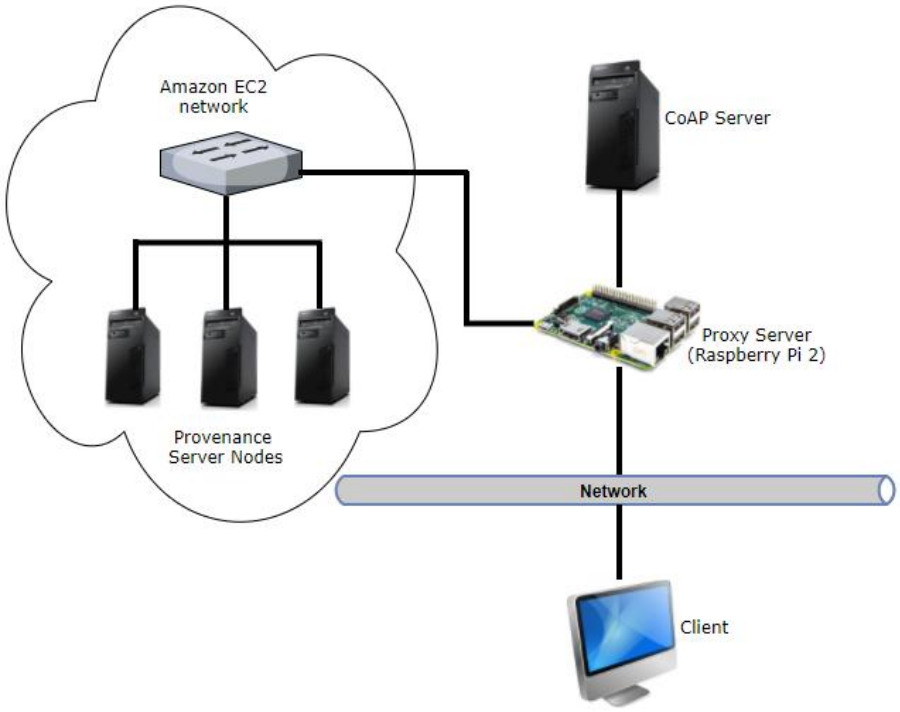

Figure 5-5 : Layout/Setup in Cloud

5.2.2 Data collection

This section gives an account of how the experiment was performed after the initial setup and how the data from the experiment were generated, collected and then aggregated. The experiments are simulated to provide the user with how the system can be used in the real-world setting, specifically in an IoT environment. To run the experiment, we downloaded and installed Apache JMeter on the client machine (iMac), an open source performance tool that support a variety of performance metrics. For Apache Jmeter to support CoAP protocol a third-party library was used and configured to work with the performance too. Furthermore, a dataset of varying payload sizes comprising of 8, 16, 32, 64, 128, 256, 512 and 1024 bytes were selected. This was chosen to simulate the kind and the amount of data that can sent in a real world using small IoT devices and based on the varying sizes supported by the CoAP protocol which is specifically targeted at small devices. In addition, the payload represents a unique URI naming scheme that can be used to identify an IoT device in the context of our system. The application was run on the client and repeated 20 times for each payload size. Big amount of data was

generated based on the payload sizes used. During the experiment, factors such as network connectivity, number of devices as well as the time taken to write to the database. The performance metric data highlighted in section 5.2.3 was selected. The throughput, response time, number of users from 1, 10, 20, 40, 80 were chosen to send concurrent users the CoAP request. well as was collected as csv format in a log file. All data captured on response time were summed up and averaged. The same was done for the throughput data. Scalability of the system was also tested with a varying number 1, 10, 20, 40 and 80 users to ensure that the system can support multiple users.

5.2.3 Performance Metrics

The response time, number of concurrent users, throughput and varying payload sizes are considered as performance metrics in other to simulate it with the system in a real-world resource constraint environment. The performance matrices considered for evaluation are described below:

- *Response Time:* The response time is the time taken for a mobile client device to send CoAP message to the agent, from the agent to the server and vice versa including writing to the blockchain database. The response time is measured in milliseconds (ms).
- *Throughput:* The throughput refers to the amount of CoAP data sent over a network per second and it is measured in Kilo Bytes per second (KB/s). Other unit of measures is in transaction per second.
- *Payload size:* The payload size refers to the CoAP data. That is the amount of data in bytes that is carried alongside a request.
- *Number of users:* The number of users refers to the number of concurrent users used to send request to a system. It determines the scalability of a system.

5.3 Performance Evaluation

In this section, the performance evaluation is discussed based on the experiment and data collected in the previous section. To better understand the behavior of the system, the system is

evaluated in two parts. The system is evaluated in local and in the cloud. Additionally, the metrics for which the system was evaluated is as follows:

- Response time (milliseconds)
- Throughput (Number of Transaction per second)

In the first part of the evaluation, the system is evaluated in local. In local, the payload size ranging from 8, 16, 32, 64, 128, 256, 512 and 1024 collected in section 5.2.2 as well their corresponding response time was plotted as shown in Figure 5-6. The graph is based on one user. In other words, a single user was made to send a request 20 times repeatedly. Based on that the response time data for each payload size was calculated by divided each averaged response time by the total number of iterations. We observed that a payload range starting from 8 through to 64 maintains a steady slope with respects to the response time; however, at 128 bytes the response time drops slightly whiles maintaining a level through to 1024.

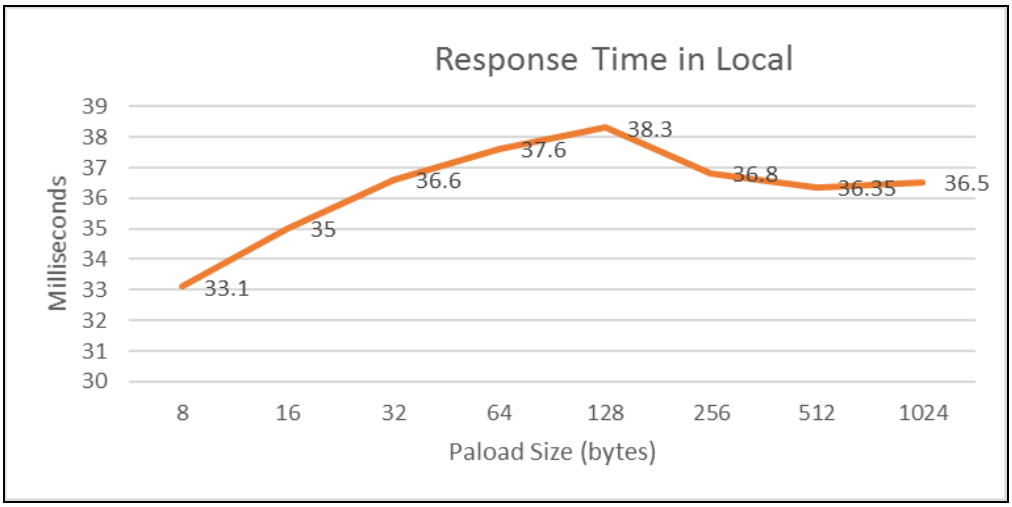

Figure 5-6 : Response time vrs Payload based on a single user in local

Furthermore, to determine the response times based on varying payload sizes as well as different number of concurrent user's, a graph was plotted based on the data collected and calculated. The graph is shown in Figure 5-7. The x -axis represents the number of concurrent from 1, 10, 20, 40 and 80 whereas the y-axis represents the response time. We noticed from the graph that starting from a single (1 user) through to 40 users, there is a continuous increasing response time in close linear slope; However, there is a close 2.5 times response time increased on 80.

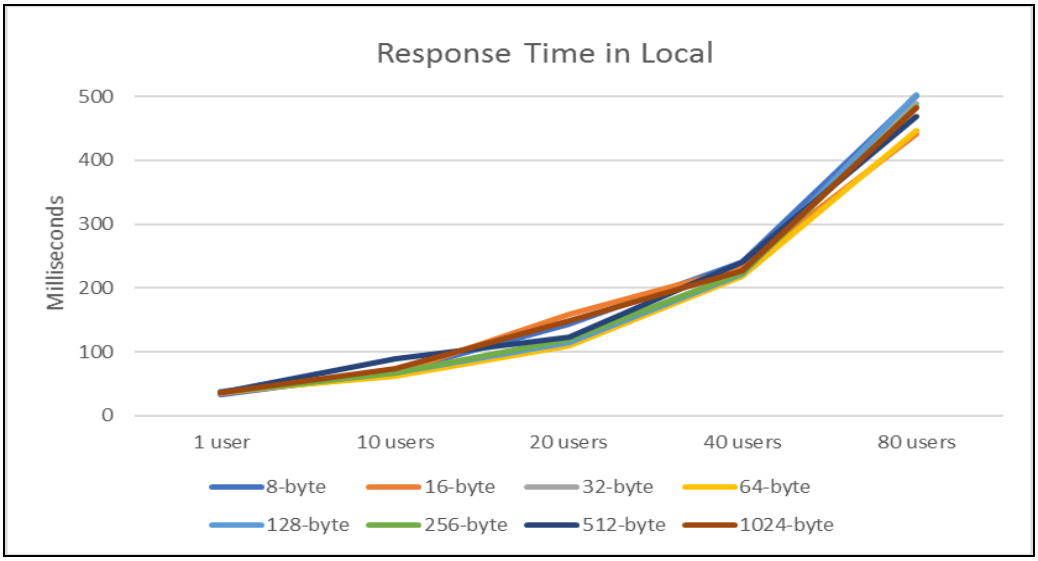

Figure 5-7 : Response Time vrs Number of Users based on Payload in local

In addition to the above generated graph, the scalability of the system was explored and evaluated accordingly. Scalability refers to how a system can handle loads and within the context of this thesis the unit of measure for throughput for our scalability is transactions per second. To arrive at thiese throughputs, the collected data was aggrecated and calculated. Each throughput for each payload size was calculated based on the number of user(s).

Based on the results from the throughput, a graph was plotted. This is shown in Figure 5-8. In the graph, the x -axis represents the number of users starting 1, 10, 20, 40 and 80 whereas the y-axis represents the number of transaction per second. Based on the graph we observe that the throughput can scale from 1 to 10 users. And there is a slower throughput increased from 10 users to 20 users. However, we do not see any significant throughput increasing after 40 users. Based on this observation, we realised that there is a system boundary either on computation or network capability which does not allow a further throughput increase. The system is unable well to handle that amount of load as the number of users increases from 10. Hence keeping the graph limited whiles increasing number of users.

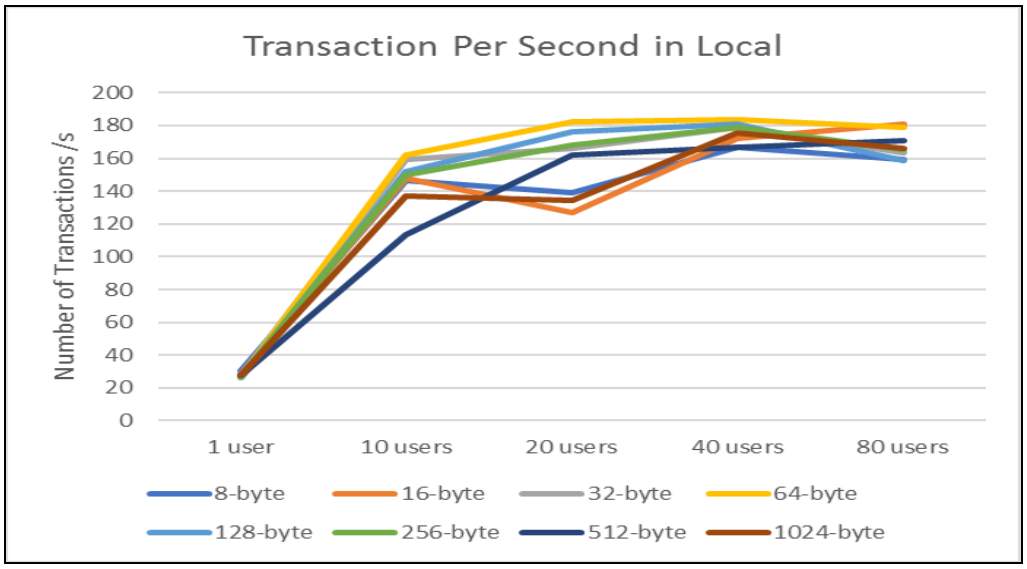

Figure 5-8 : System Scalability (Number of transaction per second) in local

The second part of the system performance evaluation was evaluated in the cloud. Similar to the local, the payload data sizes ranging from 8, 16, 32, 64, 128, 256, 512 and 1024 was collected and selected with their corresponding response time. The graph was plotted as shown in Figure 5-9. The graph is based on one user. In other words, a single user was made to send a request 20 times repeatedly. We observed that a payload range starting from 8 to 256-byte payload size maintains their response time in the range between 380ms to 384ms. However, from 512-byte payload size, there is an higher response time increased. Compared to our local test, there is about 10 times slower response time observed due to Amazon AWS server setup and shared network communications. Also long distance communication contributes the slower response time.

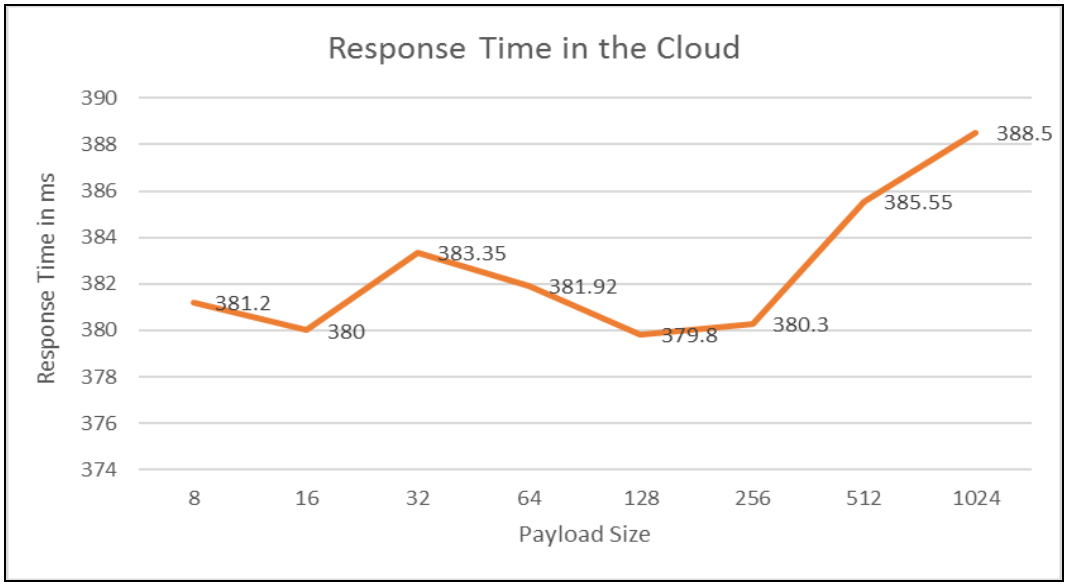

Figure 5-9 : Response time vrs Payload based on a single user in Cloud

In Figure 5-10, the x -axis represents the number of users from 1, 10, 20, 40 and 80 whereas the y-axis represents the response time. We observed that as the number of user's increases, the response time also increases especially from 40 user case. In comparison to the local, the response time rises because the network is engaged with the Amazon service and the Internet. Thus, these become dominant factors that impacts on the significant increase in the response time. The network delay may have been one factor that increased the response time. Additionally, the virtual resources may be shared and are in a controlled environment restricting resource allocation as well. It was also observed that as long as the payload size increases, the response time is increases.

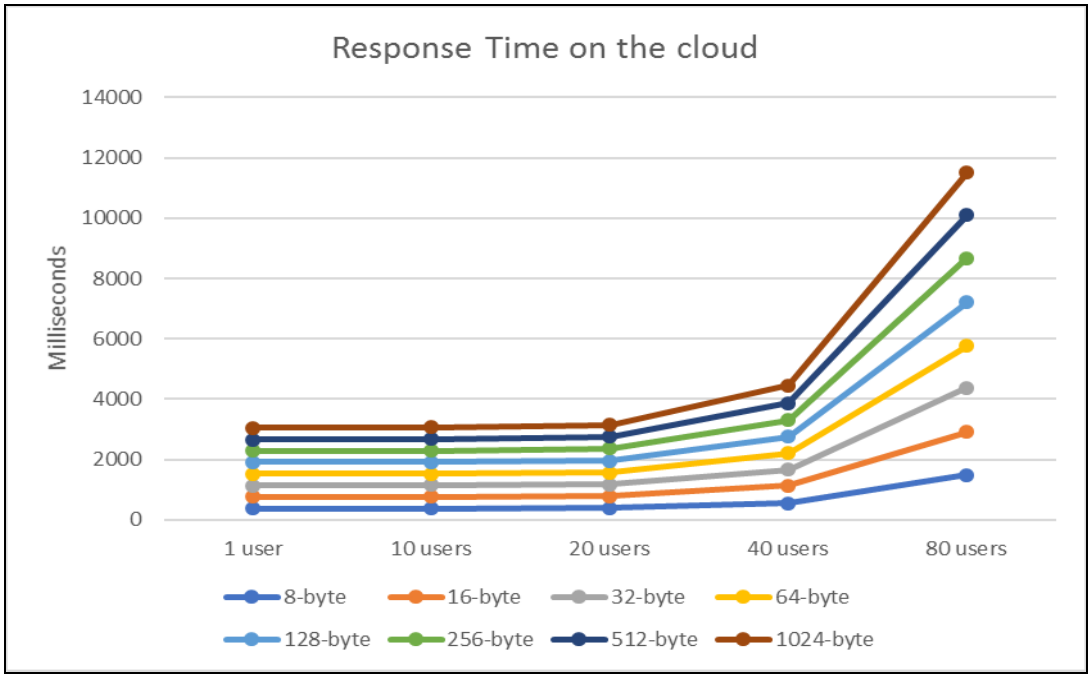

Figure 5-10 : Response Time vrs Number of Users based on Payload in Cloud

Also, in Figure 5-11, a graph showing the number of user on the x -axis from 1, 10, 20, 40 and 80 is plotted against the throughput (number of transaction/second) on the y-axis. Based on this graph and in comparison to Figure 5-10, we observed that due to the significant increase in the response time it causes the throughput drops the moment the number of users reaches 80 as shown in Figure 5-11. Therefore processing less amount of transaction per second .

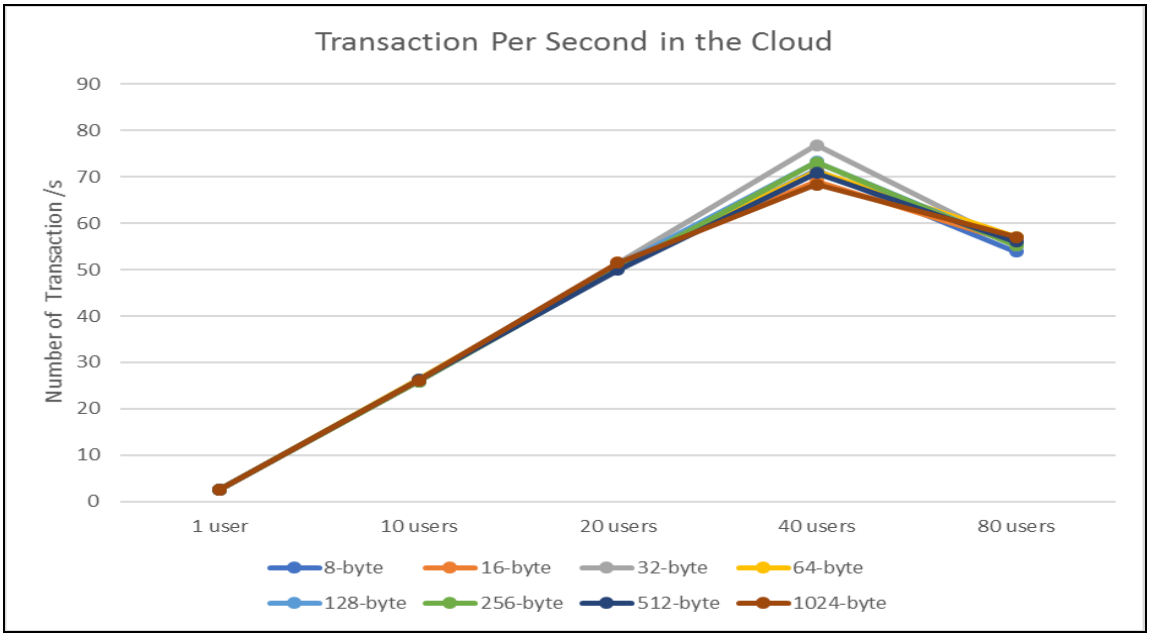

Figure 5-11 : System Scalability (Number of transaction per second) in local

5.4 Summary

This chapter focused on how the system was setup, and how the experiments were conducted. In addition, the performance of the system is evaluated taking into consideration the response time, throughput and varying payload sizes. Section 5.2 describes in detail how the system was setup to Section 5.2.1 describes the actual physical layout in both local and cloud. Section 5.2.2 describes how the data was collected. Section 5.2.3 discusses the performance metrics. Finally, Section 5.2.4 discusses the performance evaluation of the system. It is expected that this system meet will answer our research question. In addition, the results of the performance evaluation show that the system can scale to 40 users in the cloud environment. Also because of the local experiment setup, our system is limited in terms of network and computation; hence the system is unable to scale well after the 10 concurrent users. Furthermore, we notice that the 40 users become the turning point for a bigger response time in both test environments.

CHAPTER 6

6. CONCLUSION AND FUTURE WORKS

In conclusion, Internet of Things allow for things to connect to each other and to the Internet so that data can be collected and knowledge mined which will be used to create a new generation of application and services that will better our society. These things have low power, limited memory and very limited computation. With the introduction of IoT, it has been predicted that the total number of devices that will connect to the Internet will increase to 28 billion by 2021. The growing number of IoT devices connecting to the internet and the amount of data coming out of these devices introduces a problem with respect to traceability within the IoT. In other words, it poses the question: how do we determine where request came from and where response (data) came from? In addition, lots of research have been conducted in this regard, of which provenance have proved be key in solving the issue of traceability. Provenance techniques have been used in solving the issues with regards to determining the sources of data. This shows that provenance have proved vital in addressing such issues. However, based on reviewed literature, little or nothing has been done regarding the response perspective that is from where request came from. Additionally, blockchain, a shared distributed tamper-proof database has proven to support and to be a good fit for provenance due to the advantages it brings. Blockchain provides essential features including immutability, transparency and auditability. Append only database does not allow modification of the provenance records and captures historical records in a linear form starting from the initial record it falls in-line with the idea of provenance of which this thesis capitalizes on as a supporting tool to store our provenance records. Hence this thesis, proposed, designed and developed a system that uses blockchain to support provenance in IoT with the aim of answering the questions: where did the request come from and where did the response come from? In other words, this thesis combines the trace of the bi-direction combining both the request and response in the IoT.

As shown in the Figure 6-1 the developed system includes some key components: client, agent, server, and provenance database. The client is a computerized device that starts a communication by making a request to either a remote or local server for resources. The agent serves as a proxy, an intermediary computerized device that will provide services on behalf of the clients. Multiple agents can be deployed in a resourced-constrained environment. The server owns the resources

and responds based on the client's requests. The provenance database, blockchain-based, stores all the provenance information which includes MsgID, Timestamp, From (ClientURI), To (Server URI), resource, method. This will be the key component that will enable us to answer our research questions.

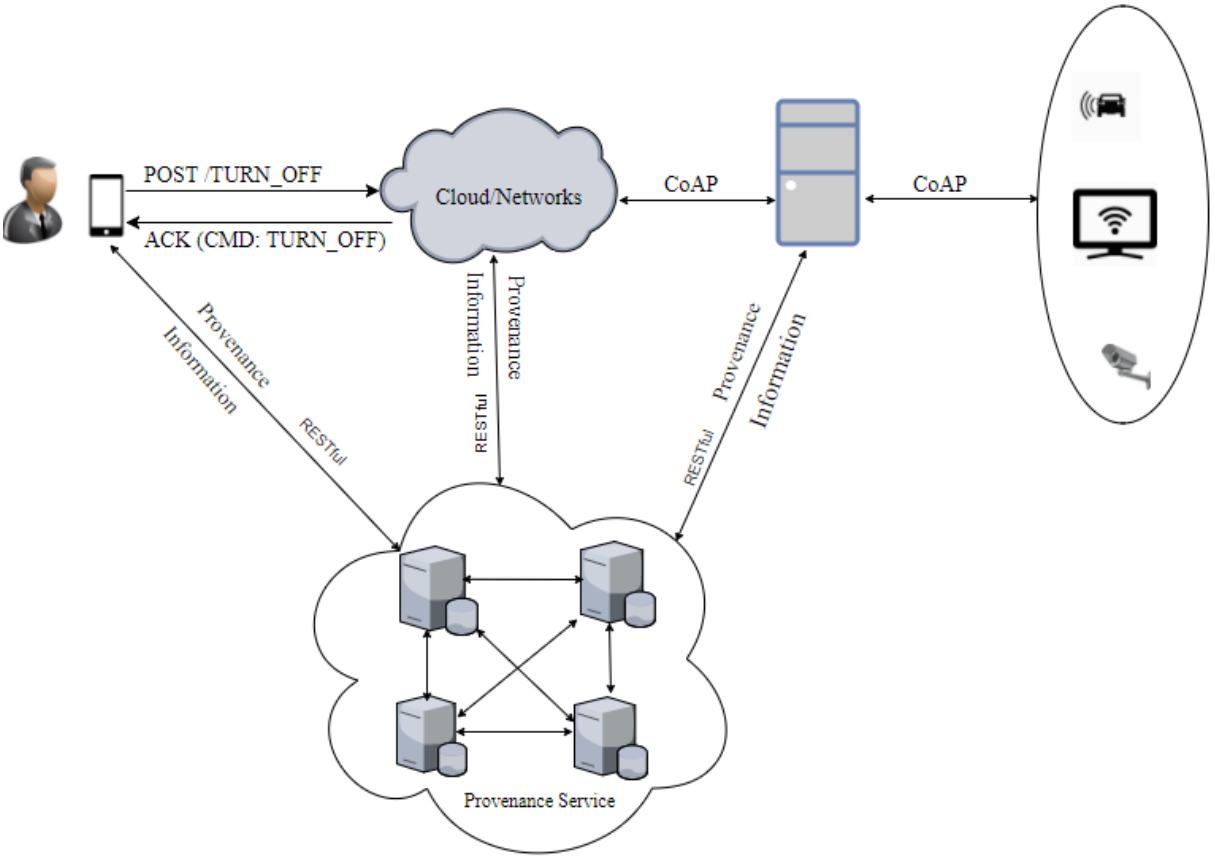

Figure 6-1: Developed provenance-based blockchain System

It is expected that by solving this problem, this work contributes to the following items that are listed:

- A prototype system to trace the communication path - A decentralized system that will make use of the provenance information captured. Also, not only store and provide well-documented evidence showing the origins as well as the chronology of ownership but stores it in an immutable form preventing modification to the provenance record. Based on this system, auditing becomes possible in that based on the record captured the record can be checked and then entities involved in the various request and response cycle

activity can be held accountable. This provides openness since everything is made transparent for all devices to see.

- Prototype system performance - The system is tested in both the local confinement. That is within the local and the cloud to ascertain its responsiveness (how quick it gets a response after sending a request) with regards to the response time. Also, the performance of the system is measured to determine the scalability. In other words, how the system will behave with real-world workloads.

- Analysis and results of the system performance - The results of the performance evaluation show that the system can scale to 40 users in the cloud environment. Also because of the local experiment setup, our system is limited in terms of network and computation; hence the system is unable to scale well after the 10 concurrent users. Furthermore, we notice that the 40 users become the turning point for a bigger response time in both test environments.

In addition, some papers with regards to this work and within this domain have been published.

- Kaku, E., Lomotey, R. K., & Deters, R. (2016). Using Provenance and CoAP to track Requests/Responses. Proceedings of the 13th International Conference on Mobile Systems and Pervasive Computing, volume 94, pp.144-151, Montreal, Quebec, Canada 15-18 August 2016.

- Lomotey, R. K., Pry, J. C., Sumanth, S., Kaku, E., Deters, R. 2017. Wearable IoT Architecture. Proceedings of the 13th IEEE World Congress on Services (Services 2017), pp.62-63, 25-30 June 2017, Honolulu, Hawaii, USA.

- Lomotey, R. K., Pry, J. C., Sumanth, S., Kaku, E., Deters, R. 2017. Middleware Framework for IoT Services Integration. Proceedings of the 6th IEEE International Conference on AI and Mobile Services(AIMS) pp.62-63, 25-30 June 2017, Honolulu, Hawaii, USA.

6.1 Future Works

Using blockchain to support provenance in terms of determining where request comes from and where data comes from is very important. That is tracking the bi-direction communication with regards to request and response has been explored. Also, various aspects of performance of the system have been discussed. However, the system still requires some improvement and features.

This chapter discusses some future works. Although the main system answers its intended research question, some features have not been implemented. A key feature under consideration is access control and authorization.

- Access control and Authorization: Access control allow for access to resources based on permissions. In the current state, the current system is not able to authorize access to resources based on provenance information inference hence hope to consider this area.

- Deploy the system on Intel Edison: The current system only used the raspberry pie 2 for this work; however, in future works I hope to test the system on the Intel Edison IoT device further evaluate the performance. Based on this, both devices (Edison and Raspberry 2) utilising the provenance system can comparatively be evaluated in terms of performance

- Multiple Agent Layers (IoT nodes): The current system uses only a single IoT node. In future works I hope to add more layers as agents to determine and evaluate the performance as we increase the number of nodes.

- Run performance test against more powerful machines with better network and see if the throughput in the local environment can be scaled better.

REFERENCES

[1] NEST, “Nest Thermostat,” 2015. [Online]. Available: https://nest.com/uk/thermostat/meet-nest-thermostat/. [Accessed: 18-Apr-2017].

[2] Phillip Tracy, “The top 5 industrial IoT use cases: Use cases: predictive maintenance.” [Online]. Available: https://www.ibm.com/blogs/internet-of-things/top-5-industrial-iot-use-cases/. [Accessed: 17-Apr-2017].

[3] “Gartner Reveals Top Predictions for IT Organizations and Users in 2017 and Beyond,” *Gartner*, 2016. [Online]. Available: http://www.gartner.com/newsroom/id/3482117. [Accessed: 16-Apr-2017].

[4] P. Cerwall *et al.*, “Ericsson Mobility Report 2016,” 2016.

[5] Cisco, “Cisco Visual Networking Index Predicts Near-Tripling of IP Traffic by 2020,” *Newsroom.Cisco.Com*, 2016. [Online]. Available: https://newsroom.cisco.com/press-release-content?type=press-release&articleId=1771211. [Accessed: 18-Apr-2017].

[6] Y. L. Simmhan, B. Plale, and D. Gannon, “A survey of data provenance in e-science,” *Sigmod*, vol. 34, no. 3, p. 31, 2005.

[7] S. Bauer, “Data Provenance in the Internet of Things,” 2013.

[8] L. Moreau, “The Foundations for Provenance on the Web,” *Found. Trends Web Sci.*, vol. 2, no. 2–3, pp. 99–241, 2010.

[9] I. Abbadi and J. Lyle, “Challenges for provenance in cloud computing,” *... Pract. Proven. (TaPP’11). USENIX ...*, 2011.

[10] L. Xu, W. He, and S. Li, “Internet of Things in Industries: A Survey,” *IEEE Trans. Ind. Informatics*, vol. PP, no. 99, pp. 1–11, 2014.

[11] Lopez Research, *An Introduction to the Internet of Things (IoT)*, vol. Part 1. of, no. November. 2013.

[12] S. Agrawal and D. Vieira, “A survey on Internet of Things,” *Abakós, Belo Horiz.*, vol. 1, no. 2, pp. 78–95, 2013.

[13] O. Said and M. Masud, “Towards internet of things: Survey and future vision,” *Int. J. Comput. Networks*, vol. 5, no. 1, pp. 1–17, 2013.

[14] N. Gershenfeld, R. Krikorian, and D. Cohen, “The Internet of things.,” *Scientific American*, vol. 291, no. 4, pp. 76–81, 2004.

[15] R. Tönjes, P. Barnaghi, M. Ali, and A. Mileo, “Real time iot stream processing and large-scale data analytics for smart city applications,” *Networks …*, pp. 1–37, 2014.

[16] J. Yin, X. Zhang, Q. Lu, and C. Xin, “IOT Based Provenance Platform for Vegetables Supplied to Hong Kong,” pp. 591–596, 2012.

[17] A. Kanuparthi, R. Karri, and S. Addepalli, “Hardware and embedded security in the context of internet of things,” *Proc. 2013 ACM Work. Secur. Priv. dependability cyber Veh. - CyCAR ’13*, pp. 61–64, 2013.

[18] K. Rose, S. Eldridge, and C. Lyman, *The internet of things: an overview*, no. October. 2015.

[19] W. Stewart, “‘The Internet Toaster.’ Living Internet,” *web*, 2000. [Online]. Available: http://www.livinginternet.com/i/ia_myths_toast.htm.

[20] Carnegie-University-Mellon-Computer -Science, “The ‘Only’ Coke Machine on the Internet,” *web*, 2005. [Online]. Available: https://www.cs.cmu.edu/~coke/history_long.txt.

[21] K. Ashton, “That ‘Internet of Things’ Thing,” *RFiD J.*, p. 4986, 2009.

[22] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, “Openmote: Open-source prototyping platform for the industrial IoT,” in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 2015, vol. 155, pp. 211–222.

[23] H. Sundmaeker, P. Guillemin, and P. Friess, *Vision and challenges for realising the Internet of Things*, no. March. 2010.

[24] EPoSS, “Internet of Things in 2020 A ROADMAP FOR THE FUTURE,” 2008.

[25] Giusto; D.; Iera; A.; Morabito; G.; Atzori; L. (Eds.), “The Internet of Things,” in *20th Tyrrhenian Workshop on Digital Communications*, 2010.

[26] R. Buyya and A. Vahid Dastjerdi, “Internet of things : principles and paradigms,” in *Internet of Things Principles and Paradigms*, 2016, p. 354.

[27] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[28] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[29] M. Presser and A. Gluhak, “The Internet of Things: Connecting the real world with the digital world,” *Eurescom Message*, 2009. [Online]. Available: http://archive.eurescom.eu/~pub/about-eurescom/message_2009_02/Eurescom_message_02_2009.pdf.

[30] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things : A survey,” *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[31] G. Sénéclauze, D. A. Becker, P. Purswani, and S. Karekar, “Internet of Things,” 2015. [Online]. Available: http://atos.net/content/dam/global/documents/your-business/atos-white-paper-internet-of-things.pdf. [Accessed: 04-Apr-2016].

[32] Y. Wang and H. Qi, “Research of Intelligent Transportation System based on the Internet of Things frame,” *Wirel. Eng. Technol.*, vol. 3, no. 3, pp. 160–166, 2012.

[33] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, “Internet of things: Vision, applications and research challenges,” *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[34] Business Insider, “BI Intelligence projects 34 billion devices will be connected by 2020,” 2015. [Online]. Available: http://www.businessinsider.com/bi-intelligence-34-billion-connected-devices-2020-2015-11?IR=T.

[35] F. Wang, L. Hu, J. Zhou, and K. Zhao, “A survey from the perspective of evolutionary process in the internet of things,” *Int. J. Distrib. Sens. Networks*, vol. 2015, 2015.

[36] Reuters, “IBM says to invest $3 billion in ‘Internet of Things’ unit,” 2015. [Online]. Available: http://www.reuters.com/article/2015/03/31/us-ibm-investment-idUSKBN0MR0BS20150331.

[37] J. a Stankovic, “Research Directions for the Internet of Things,” *Internet Things Journal, IEEE*, vol. 1, no. 1, pp. 3–9, 2014.

[38] S. Xue, R. K. Lomotey, and R. Deters, “Enabling Sensor Data Exchanges in Unstable Mobile Architectures,” *Proc. - 2015 IEEE 3rd Int. Conf. Mob. Serv. MS 2015*, pp. 391–398, 2015.

[39] Z. Shelby, K. Hartke, and C. Bormann, “The Constrained Application Protocol (CoAP),” *Rfc 7252*, p. 112, 2014.

[40] T. Jaffey, “MQTT and CoAP, IoT Protocols,” *Eclipse*, 2014. [Online]. Available: http://www.eclipse.org/community/eclipse_newsletter/2014/february/article2.php.

[41] K. Kuladinithi, O. Bergmann, Thomas Pötsch, M. Becker, and C. Görg, “Implementation of coap and its application in transport logistics,” *Proc. Extending Internet to Low Power Lossy Networks*, pp. 1–7, 2011.

[42] D. Locke, “MQ telemetry transport (mqtt) v3. 1 protocol specification, IBM developerWorks Technical Library,” *Ibm*, 2010.

[43] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[44] International Foundation for Art Research, “Provenance Guide.” [Online]. Available: https://www.ifar.org/provenance_guide.php.

[45] L. Moreau and P. Groth, “Provenance: An Introduction to PROV,” *Synth. Lect. Semant.*

Web Theory Technol., vol. 3, no. 4, pp. 1–129, 2013.

[46] ArtBusiness.com, “Art Provenance: What It Is and How to Verify It.” [Online]. Available: http://www.artbusiness.com/provwarn.html.

[47] P. T. Groth, “The Origin of Data: Enabling the Determination of Provenance in Multi-institutional Scientific Systems through the Documentation of Processes,” *PhD thesis Electron. Comput. Sci. Univ. Southampt.*, 2007.

[48] J. Cheney, S. Chong, and N. Foster, “Provenance: a future history,” *Proc. 24th …*, pp. 957–964, 2009.

[49] P. Buneman and S. Davidson, “Data provenance--the foundation of data quality,” *B. Data provenance--the Found. data Qual.*, pp. 1–8, 2010.

[50] W. W. W. Consortium, “W3C Consortium Recommendation,” 2013. [Online]. Available: http://www.w3.org/TR/2013/REC-prov-dm-20130430/. [Accessed: 20-Dec-2015].

[51] Q. Zhang, T. Huang, Y. Zhu, and M. Qiu, “A Case Study of Sensor Data Collection and Analysis in Smart City : Provenance in Smart Food Supply Chain,” vol. 2013, 2013.

[52] P. Buneman, S. Khanna, and T. Wang-Chiew, “Why and Where: A Characterization of Data Provenance,” in *Database Theory – ICDT 2001*, 2001, vol. 1973, pp. 316–330.

[53] S. . Alvarez, J. Vázquez-Salceda, T. . Kifor, L. Z. . Varga, and S. . Willmott, “Applying provenance in distributed organ transplant management,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4145 LNCS, pp. 28–36, 2006.

[54] A. Arienzo, C. Coff, and D. Barling, “The European Union and the Regulation of Food Traceability: From Risk Management to Informed Choice?,” in *The International Library of Environmental, Agricultural and Food Ethics*.

[55] “Why and Where: A Characterization of Data Provenance,” in *In: Van den Bussche J., Vianu V. (eds) Database Theory — ICDT 2001. ICDT 2001. Lecture Notes in Computer Science, vol 1973. Springer, Berlin, Heidelberg*.

[56] P. Buneman, A. Chapman, and J. Cheney, “Provenance management in curated databases,” *Proc. 2006 ACM SIGMOD Int. Conf. Manag. data SIGMOD 06*, vol. 1, no. c, pp. 539–550, 2006.

[57] R. Bose and J. Frew, “Lineage retrieval for scientific data processing: a survey,” *ACM Comput. Surv.*, vol. 37, no. 1, pp. 1–28, 2005.

[58] D. Zhao, C. Shou, T. Maliky, and I. Raicu, “Distributed data provenance for large-scale data-intensive computing,” *Proc. - IEEE Int. Conf. Clust. Comput. ICCC*, pp. 8–10, 2013.

[59] J. Yin, J. Li, and P. Ke, “A provenance based scheduling algorithm for logistics chain in IOT,” *Proc. 2013 6th Int. Conf. Inf. Manag. Innov. Manag. Ind. Eng. ICIII 2013*, vol. 1, pp. 324–327, 2013.

[60] R. Beneficiary, I. M. L. FhG, S. H. SAP, E. H. HSG, and C. Jardak, “Internet of Things-Architecture IoT-A Deliverable D1. 3–Updated reference model for IoT v1. 5,” *Cocoa.Ethz.Ch* .

[61] L. Moreau and P. Missier, “PROV-DM: The PROV Data Model,” 2013.

[62] T. Lebo, S. Sahoo, and D. McGuinness, “PROV-O: The PROV Ontology,” 2013.

[63] E. Pignotti and P. Edwards, “Trusted tiny things: making the internet of things more transparent to users,” *ASPI ’13 Proc. Int. Work. Adapt. Secur.*, 2013.

[64] S. Kolozali, M. Bermudez-Edo, D. Puschmann, F. Ganz, and P. Barnaghi, “A knowledge-based approach for real-time IoT data stream annotation and processing,” *Proc. - 2014 IEEE Int. Conf. Internet Things, iThings 2014, 2014 IEEE Int. Conf. Green Comput. Commun. GreenCom 2014 2014 IEEE Int. Conf. Cyber-Physical-Social Comput. CPS 20*, pp. 215–222, 2014.

[65] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” *Building*, vol. 54, p. 162, 2000.

[66] L. Richardson and S. Ruby, *RESTful Web Services*. 2008.

[67] C. Pautasso, O. Zimmermann, and F. Leymann, “RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision, 17th International World Wide Web Conference (WWW2008,” 2008.

[68] S. Tyagi, “RESTful Web Services.” .

[69] L. Zhang, S. Yu, X. Ding, and X. Wang, “Research on IOT RESTful web service asynchronous composition based on BPEL,” *Proc. - 2014 6th Int. Conf. Intell. Human-Machine Syst. Cybern. IHMSC 2014*, vol. 1, pp. 62–65, 2014.

[70] L. Pescosolido, R. Berta, L. Scalise, G. M. Revel, A. De Gloriay, and G. Orlandi, “An IoT-inspired cloud-based web service architecture for e-health applications,” *IEEE 2nd Int. Smart Cities Conf. Improv. Citizens Qual. Life, ISC2 2016 - Proc.*, 2016.

[71] D. Guinard, V. Trifa, T. Pham, and O. Liechti, “Towards physical mashups in the web of things,” *INSS2009 - 6th Int. Conf. Networked Sens. Syst.*, pp. 196–199, 2009.

[72] T. R. Gruber, “A translation approach to portable ontology specifications,” *Knowl. Acquis.*, vol. 5, no. 2, pp. 99–220, 1993.

[73] N. F. Noy and D. L. McGuinness, “Ontology Development 101: A Guide to Creating Your First Ontology,” 2001.

[74] E Miller, “An Introduction to the Resource Description Framework,” 1998.

[75] G. Klyne and J. J. Carroll, “Resource Description Framework (RDF): Concepts and Abstract Syntax,” *W3C Recomm.*, vol. 10, no. October, pp. 1--20, 2004.

[76] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” *Www.Bitcoin.Org*, p. 9,

2008.

[77] Glynn Bird, “Block chain technology, smart contracts and Ethereum.”

[78] K. Christidis and M. Devetsikiotis, “Blockchains and Smart Contracts for the Internet of Things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[79] Bitfury and J. Garzik, “Public versus Private Blockchains,” pp. 1–23, 2015.

[80] V. Buterin, “On Public and Private Blockchains,” 2015. .

[81] Amit, “Public and Private Blockchain Concepts and Examples.” .

[82] A. Mooney, “Blockchain successfully tested in sale of mutual funds,” 2017. [Online]. Available: https://www.ft.com/content/5927fa2c-4c73-11e7-919a-1e14ce4af89b?mhq5j=e1. [Accessed: 11-Jun-2017].

[83] IBM, “IBM Watson Health Announces Collaboration to Study the Use of Blockchain Technology for Secure Exchange of Healthcare Data.” [Online]. Available: http://www-03.ibm.com/press/us/en/pressrelease/51394.wss. [Accessed: 12-May-2017].

[84] M. Mettler, “Blockchain technology in healthcare: The revolution starts here,” in *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services, Healthcom 2016*, 2016.

[85] A. Tapscott and D. Tapscott, “How Blockchain Is Changing Finance.” [Online]. Available: https://hbr.org/2017/03/how-blockchain-is-changing-finance. [Accessed: 20-Apr-2017].

[86] A. Mizrahi, “A blockchain-based property ownership recording system.”

[87] deloitte, “Blockchain in commercial real estate The future is here!”

[88] T. DOUGLAS, “Blockchain a ‘Next Big Transformational Technology’ in Government,” 2017. [Online]. Available: http://www.govtech.com/security/Blockchain-a-Next-Big-Transformational-Technology-in-Government.html. [Accessed: 10-May-2017].

[89] A. Shelkovnikov, “Blockchain applications in the public sector - Deloitte.”

[90] G. Dr. Greenspan, “MultiChain Private Blockchain - White Paper,” 2013.

[91] Federal Trade Commission, “VIZIO to Pay $2.2 Million to FTC, State of New Jersey to Settle Charges It Collected Viewing Histories on 11 Million Smart Televisions without Users’ Consent.” [Online]. Available: https://www.ftc.gov/news-events/press-releases/2017/02/vizio-pay-22-million-ftc-state-new-jersey-settle-charges-it. [Accessed: 06-Jun-2017].

[92] X. Wang, L. Feng, H. Zhang, C. Lyu, L. Wang, and Y. You, “Human Resource Information Management Model based on Blockchain Technology,” in *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE*, 2017, pp. 168–173.

[93] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, “MedRec: Using blockchain for medical data access and permission management,” *Proc. - 2016 2nd Int. Conf. Open Big Data, OBD 2016*, pp. 25–30, 2016.

[94] E. Kaku, R. K. Lomotey, and R. Deters, “Using Provenance and CoAP to track Requests/Responses in IoT,” in *Procedia Computer Science*, 2016, vol. 94.

[95] Google, “The Go Programming Language,” 2013. [Online]. Available: http://golang.org/.