Access level control for shared content in Inter Planetary File System (IPFS)

R. A. H. A. De Alwis
2020

Access level control for shared content in Inter Planetary File System (IPFS)

A dissertation submitted for the Degree of Master of Science in Computer Science

R. A. H. A. De Alwis
University of Colombo School of Computing
2020

Declaration

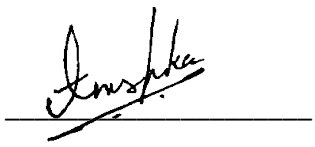
The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: R. A. H. A. De Alwis

Registration Number: 2017/MCS/019

Index Number: 17440194



Signature: Date: 19-11-2020

This is to certify that this thesis is based on the work of Mr. R. A. H. A. De Alwis under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. Kasun De Zoysa

_____

Signature: Date: 19-11-2020

Abstract

Distributed data sharing protocols are now popular through the web users because of the benefits such as high availability, low bandwidth usage and high speed etc. However, these distributed protocols cannot guaranteed the confidentiality of the data, which shared in it. In other words, these distributed systems has not in-build access level control mechanisms.

Currently these distributed systems are using several mechanisms for grant access levels in a data sharing systems. Most of them has centralized node or authority for grant the permissions. Because of the centralized party, these systems are deviate the distributed system properties. Some of the distributed data sharing systems like Acl-IPFS [1] uses the Ethereum blockchain [2] for grant permissions. This access control mechanism is depend on another party.

IPFS (Inter Planetary File System) [3] is a pure distributed file sharing protocol, which originally designed for the permanent web approach. It access the shared content (data) by the hash of the data and apart from the above benefits, there is no node level data duplication in the IPFS network. IPFS system has in-build way for achieve integrity, availability and non-reputability of the data. Even IPFS has the distributed system effectively work as distributed file sharing protocol; it has not in-built access level control mechanism. Therefore, uses cannot share the sensitive data through the IPFS.

In this research, there are comparative study about the available distributed data sharing protocols and currents researches. By the constructive research approach, it proposed a new system called E-IPFS (Extended Version of Inter Planetary File System). The proposed e-IPFS system use the cryptographic mechanism to achieve the confidentiality of the data. E-IPFS works on top of the IPFS protocol and the access control mechanism not depend on the any other system or third party.

By the analysing research experimental results in the evaluation it concluded that proposed E-IPFS system can use to share the private data with confidentiality with cost of time for the additional process.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Dr. Kasun De Zoysa senior lecturer of the University Of Colombo School Of Computing – UCSC for the continuous support of my research work, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also goes to my Co-Supervisor Mr. Kenneth Thilakarathna, lecturer of the University Of Colombo School Of Computing – UCSC for his feedback provided on research works, which helped me to look at different perspectives of my research and improve the study and un-ceasing guidance, constant supervision and providing valuable resources throughout my masters degree project.

I would also like to thank our research project coordinator Dr. L.N.C. De Silva senior lecturer of the University Of Colombo School Of Computing – UCSC for her guidance given throughout the year. In addition, I want to show my gratitude for the university staff and all the lecturers for the support that given to complete this research successfully.

Last but not the least; I would like to thank my family and my friends for supporting me throughout the research work and my life in general.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Acl-IPFS - Access Controlled IPFS

ACM - Access Control Manager

AES - Advanced Encryption Standard

AES - Cryptosystem (Advanced Encryption Standard)

DAG - Directed Acyclic Graph

CSV File - Comma-separated Values (CSV) File

DHT - Distributed Hash Tables

E-IPFS - Extended Version of the Inter Planetary File System

Gen - Generation

HDFS - Hadoop Distributed File System

HTTP - Hypertext Transfer Protocol

IPFS - Inter Planetary File System

IPNS - Inter Planetary Name System

Merkle DAG - Merkle Directed Acyclic Graph

P2P system – Peer-to-Peer system

Q&A system - Questions and Answers system

RAM – Random Access Memory

RSA - Cryptosystem (Rivest - Shamir - Adleman)

SFS - Self-Certified File systems

SLA - Service Level Agreement

UML - Unified Modeling Language

VRM - Virtual Resource Manager

Chapter 1: Introduction

This chapter outlines the motivation (section 1.1) and the aims and objectives (section 1.2) of the research. The section 1.3 describes the significance and scope of this research and provides definitions of terms used. Finally, section 1.4 includes an outline of the remaining chapters of the thesis.

1.1 MOTIVATION

Distributed files sharing mechanism are now getting poplar in the web users around the world. Through these file distributed sharing systems, it unable to control the access levels of the shared contents because of its distributed manner. By the literature review, it identified the barrier for implement the access control in a distributed file sharing system. In this research it propose a cryptographic mechanism to control the access of the shared contents through a distributed file sharing protocol of Inter Planetary File System Protocol (IPFS) [3].

1.2 AIMS AND OBJECTIVES

The current web application systems can be divided into three main categories. They are centralized web, decentralized web and distributed web. Among these, the distributed file systems are becoming a new trend in computer system domain. High availability is a major advantage in the distributed systems. In that sense, even upon server or disk drive failures the system will be available. Low latency, location independency, scalability, increased scalability and Interoperability are the other benefits in distributed systems [4].

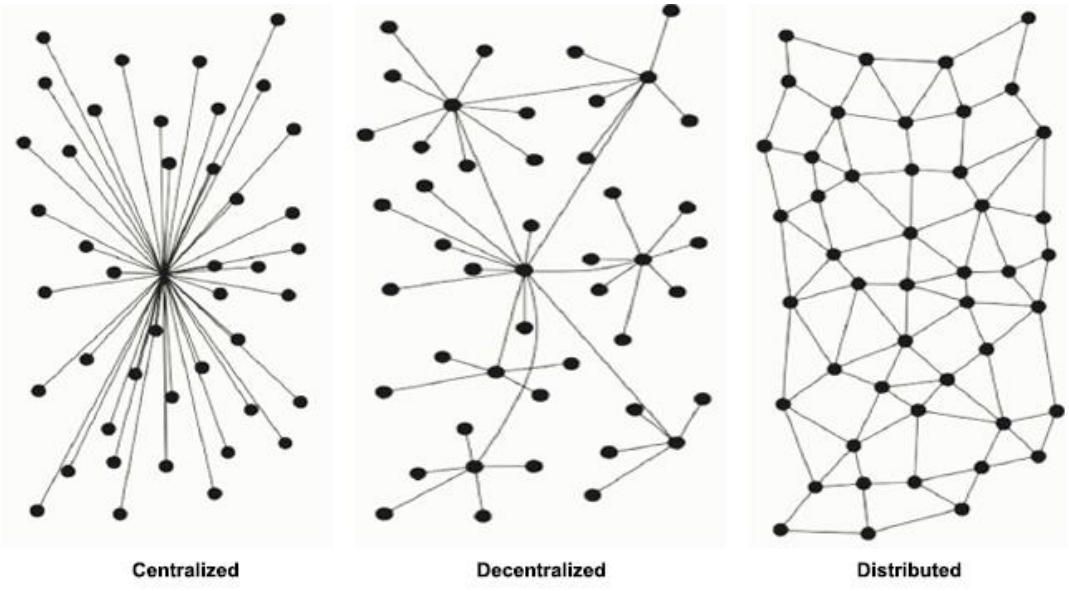


Fig 1.2.1 Centralized, Decentralized and distributed web

The Figure 1.2.1 shows the architecture pattern of current web systems. A centralized web system has one master node which is connected to all other nodes [5]. This master node govern the entire network and it has fully control over the data. In decentralized systems, there are master nodes. However, every node makes its own decisions and finally collaborate all of the individual nodes. Decentralized systems do not have one central owner to their network. A distributed system has equal level of nodes and every node makes its own decision. Every node is connected to the other surrounding nodes that are in nearby. This type of networks are not depend on a master node. Nevertheless, network failures will be minimum [5].

Most of the time distributed and decentralized file systems store the files in its clients' machines. Then it capacious for share the stored files throughout its network. A node request comes to the network; the peers are liable for seed the files that are stored in their local machines. There are many services, which are built using distributed and decentralized protocols for sharing files. According to the internet, statistics the peer-to-peer networks takes approximately 40% to 70% of all Internet traffic in the world [6]. Bit-Torrent [7] is the most popular file sharing system that use to share files free of charge through users over the internet [8]. It breaks the files up into small chunks and make platform to download from multiple nodes without the need for a central server.

Inter Planetary File System Protocol (IPFS) is a peer-to-peer hypermedia protocol which is developed by Protocol labs [9]. It is designed to create a network of peers as the distributed file system which seeks to connect all computing devices with the same set of files [3]. By considering the available distributed systems they does not have an efficient and in built distributed access level control mechanism. This research introduced novel extended version of IPFS protocol called “extended IPFS” (E-IPFS) which uses cryptographic mechanism, that can share private content with the control of uses’ access levels.

By this research, it proposes a new mechanism to perform access control (or capability) system for private content. Publishers need to be able to restrict access to enable use-cases like Dropbox [10] without involving centralized third party.

1.3 SIGNIFICANCE, SCOPE AND DEFINITIONS

In this research, it comparatively studied the existing distributed and decentralized file sharing mechanisms, how these distributed systems and protocols grant access levels for users. According to the study, it concluded that there is no distributed way to grant the access levels in distributed file sharing mechanisms. These mechanisms have centralized or decentralized authorization systems, which can consider as special nodes that provide authorization and grant the access levels to the entire system. When the authorization nodes are not online, these systems are unable to grant the access levels for shared contents.

In this research, it covers the area distributed web protocol called Interplanetary File System (IPFS). It is based on several technologies; Distributed Hash Tables (DHTs) to coordinate and maintain metadata, BitSwap (Bit Torrent inspired protocol) data exchange protocol to distributing pieces of files to each other, Version Control System (Git) for representing immutable objects files as Merkle Directed Acyclic Graph (Merkle DAG) and Self-Certified File systems (SFS) for distributing the trust chains. Furthermore, it identifies the different between current web and distributed web, other protocols and services that build on IPFS and access level control mechanisms in decentralized and distributed web protocols.

1.4 THESIS OUTLINE

This thesis has six chapters, references and appendices sections. Chapter 1: give the introduction for the research area. Chapter 2: is the Literature review of the research. It contain comparative study about the research area. Research design describes in Chapter 3: and it mention the methodology of the research. Chapter 3: describes the research design with the proposed solution architecture. Research evaluation and experimental results are in Chapter 4: and Chapter 5: describes the conclusion of the research and new area that open through this research.

Chapter 2: Literature Review

2.1 INTRODUCTION

In this chapter, it begins with a background (section 2.2) and reviews literature on the following topics: BitTorrent (section 2.3) which is most popular peer to peer file sharing system in the world; Blockchain-Based, Decentralized Access Control for IPFS (section 2.4) is a modified version of the IPFS that uses Ethereum smart contracts to secure the private data; Hadoop Distributed File System (section 2.5) which designed for storage the less number of large data files rather than the huge number of small data files; Interplanetary File System (section 2.6) peer-to-peer hypermedia protocol designed on the principle of permanent web; Open Peer-to-Peer Systems over Blockchain and IPFS (section 2.7); Distributed Decentralized Data Storage Using IPFS; (section 2.8) which is an application that can store and retrieve data from the IPFS; A Search Engine Based on IPFS (IPSE) (section 2.9) is a protocol that can use to search the contents in IPFS; and Distributed Access Control in Cloud Computing Systems (section 2.10). Section 2.11 highlights the implications from the literature and develops the conceptual framework for the study. Finally, the problem definition of the research is describes in the section 2.12.

2.2 BACKGROUND

This section reviews the related works of file sharing mechanisms, which are associated with distributed systems. Each of these describes how it relates with this research, how far it comes to this research question and what approaches it used. Furthermore, it mention the identified gap between the research question and each approaches.

2.3 BIT TORRENT

BitTorrent [11] is the most popular distributed file sharing system over the Internet. It enables fast downloading even for the large files using minimum Internet bandwidth. BitTorrent spread the file chunks (smaller pieces of the file) through its networks. These file holders named as “seeds”. The file transfer load is distributed between the users/computers, which are called “peers”. Anyone with the torrent file

can download or get that file by using a software application called a “client” that hosts a tracker. The tracker conducts the search and keeps the data record of peers provide pieces of file copies to download for peers [11].

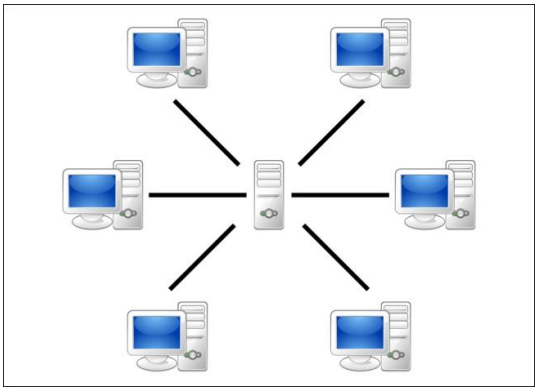


Fig 2.3.1 Centralized service share files [12]

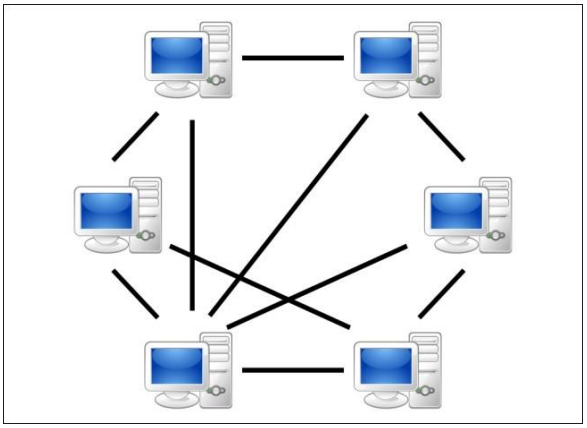


Fig 2.3.2 Bit-torrent swarm share files [12]

Centralized servers share the files from the central server as shown in Figure 2.3.1. The central server is facing a huge traffic and has to be sufficient network bandwidth because if all users dealing with it. However, the Torrent has not centralized servers where the users can download files. As shown in Figure 2.3.2 all the files are copied from other users which are called seeds who have pieces of the file. Peers are the people who are also downloading the file. Once peers get a piece of a file, it may also start to seed to the network. Seeds are copies of the entire file that has been placed. Then the other downloaders have a source. Once the download is complete the peer is also become a seeder. It can also share for the network. Now it works as a P2P system. In additionally peers can survive without the seed. Peers can share pieces among each other until the original file is completed.

Bit-torrent can send and receive requests for multiple nodes [11]. Therefore, most of the users are using Bit-torrent because of its efficient use of network bandwidth. In present, Bit-torrent use to share copyright protected materials sharing. There is an active legal campaign against it. Even though Bit-torrent support for effectively share files, there is no any access level control mechanism built in the BitTorrent protocol. Anyone how has torrent file can download the file content without file owner's restrictions.

2.4 BLOCKCHAIN-BASED, DECENTRALIZED ACCESS CONTROL FOR IPFS

Blockchain-Based, Decentralized Access Control for IPFS (Acl-IPFS) [1] – This research present a modified version of IPFS that can share the private and sensitive data using IPFS and Ethereum smart contracts. It identified that IPFS is a more efficient way to store and share large data with respect to Blockchains, which are list below.

- Storing large content files on the blockchain is inefficient.
- Blockchain becomes bloated because the data being replicated on a large amount of nodes.
- Large size nodes mining will become more expensive.
- Require higher bandwidths and more storage spaces to bestow the blockchain.

It takes InterPlanetary File System (IPFS) for store large file. Files transfers through the IPFS cryptographic hashes of their contents. In [4 p.1499] it describes that "IPFS does not permit users to share files with selected parties. This is necessary, if sensitive or personal data needs to be shared". Therefore, IPFS has not in-built mechanism to share files in selected parties.

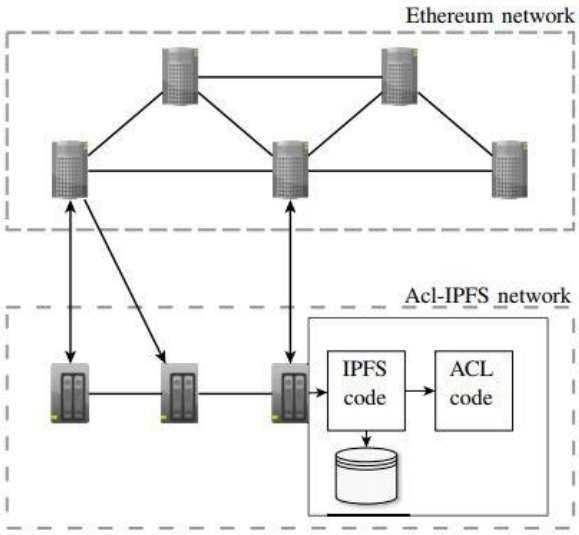


Fig 2.4.1 The architecture of the Acl-IPFS network [1]

The presented version is called as "Acl-IPFS" and it uses Ethereum smart contract [2] is used to maintain the access control of the shared file in IPFS. Ethereum blockchain handle the access level controls and IPFS store the file contents. Every file access is verified by a block in the Ethereum smart contract. In other words there is a list of permissions that are stored in the Ethereum block chain that gives the permission to the requested object in IPFS network. It does not describe the exact reason for select Ethereum blockchain throughout popular blockchains.

This research also evaluate the proposed system vs original IPFS in several based on latency and stress by using an experimental setup. It founds that Acl-IPFS is seconds slower than the original IPFS operation [4 p.1504]. According to the conclusion of the research, operation delay reason is that Acl-IPFS depend with Ethereum (another network). In this research, it propose an access level control mechanism, which operates entirely inside the IPFS network.

2.5 HDFS (HADOOP DISTRIBUTED FILE SYSTEM)

HDFS (Hadoop Distributed File System) [13] is a popular distributed file system that can be used for file sharing. This is an alternative solution for store big data without centralized server. HDFS store the massive amount of data by a distributed approach. An HDFS has a manage node for its cluster data called "NameNode". There are nodes in various machines that store data by broken down into smaller chunks. The

default size of one block (chunk) is 128MB. Not only this, it also makes copies of this data and uses these copies if one machine fails.

HDFS has a replication method for overcomes the issue of DataNode. It copies data through the network DataNodes. HDFS workload is immediately transferred to another node and avoid failures. In other words HDFS uses these copies if one machine fails. These nodes can be easily accessed by for data storage and processing, HDFS uses the power of the whole cluster. This is an advantage compared to other distributed data storages.

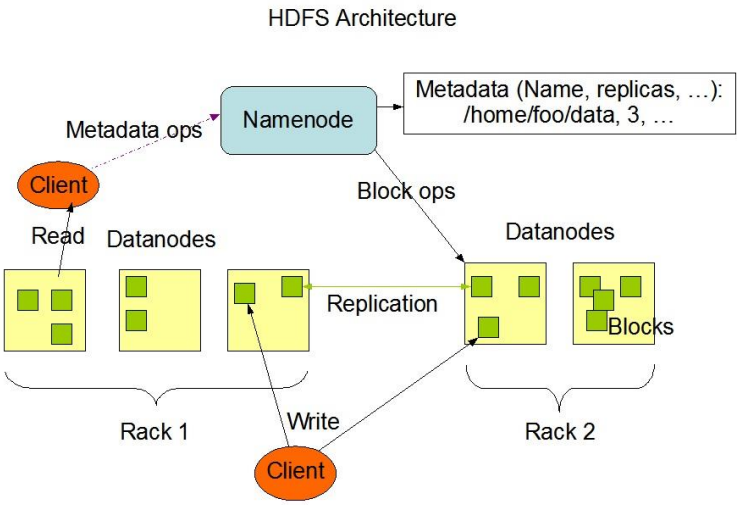


Fig 2.5.1 HDFS Architecture [14]

An HDFS cluster is contains of a NameNode. It can manages the cluster metadata, and DataNodes that store the data. Inode is the node that represent the files and directories in the HDFS. It can record the additional parameters like permission, modification and access times. HDFS has already built is access level mechanism. But it regulate clients' access by 'NameNode' [14]. The client first requests the NameNode to read the Data. The NameNode allows the users to read the requested data from the DataNodes. The data are read from the DataNodes and sent to the client. In other words, NameNode is centralized manage and maintain the slave nodes (data nodes: which actually data holds on). Then HDFS cannot handle the distributed system without this centralized node. Then HDFS cannot be considered as a pure distributed system.

2.6 INTERPLANETARY FILE SYSTEM (IPFS)

Interplanetary File System (IPFS) [3] is the modern approach to solve the above problems in the client-server model and the HTTP web. IPFS is a peer-to-peer hypermedia protocol that stores the file in a distributed manner. It uses content-addressing rather than the location-addressing like in Hypertext Transfer Protocol (HTTP). IPFS loads contents faster by using less bandwidth, deduplication data [3]. This protocol uses Distributed Hash Tables (DHTs) to coordinate and maintain metadata, BitSwap data exchange protocol to distributing pieces of files to each other, Version Control System (Git) for representing immutable objects files as Merkle Directed Acyclic Graph (Merkle DAG) and Self-Certified File systems (SFS) for distributing the trust chains. Figure 2.6.1 shows the architectural structure of the IPFS protocol.

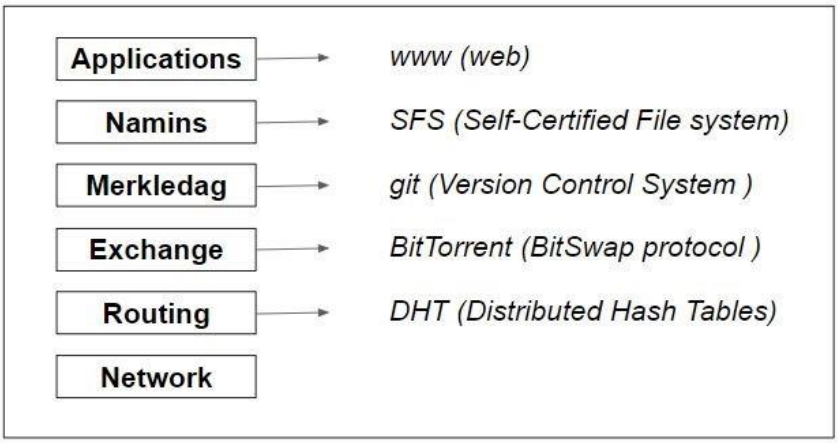


Fig 2.6.1 IPFS architectural stack

IPFS uses distributed hash tables for store hashes with IPFS objects hashes. IPFS objects are the small chunks that use for store files by breaking them and each object can store 256 kb of data. Files that are less than 256 kb create a one IPFS object.

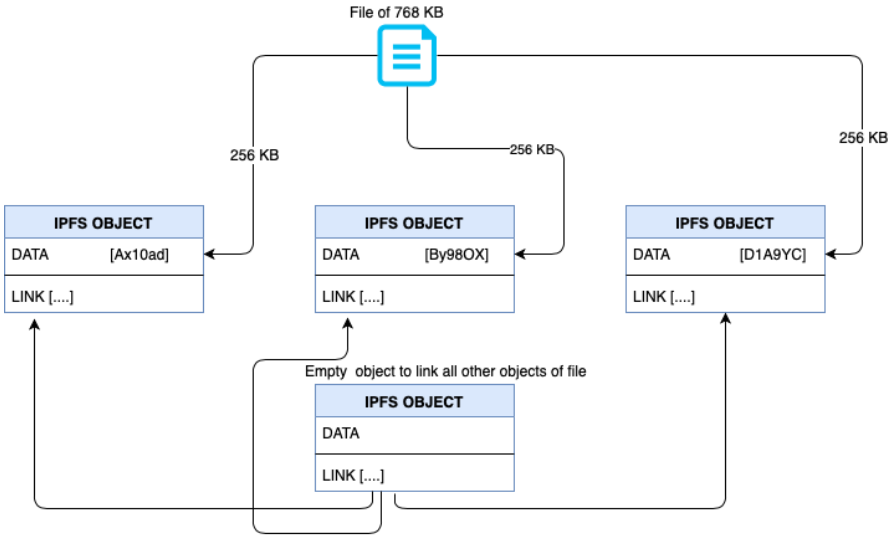


Fig 2.6.2 IPFS objects structure [15]

When a file is large, it creates a main object that would link to all the objects that make up that file as shown in Figure 2.6.2. These IPFS objects are accessed by its hash value. Further IPFS routes the request to the content using this hash values. This has value depend on contend that in the IPFS object [1 p.3].

IPFS is mainly focus on the permanent-web approach. It has many beneficial own properties over the current web. Given files are not changed because of match the hash ensure that integrity property of the system. In fact, everything is addressed by a hash value, IPFS has deduplication property.

Any user in the network can access any of the IPFS content or data, which shared in the network. The only constraint is that one of a live node in the IPFS network that has to contain that requested data. Further the connected IPFS node or deploy an IPFS node able to request and viewed any of the data that shared in the IPFS network. There is no central authority in IPFS. In that sense, IPFS has not in-built access level control mechanism.

2.7 OPEN PEER-TO-PEER SYSTEMS OVER BLOCKCHAIN AND IPFS

Open Peer-to-Peer Systems over Blockchain and IPFS [16] is an Agent-Oriented Framework. This system can reveals data access, discovery and trust in peer-to-peer systems. In additionally it proposed the distributed architecture for these open systems.

This architecture can give guidelines to decide in which cases Blockchain technology may be required. In other words this work presents a framework to build peer-to-peer open systems as a multi-agent systems.

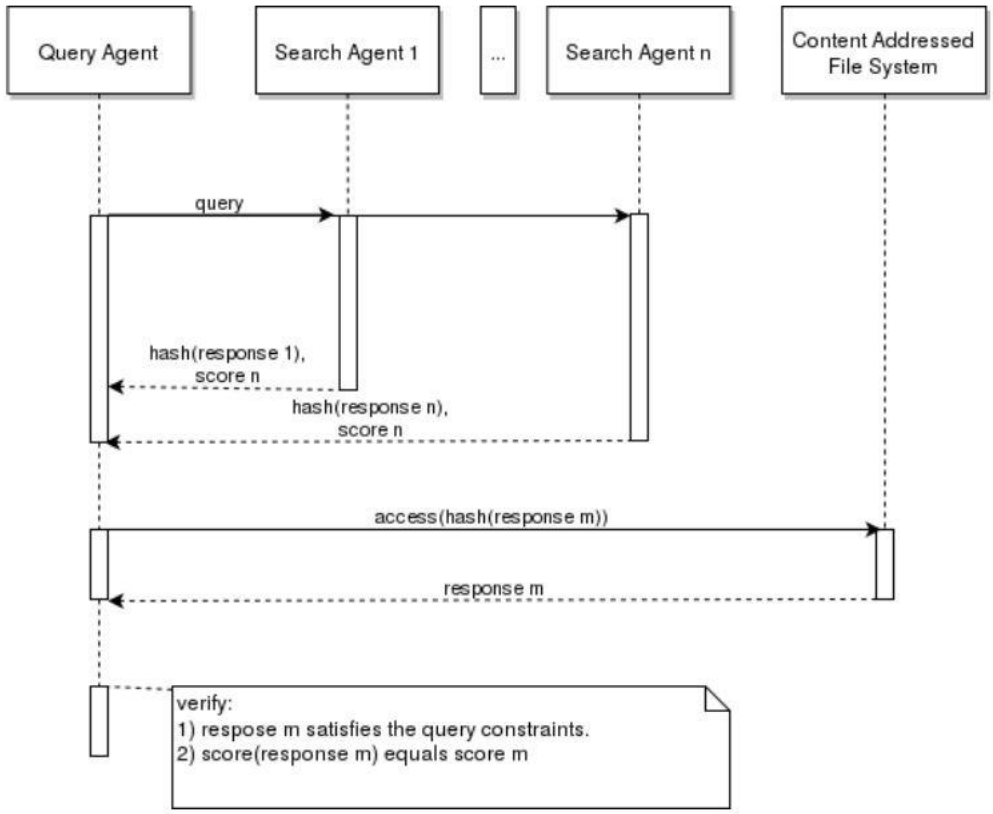


Fig 2.7.1 Distributed discovery protocol - UML sequence diagram [16]

As shown in Figure 2.7.1 (UML sequence diagram) this system use agents for performing each action which are in a distributed system. As an example, it takes simple Questions and Answers (Q&A) system. By using this Q&A system, it describes the difference between content-addressable distributed systems with respect to the traditional centralized Q&A system [9 p.3].

According to this research it can identified that how to provide security levels and protect the privacy of the data in distributed systems. It has identified the limitations and challenges that are faced in the distributed technology as well [9 p.5]. This multi-agent model is depend on two distributed systems, IPFS and the Blockchain. The new E-IPFS system is only depend on the IPFS network then there will be a high performance and efficiency.

2.8 DISTRIBUTED DECENTRALIZED DATA STORAGE USING IPFS

Distributed Decentralized Data Storage Using IPFS [17] - This research implements a free data storage application that can store and download files using Apache, PHP, and MySQL on top of the IPFS. Object is to develop a platform that can used as online Drive storage, but using decentralized system. That is free drive storage and data will permanently store there until 51% nodes of IPFS are shutdown. This runs on top of the IPFS network. It keeps a local MySQL database for store data file names and the corresponding hash values form the IPFS network.

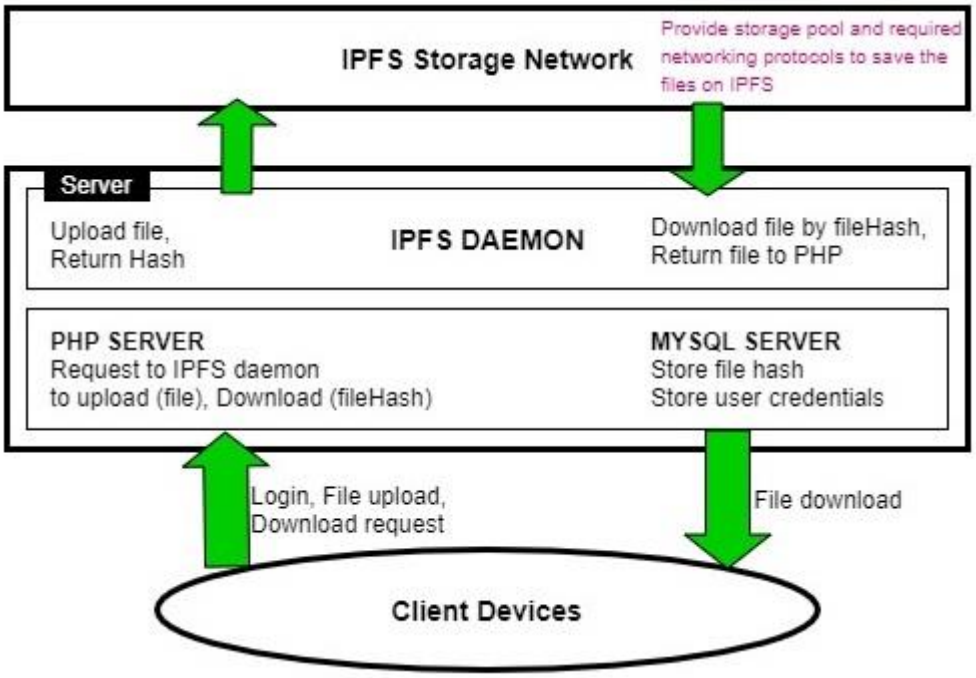


Fig 2.8.1 Decentralized storage architecture [17]

This has only a search mechanism for the reference name, which is stored in the MySQL relational database. As its structure data privacy can be achieved by encrypting the data before uploading to storage [10 p.1641]. In other words, it achieves the confidentiality of the data by encrypting from the client-side. It will protect the data by anyone who gets it from IPFS network using the hash value that is stored in MySql database locally. Major disadvantage of this system functions are totally depends on the locally stored components. The proposed E-IPFS system is not depending on any other parties.

2.9 IPSE: A SEARCH ENGINE BASED ON IPFS

IPSE: A Search Engine Based on IPFS [18] – This research introduced a new protocol on top of IPFS, which is called as Inter planetary search engine (IPSE). That maps hash addresses which are identifiers of IPFS objects, to semantic tags that can understand by a human. Users can share contents to IPFS network and add tags to complete data indexing. Users can find contents by retrieving relevant tags.

Fig 2.9.1 Create tags for the IPFS object hash address [18]

As shown in Figure 2.9.1, this creates an appropriate tags for each IPFS object hash addresses. This tags represent the contents which are related to the IPFS object. In other words these tags can give a sense of objects contents without reading the internal data. In this research it ensures the users personal data privacy and protection through the blockchain technology which is service by IPSE called application-specific blockchain [11 p.22].

This system share the contents and add tags for a mining process. This mining process generates the Token rewards [11 p.5]. Then the SuperNodes can perform the search functionality by using these storage index data. This distributed ecosystem unable to restrict search contents, which are shared with selected users. This system does not add any access level control mechanism.

2.10 DISTRIBUTED ACCESS CONTROL IN CLOUD COMPUTING SYSTEMS

Distributed Access Control in Cloud Computing Systems [19] is a research that focuses on the make an effective access control mechanism for distributed applications, in collaborative, distributed, cooperative environments. There is a vital study about how the cloud services handle the users' access rights. Cloud service can be considered as distributed computer paradigm, because of the users can access the resources, share the resources and get the provided services hosted by using its cloud service.

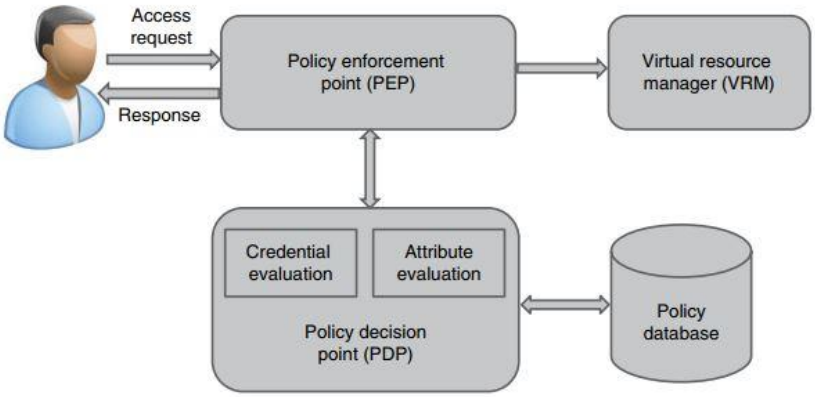


Fig 2.10.1 Access control manager (ACM) architecture [19]

The cloud service authorization consists of virtual resource manager (VRM), access control manager (ACM) and service level agreement (SLA) [12 p.420]. By using this architecture, it propose several access control models like role based, attribute based, risk based etc. These models ensure the confidentiality, integrity and availability of the data, which shared in the system. Furthermore, it uses standard encryption mechanism such as SSL/TLS for the security of the data.

In this research, various access-control policies and models were analyzed for the distributed access control. These models can modified and applied for the distributed systems level as well. It gave a motivation for design an access level control mechanism in E-IPFS system.

2.11 SUMMARY AND IMPLICATIONS

	Distributed system / research	**Identified key features and limitations with respect to research problem**
1	BitTorrent [11]	● No access level control mechanism ● Anyone with the torrent file can get the content ● Node level data duplication is happen
2	Blockchain-Based, Decentralized Access Control for IPFS [1]	● Designed on top of IPFS protocol ● Access level control depend on Ethereum blockchain ● Identified more latency and stress rather than the original IPFS
3	HDFS (Hadoop Distributed File System) [20]	● Has a centralized node for handle authentication and authorization
4	Interplanetary File System (IPFS) [3]	● Pure distributed system & No centralized control ● Node level data deduplication ● No in-build access level control mechanism
5	Open Peer-to-Peer Systems over Blockchain and IPFS [16]	● Build using IPFS protocol ● Depend on the Ethereum Blockchain ● High latency with respect to normal IPFS
6	Distributed Decentralized Data Storage Using IPFS [11]	● Build on using IPFS protocol ● User has to maintain centralized application and system availability and authentication depend on it
7	IPSE: A Search Engine Based on IPFS [18]	● Build on using IPFS protocol ● Unable to restrict search contents which are shared with selected users
8	Distributed Access Control in Cloud Computing Systems [19]	● Solution for the distributed cloud services ● Multiple designed for grant access level in distributed manner
9	XtreemFS [21]	● Has server-client architecture ● Authentication and authorization depend on the ‘server nodes’

Table 2.11.1 Summary of findings of Literature Review

In distributed systems has many there are advantages with compere to the centralized web systems. Location independency is one of a main advantage in a distributed system. IPFS (Interplanetary File System) [22] is addressing and retrieving the contents by content addressing mechanism. It does not depend on the location where the original content is located. Low latency and high availability are the other advantages of a distributed system. The content that are requested by most of users are highly spread through the network nodes and the content will be high available for new user. The requested contents will be serve form the nearest available node with low latency. Distributed system can easily scale up according to the requirement with the minimum configuration. Then these systems has scalability property. The node level data duplication is not happen in the IPFS network. Therefore, it prevent the unwanted data replication in the nodes.

According to the literature review of the research, most of the available distributed file systems use the authentication nodes for manage the authentication for shared contents. Without these nodes this systems unable to manage authentication. In other words, these systems not operate as the pure distributed way. In several systems, it uses the blockchain to manage the authentication of the shared contents. This method is add a dependency on the another system for the distributed system.

Apart from that, controlling access levels the system has ensure the Information security parameters. They are confidentiality, integrity, availability, non-repudiation and authenticity [23]. Confidentiality ensure that data is not disclosed to the unauthorized persons. Integrity maintain the accuracy of the data and not edited in an authorized way. Availability is the factor that the data is available when it wants to the user. Non-repudiation means that third party cannot deny the access to the data. In Cryptography, the digital signatures ensure the non-repudiation property of the data. Authenticity means that user can verify the data sender and ensure that data comes from the trusted source.

There it can identify the research gap. Even these distributed systems have benefits; it does not have in built distributed access level control mechanism for share the private contents through these networks. In this research, it takes the IPFS (Interplanetary File System) distributed network for the implementation. IPFS has not in-built access level control mechanism for its contents. Then the research is focus into

how the content access control can be implemented within IPFS distributed file system.

2.12 PROBLEM DEFINITION

An IPFS has not in-built access level control mechanism for its contents. How content access control can be implemented within IPFS distributed file system?

Chapter 3: Research Design

3.1 INTRODUCTION

This chapter describes the design of the research. Furthermore, the how the given design can achieve the aims and objectives stated in section 1.2 of Chapter 1. The section 3.2 discusses the methodology that used in the research, the stages by which the methodology was implemented, and the research design; section 3.3 outlines the procedure that used of the research; section 3.4 discusses how the data was analysed; section 3.5 discusses the research problems and limitations. Finally, the proposed solution of the research is describes in section 3.6.

3.2 METHODOLOGY AND RESEARCH DESIGN

3.2.1 Methodology

The main goal of this research is to design and create new model for grant access levels in IPFS distributed shared network. By considering research methodologies, this research is considered as a design science research.

3.2.2 Research Design

According to the process of design science research framework, the research is going to proceed in constructive approach. In the constructive approach, there are five main step to follow.

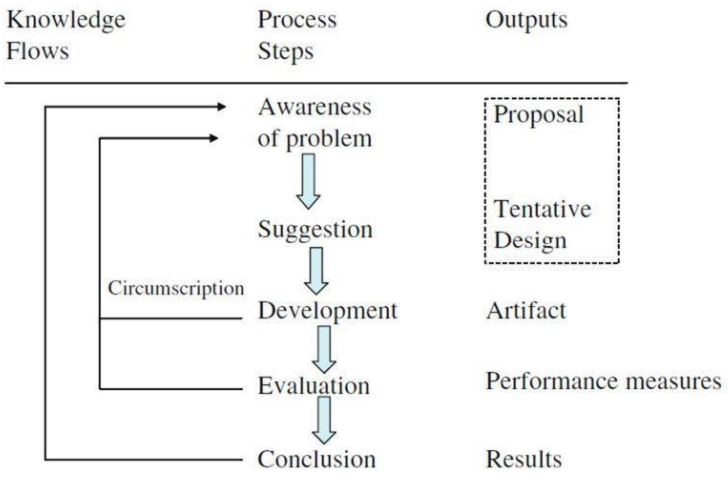


Fig 3.2.1 Design Science Research Framework [24]

First step is the analysis, which is use to recognition of the research problem and determine research goals. Second step is the suggestion. It uses to discuss what kind of process or a system might solve the above-identified problem. In the development step, it develop or construct artefact with recognized methodologies and justify solution. Fourth step is for the evaluation, which validate the final approach with respect to the research goals. Finally, research end with the conclusion.

3.3 PROCEDURE

According to the research problem, it has to design system for share the content in IPFS network and it should only be readable for those who gave access to it by the owner of the content. By considering IPFS architecture, there is no central authority. Then the access levels cannot grant form central authority to govern this process. Therefore, in this research it propose a cryptographic implementation to secure the contents, which are shared in IPFS network.

In the Analysis, it describes the recognition of the problem and determine research goals. In first, identify that theoretical design and technological implementation background and the benefits in IPFS protocol. Then clearly identify limitations in IPFS protocol for share private contents for selected parties. Identify how related researches give solutions for the related research question and drawbacks of them.

In this suggestion step, it is discussing what kind of system might solve the problem. Then develop the artefacts with recognized methodologies, Justify solution, and differentiate from known solutions. After it, design a new model for overcome above issues and address the research question with necessary assumptions.

In the evaluation process, it validate approach with respect to research goals by checks whether the proposed system solves the problem, analysing its strengths and weaknesses. Moreover, it checks the limitations and evaluate the results of the proposed mechanism with respect to research problem. Furthermore, it identify further enhancements and future works of this research.

Finally, according to the evaluation, the research has to end up with the conclusion.

3.4 ANALYSIS

The distributed system has at least two parties to share the contents. Therefore, it can identified that there is two main functions. They are data adding and retrieving. For that it can use the data adding time, data retrieving time, no of revivers and size of the data are the parameters for the analysis. In this research, it uses the mechanism to analysis of above two parameters in the experiments by fix other parameters in the constant state.

3.5 ETHICS AND LIMITATIONS

When generating the data sets there can be an unexpected outlies which can highly effect for the results. Therefore, in this experiments repeat the same process, get three datasets, and calculate average for time calculations. Apart from that, to reduce other factors like processing speed, it uses identically same computers for data collection. Before each experiment perform the IPFS nodes are refresh and reinitialized both computers. Additionally, both computes maintain in their ideal state without running any other applications.

3.6 PROPOSED SOLUTION DETAILS

According to the research problem, it has to design system for share the content in IPFS network and it should only be readable for those who gave access to it by the owner of the content. By considering IPFS architecture, there is no central authority. Then the access levels cannot grant form central authority to govern this process. Therefore, in this research it propose a cryptographic implementation to secure the contents, which are shared in IPFS network.

This describes the three phases of the proposed solution, which named as extended version of IPFS (E-IPFS). In first phase it discuss that generate private and public key pair for the user and share it in IPFS network. In second phase of the proposed solution, which is add encrypted content to IPFS. Then it discuss the third phase of the proposed solution, which share the key with users and retrieve shared content from the IPFS.

3.6.1 Generate kay pair and share public key in IPFS

System design can divide into three main phases. In the phase one, it generate own private and public key pair for a user and share the public key in IPFS network. In the proposed system, the first step is user has to generate own public and private key pair by using RSA algorithm [30]. This RSA key pair generated with the size of 2048-bit which is the current stranded. After generate this key pair, users can use them for E-IPFS system in any computer by keep it. The public key has to share with through the IPFS network.

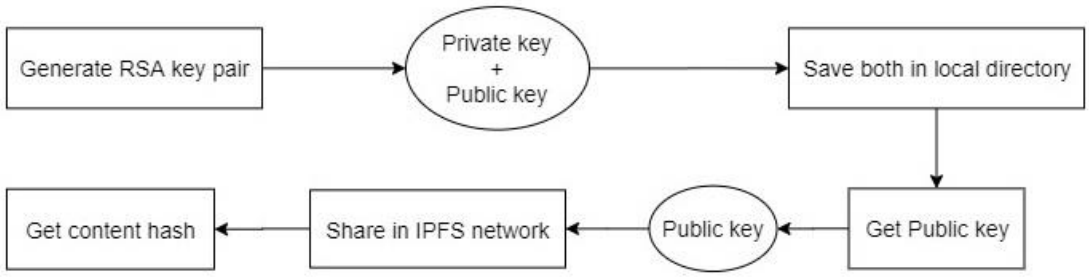


Fig 3.6.1 Generate RSA key pair and share public key in IPFS

```
Keys <- GenerateKeys('algorithem',{'private key options', 'public key options'})
WriteToFile(Keys['publicKey.pem'])
WriteToFile(Keys['privateKey.pem'])
PublicKeyHash <- IPFSAdd('publicKey.pem')
end process
```

Fig 3.6.2 Pseudocode for generate RSA ker pair and share public key in IPFS

3.6.2 Add encrypted content to IPFS

In the phase II, it add the contents to the IPFS network using a random generated symmetric key by the E-IPFS system. This is a symmetric key and receiver can read the content using that key. Then the shared content is stored by converting it into cypher text and secured in cryptographic way.

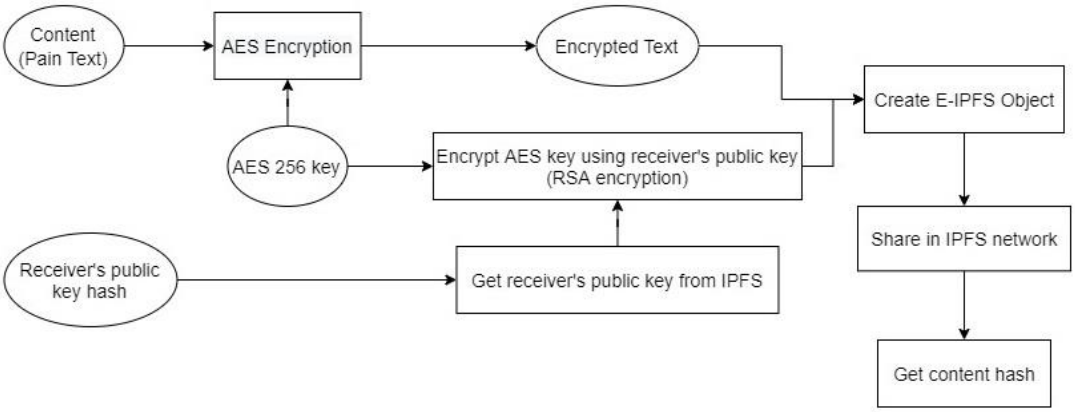


Fig 3.6.3 Add Encrypted Content to IPFS

Figure 3.6.3 describe that phase II of the research implementation. First, the content (plain text) that want to share within IPFS network is encrypted by a symmetric key. Figure 3.6.4 shows that pseudocode for add the encrypted data to IPFS.

```
PlainText <- get from user input
RecieversPublicKeyHash <- get from user input
AESKey <- GenerateAESKey('keySize')
EncryptedText <- AESEncryption('PlainText', 'AESKey')
EncryptedTextHash <- IPFSAdd('EncryptedText')
RecieversPublicKey <- IPFSGet('RecieversPublicKeyHash ')
EncryptedAESKey <- RSAEncryption('AESkey', 'RecieversPublicKey')
EIPFSObject <- CreateEIPFSObject('RecieversPublicKeyHash', 'EncryptedTextHash', 'EncryptedAESKey")
EIPFSObjectHash <- IPFSAdd('EIPFSObject')
end process
```

Fig 3.6.4 Pseudocode for add the encrypted data to IPFS

In this process is uses AES (Advanced Encryption Standard) [31] algorithm for the symmetric key encryption. It uses the size 256-bit key, which is the current stranded. This encrypted data shared to the IPFS network. The encryption mechanism

guaranteed that shared information could not read and understand by other users because it is in encrypted format. Then it has to feed the receiver's public key path, which shared in the IPFS. E-IPFS system get the public key of the receiver, encrypt the symmetric key using receiver's public key, and create new object called E-IPFS object.

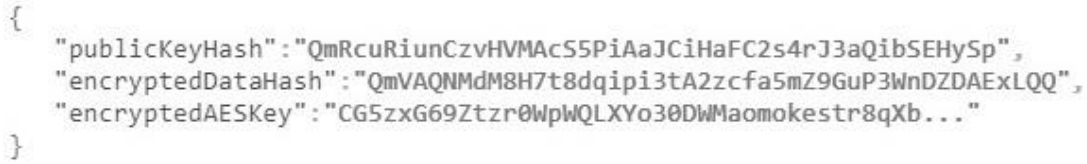


Fig 3.6.5 Structure of the E-IPFS object

E-IPFS object contains the three parameters. The "publicKeyHash" contains the public key of the receiver that shared in the IPFS network. The "encryptedDataHash" is the IPFS path that holds the encrypted data shared in the IPFS network. The "encyptedAESKey" parameter contains the symmetric key, encrypted by the receiver's public key that use to encrypt original data. The sample of an E-IPFS object is shows in the Figure 3.6.5. This E-IPFS object use to share the details to the receiver in proposed system.

In the implementation, there use "publishDataToEIPFS.js" for the above process. This script accept the user input data (content) and the receiver's public key as the parameters. Then it return the E-IPFS object shared hash path in IPFS which receiver has to use to access the shared data.

3.6.3 Retrieve shared content from IPFS

In the third phase, it propose a system to retrieve the symmetric key and read the shared content. In first, it get the E-IPFS object form the IPFS network by its path. Then check the user's public key hash ("publicKeyHash" parameter). If it match, the process will continue. Otherwise, the process will terminate.

One the user's public key match the system get the encrypted data from the IPFS network by the use of "encryptedDataHash" parameter in the E-IPFS object. Then it get the AES key by the "encyptedAESKey" parameter of the E-IPFS object and decrypt it user's private key. After that, E-IPFS system can get the original content

(plain text) by decrypting the encrypted data using AES key. Figure 3.6.6 shows the process that receiver get E-IPFS object and read the shared data.

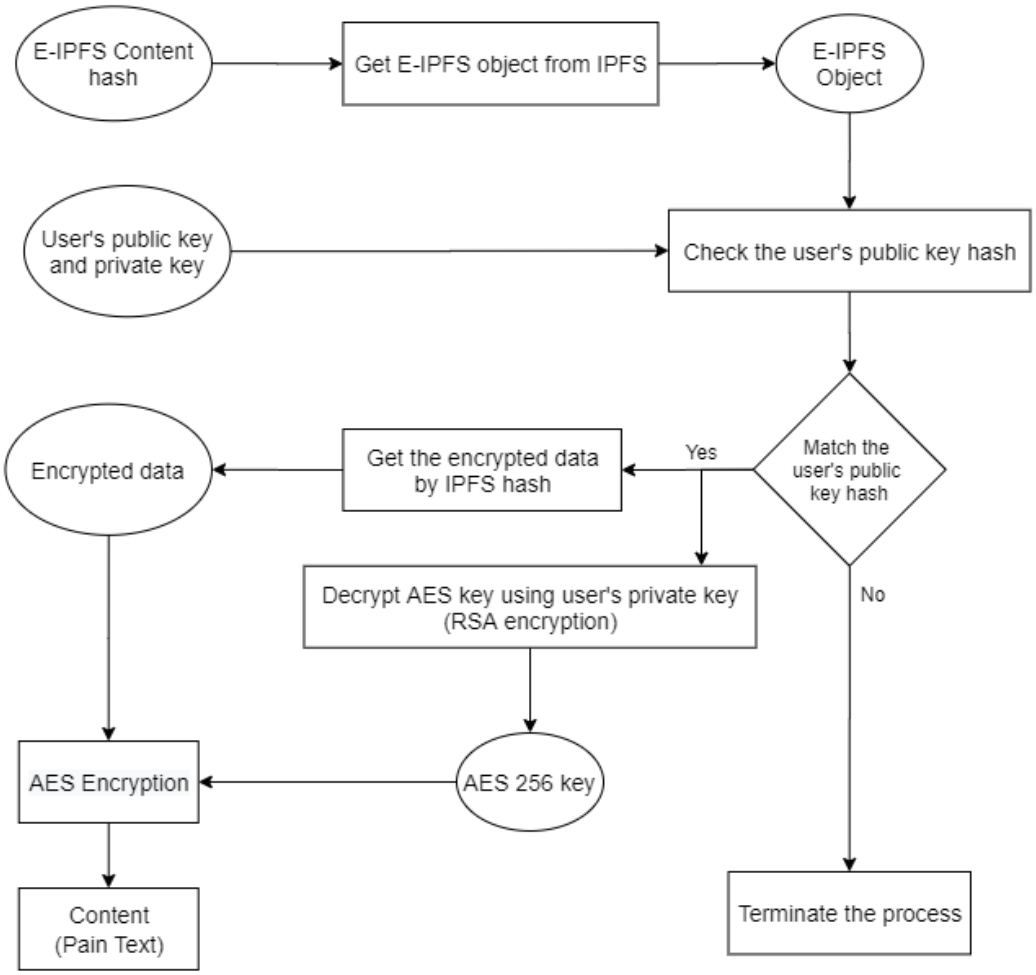


Fig 3.6.6 Retrieve shared content from IPFS

In the implementation, there use “getDataToEIPFS.js” file script and it accept the E-IPFS object path. After the above process is completed, it return the original data (content) that shared for the user.

Figure 3.6.7 shows that structure of the pseudocode for the retrieve shared data from IPFS.

```
UserPrivateKey <- get form file path (private.pem)
UserPublicKey <- get form file path (public.pem)
EIPFSObjectHash <- get from user input
EIPFSObject <- IPFSGet('EIPFSObjectHash')
RecieversPublicKeyHash <- get from EIPFSObject
EncryptedTextHash <- get from EIPFSObject
EncryptedAESKey <- get from EIPFSObject
if(RecieversPublicKeyHash == HASH of (UserPublicKey))
        AESKey <- RSADecryption('EncryptedAESKey ', 'UserPrivateKey ')
        EncryptedText = IPFSGet('EncryptedTextHash')
        PlainText<- AESDecryption('EncryptedText', 'AESKey')
        end process
else
        end process
```

Fig 3.6.7 Pseudocode for retrieve shared data from IPFS

Chapter 4: Evaluation and Results

4.1 INTRODUCTION

This chapter describe the research evaluation criteria in section 4.2 and section 4.3 mention the instruments and software that used in the research. The results, which are used to evaluate the proposed solution of the research describes in section 4.4.

4.2 EVALUATION CRITERIA

In order to extend IPFS with an access control mechanism, the following information security requirements need to be considered. They are confidentiality, integrity, availability, non-repudiation and authenticity.

- Confidentiality

The proposed system of this research ensure the confidentiality by the cryptographic mechanism. It uses AES 256 bit key for encrypt data before it shared to the IPFS network. The 256 bit key size is the current standard for the AES encryption and it takes trillions of years to break by a brute force attack [32]. To share the AES key it uses RSA public and private key mechanism with the key size of 2048 bit. Currently stranded RSA key size is 2048 bit [33].

- Integrity, non-repudiation and authenticity

IPFS is designed for the permanent web. It access the shared contents by the content hashes. Ones the content is change, the hash of the content also has to be changed. IPFS assign unique node id for every node in the network and it verifies the sender of the data. Therefore integrity, non-repudiation and authenticity is already implement within the IPFS protocol [3]. Then E-IPFS system also ensure these properties because it works on top of the IPFS.

- Availability

As mention in the Literature review, IPFS distributed system has higher availability rather than the centralized web system. This will ensure the availability of proposed E-IPFS because it also run on top of the IPFS protocol.

The following test cases that use for the evaluation process in experimental process (with respect to original IPFS and E-IPFS).

- Latency Test
- Stress Test
- General Usability

4.3 INSTRUMENTS

For the experiment in this research, it uses two identically same HP laptop computers (Intel Core i5 @ 1.70GHz, 4th Gen with 4GB RAM) that run Windows 10 Pro [25] Operating system and installed js-ipfs [26], [27] version 0.40.0. These computers are connected to a same WiFi network throughout the experiment. For the cording, it uses sublime text 3 [28] text editor and Microsoft excel 2015 [29] spread sheet for data analysis and draw graphs.

4.4 RESULTS

The first data set are generated for evaluate the efficiency of the proposed system by measure the time for executions with compare to the default IPFS process.

4.4.1 Latency Test

4.4.1.1 Measure average time for generate RSA key pair and share public key in IPFS

As the first latency, it noticed that there would be an additional time to generate RSA key pair for user and share public key in IPFS network. For this experiment, it generated random 100 RSA key pairs (size of 2048-bit) and share it in IPFS network. Then it calculate an average time for the above process.

In this experiment, 100 random RSA key pairs with the size of 2048-bit, save it in local machine and share the public key in the IPFS network. This process measure the time for each cycle in milliseconds. The key number (given by the loop index), process time and returned IPFS hash path saved in a ".CSV" file [Appendix A]. The

key files are located on a folder in the local machine directory that according to the key number.

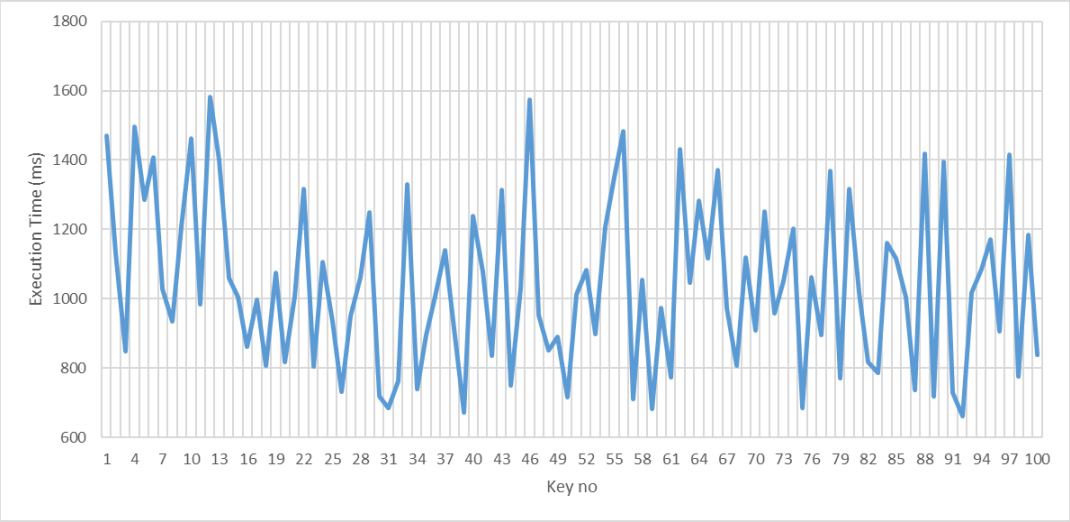


Fig 4.4.1 Time takes to generate RSA key pair and share public key in IPFS

The Figure 4.4.1 shows that the time taken to generate RSA key pair and share public key in IPFS in milliseconds. It gave the average time of 1039.54ms for one key pair generate and public key sharing process. It approximately equal to the one second. Relative to the practical user experience, this time can considered as very small factor.

4.4.1.2 Compare the data adding time to IPFS and E-IPFS

In this experiment, it measures the average time that takes for add the given size data in to IPFS and E-IPFS. This test is used three different file sizes that identically same as the latency test in Acl-IPFS experiment [11 p.1054]. They are 246kB, 2MB (2048kB) and 10MB (10240kB). The 100 random files are generated for each file sizes and calculate average time for add data to IPFS and E-IPFS [Appendix B].

For the E-IPFS, there use different public keys to share the given data to ensure that there has to spend time on retrieving the receiver's public key in this process. There public key path hashes are generated according to the process, which mention in section 4.4.1.1.

The boxplot compression in the Figure 4.4.2 shows that the difference of the average time for add data to IPFS and E-IPFS is lies around the 1.5 seconds. For the E-IPFS encryption process, it uses same receiver's RSA public key and randomly

generated 100 AES keys. This ensured that time takes for retrieve receiver's RSA public key is constant for the all file sizes. The returned values of both IPFS and E-IPFS address paths are saved in a ".CSV" file within the experiment.

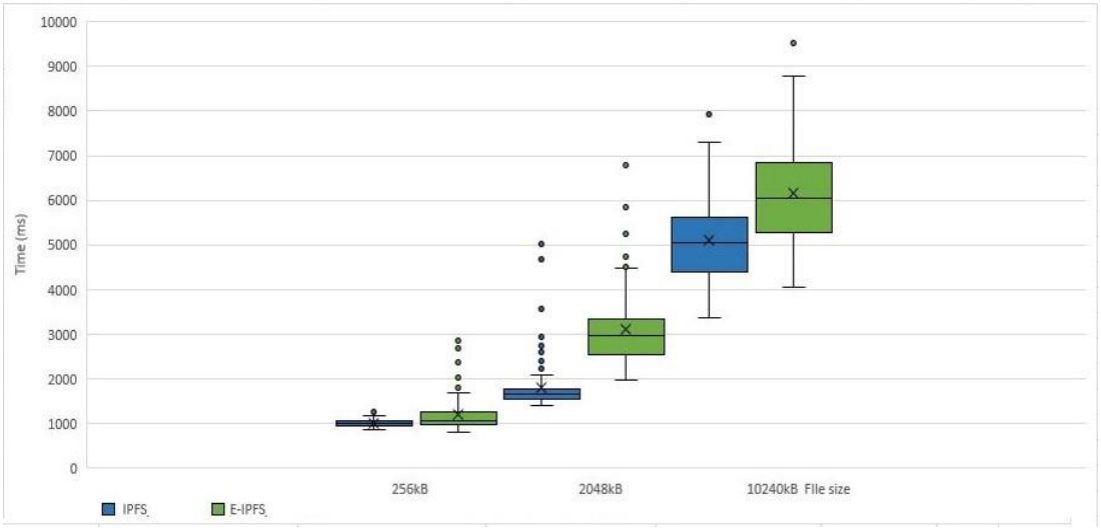


Fig 4.4.2 Data adding time to IPFS and E-IPFS

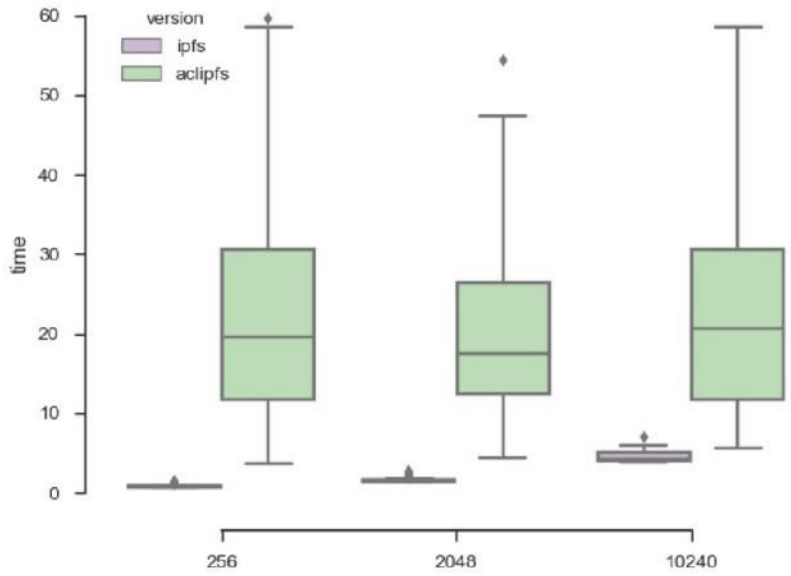


Fig 4.4.3 Boxplot for comparing adding files to IPFS and acl-IPFS [1]

The figure 4.4.3 shows the box Boxplot for comparing adding files of 100 files three different sizes (256kB, 2MB and 10MB) to IPFS and acl-IPFS [1]. These acl-IPFS time figures are always more than the 10 seconds.

File size	Average time for add data (ms)		
	IPFS	Acl-IPFS (Approximate)	E-IPFS
256kB	1011.2	19000	1201.91
2MB (2048kB)	1793.35	17000	3126.12
10MB (10240kB)	5116.83	21000	6166.2

Table 4.4.1 Average time for add data to IPFS, Acl-IPFS and E-IPFS

The table 4.4.1 shows that summary of the experiment results, which are describes in section 4.4.1.2. When the file size is 265kB, there are 190.71ms delay gap and 1049.37ms time delay for 10MB data in data adding operation in E-IPFS. These values are in millisecond scale. For the practical usage this delay will be not very dominate factor for the user. Figure 4.4.4 visualize the average time variation of the IPFS, acl-IPFS and E-IPFS as bar chart.

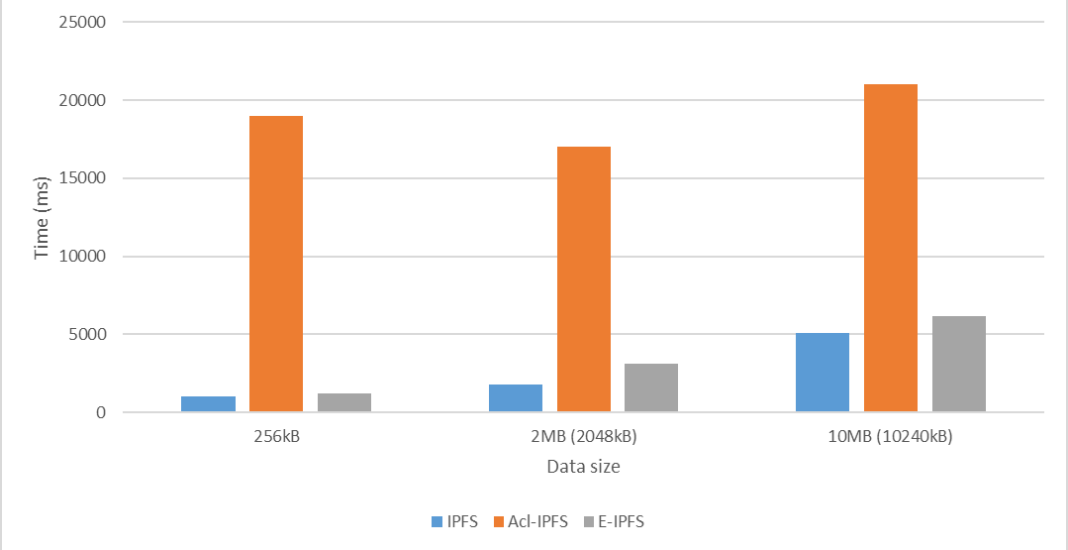


Fig 4.4.4 Average time for add data to IPFS, Acl-IPFS and E-IPFS

This time gaps are increased when the file size get larger. However, even the 2MB data, there are only 1049.37ms delay gap for data adding and 375.18ms delay gap for data retrieving in proposed E-IPFS. Acl-IPFS system encryption mechanism depend on the Ethereum block-chain. Every process it has to communicate and make interaction with the external Ethereum network smart contract. Therefore, acl-IPFS takes very large time with compare to the original IPFS network. However, with compare to acl-IPFS, the proposed E-IPFS is more efficient.

4.4.1.3 Compare the data retrieving time from IPFS and E-IPFS

This experiment retrieves the data from the IPFS and E-IPFS, which are shared in previous experiment (section 4.4.1.2). The previous experiment retrieved IPFS hashes (data shared IPFS addresses) are saved in a ".CSV" file and it used to retrieve the data in here. Identically the retrieved file sizes are 246kB, 2MB (2048kB) and 10MB (10240kB). The 100 random files are generated for each file sizes and calculate the average time taken for retrieve data from IPFS and E-IPFS [Appendix C]. For E-IPFS encryption process, it uses the RSA public and private key pair, which generated in previously (section 4.4.1.1).

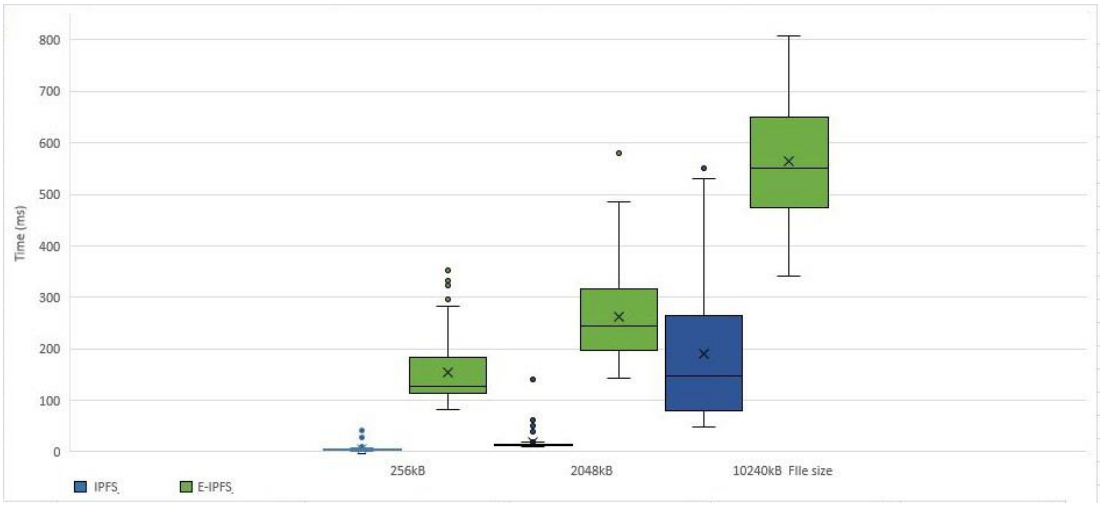


Fig 4.4.5 Data retrieving time from IPFS and E-IPFS

The boxplot compression in the Figure 4.4.5 shows that the difference of the average time to retrieve data from IPFS and E-IPFS. In acl-IPFS experiments, it not measured the time that taken for retrieve data from acl-IPFS network.

File size	Average time for retrieve data (ms)	
	IPFS	E-IPFS
256kB	5.79	153.4
2MB (2048kB)	18.62	263.45
10MB (10240kB)	189.68	564.86

Table 4.4.2 Average time for retrieve data from IPFS and E-IPFS

The table 4.4.2 shows that summary of the experiment results, which are describes in section 4.4.1.3. When the file size is 265kB, there are 147.61ms delay gap and 375.18ms time delay for 10MB data in data retrieving operation in E-IPFS. These values are in millisecond scale. This is the additional delay, which caused by the

decryption process of the proposed E-IPFS. For the practical usage this delay will be not very dominate factor for the user.

4.4.2 Stress Test

4.4.2.1 Compare the data adding average time with respect to the data size in IPFS and E-IPFS

The goal of the stress is that to identify the time variant, which respect to the data size. It measures the operational delay for the both IPFS and E-IPFS network by increasing the data size which going to share through the system for a same receiver.

First, it generates three sets of 100 random data 10kb to 1MB. Then add them to the IPFS network and recorded the operational time in a ".CSV" file. Then it calculate the average time for each execution for minimize the outlines in the dataset. The calculated the average time with respect to the data size as in table [Appendix D].

For E-IPFS, there is few more steps to cover. First, it generates the 100 random data 10kb to 1MB. Then encrypt each data using randomly generated AES key. This AES key is encrypt using receiver's public key (public key is same because the same receiver).

After the receiver get the symmetric key, it generated two data sets for different content sizes that shared for IPFS network. By using the AES decryption.

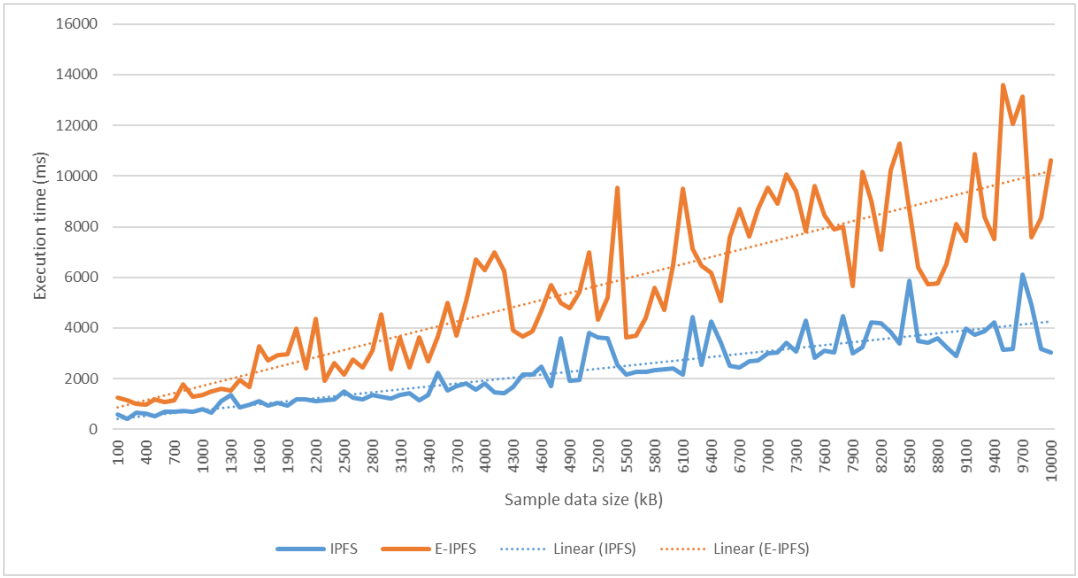


Fig 4.4.6 Data adding avarage time with respect to the data size in IPFS and E-IPFS

As expected, the graphs lines, which are in Figure 4.4.6 shows that the growth of the execution time that increase with the file, size in both IPFS and E-IPFS. The delay of the E-IPFS. Is due to the encryption process that takes more time when file size get larger. But this time is relatively lower than the Acl-IPFS system [1] which use Ethereum block chain for maintain the access control.

By the regression analysis, it generate the maximum R^2 (R-squared) (also called the coefficient of determination) value that most fitting curve for the datasets. This gives the trend line for each graphs. The most fitted (simplest model) equation, which generated by the Excel software for IPFS shown in equation 4.4.1 and it has 0.778 value for R^2.

$$y = 38.711x + 387.99$$

Equation 4.4.1 Trendline equation for data adding time in IPFS

The trend line equation for E-IPFS, shown in equation 4.4.2 and it has 0.7707 value for R^2.

$$y = 94.052x + 783.73$$

Equation 4.4.2 Trendline equation for data adding time in E-IPFS

This is the experimental model for the data adding times for IPFS and E-IPFS with compare to the data size. By theses equations it can calculate approximate values for the time that taken to add data given size data for both IPFS and E-IPFS networks. In general these calculated time values gives the comparison and between IPFS and E-IPFS. Therefore, user can decide the additional time taken by the E-IPFS for add the given size data is reasonable.

4.4.2.2 Time to share the given size data to different number of receivers in IPFS and E-IPF

In this experiment, it generate fix size (1024kB) of random data and share it to the 1 to 25 number of receivers in both IPFS and E-IPFS. In here random data is use to ensure that no node level replication occur in at the experiment nodes. Because if the data is already in the node, IPFS no longer try to add it to the network. To reduce the data outlies it generate three set of data and calculate average time to share the data for each number of users [Appendix E].

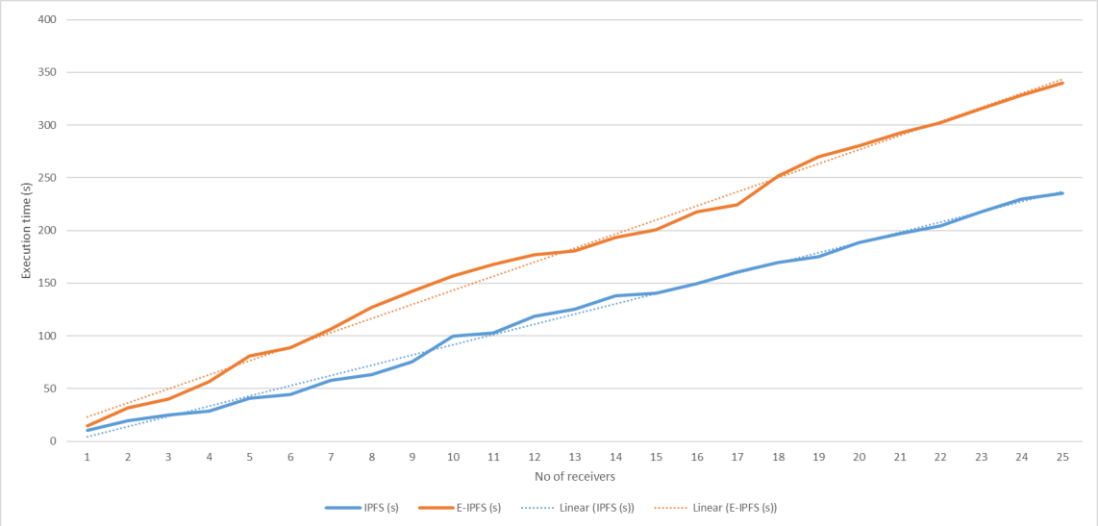


Fig 4.4.7 Time to share the 1MB data to different number of receivers in IPFS and E-IPFS

Figure 4.4.7 show that the average time to share 1MB data to 25 users in IPFS and E-IPFS. The difference of the graphs vary form 50 seconds to 100 seconds by the number of users.

However, when the additional time for E-IPFS system is increasing by the number of users, which data going to shared. The IPFS takes average 250 seconds to add 1MB file to the network. This value is high because the data context is different. IPFS not takes that much of time for same data because it has not allow to node level data duplication.

The data will not added to the node if it is already there. In other hand, E-IPFS entirely add new data to the network in every time. Even there is same data, the encrypted data will be different because of the encryption key is different for every user. This can identified as the main drawback of the proposed E-IPFS system.

According to the regression analysis, it can generate most fitting curve with the maximum R^2 (R-squared) (also called the coefficient of determination) value for the results datasets separately. This gives the trend line for each graphs. The most fitted (simplest model) equation, which generated by the Excel software for IPFS shown in equation 4.4.3 and it has 0.9955 value for R^2.

$$y = 38.711x - 5.3357$$

Equation 4.4.3 Trendline equation for share 1MB data in IPFS

The most fitted curve equation for E-IPFS, shown in equation 4.4.4 and it has 0.9945 value for R^2.

$$y = 13.35x + 9.9316$$

Equation 4.4.4 Trendline equation for share 1MB data in E-IPFS

These are the models that generated by the experimental results. By theses equations it can calculate approximate values for the time that taken to share 1MB data to given number of users. In doing so, user can compare the additional time that taken from the E-IPFS, with respect to the original IPFS network.

4.4.3 General Usability

For the common usage, there are two main commands in IPFS. They are used for the data add and data retrieve from the IPFS network. Description of each commands are in table 4.4.3.

	Command	Parameters	Return	Description
1	ipfs add	Data which want to share	IPFS path	Add given data to IPFS network
2	ipfs get	IPFS path	Original data	Retrieve the data from IPFS network

Table 4.4.3 General usable commands in IPFS

E-IPFS system is used the identically same approach as IPFS commands. In addition to that, it has to generate user's private and public key pairs and share the public key in IPFS network. These commands have to execute as '.js' files. Each However, this additional work is one time for the user. The description of each commands in proposed E-IPFS are in table 4.4.4.

	Command	Parameters	Return	Description
1	generateRSAKeyPair	None	private and public key pair	To generate private and public key pair for the user (one time action)
2	publishRSAPublicKeyToIPFS	None	public key IPFS path	Publish user's public key to the IPFS network (one time action)
3	publishDataToEIPFS	Data which want to share and receiver's public key IPFS path	IPFS path	Add given data to IPFS network
4	getDataFromEIPFS	IPFS path	Original data	Retrieve the data from IPFS network

Table 4.4.4 General usable commands in E-IPFS

These E-IPFS commands are tottally independent of the version of the IPFS . It uses only the 'ipfs add' and 'ipfs get' basic IPFS commands. Therefore the genaral useblility is very much similar to the IPFS.

Chapter 5: Conclusion and Future Work

This research focused to make a distributed data sharing mechanism, which can control the access level. This uses IPFS as the distributed system and it proposed a cryptographic mechanism to control the access level.

According to the problem definition, IPFS distributed system has not in-built access level control mechanism. By the literature review, it identified that there cannot control the access by an admin node because IPFS is purely distributed system. Then this research answered to the research question by proposing a cryptographic mechanism which called E-IPFS (Extended version of IPFS), which can implemented within the IPFS distributed file system. By the proposed mechanism it achieves the information security constrains which are describes in evaluation (section 4.2).

The experiments, which are, perform for test the latency and the stress of the E-IPFS system implies that there are additional delay with respect to the original IPFS because of the encryption process. However, this time delays are not relatively large factors. At the evaluation of the Acl-IPFS system [1] is conclude that the delay for the access control mechanism is due to the process that run with the Ethereum block chain. Therefore, E-IPFS system uses a key sharing mechanism within the IPFS network is more efficient than the Acl-IPFS system. Furthermore, E-IPFS system not depend on the any other protocols to control access level of the shared data.

As the design of the E-IPFS system, it has to generate RSA public and private key pair for every user. One it generated E-IPFS facilitate that, user can use the same key pair for E-IPFS system in any computer, which run IPFS network. In other words, the E-IPFS key pair can use as a password to the access private contents that shared in IPFS network at anywhere.

By considering the evaluation, it can conclude that the proposed E-IPFS system can use to share the confidential data in secure way through the IPFS network. However, as expected there is an additional time for that cryptographic operation. These additional time factor behaviors are modeled as equations using the experiment data sets.

E-IPFS system designed on top of the IPFS. Because of that, it uses the common commands in IPFS and new commands are identically same as the IPFS commands. This will ensure the general usability of the system.

E-IPFS system cannot retrieve the access form the receiver that already shared to the IPFS network because of IPFS objects are immutable. But IPFS protocol has mutable object version calls IPNS [1 p. 9], [30]. It can enhance the E-IPFS system to share the E-IPFS object through the IPNS network, the date owner can edit access i.e. revoke the permission by deleting encrypted AES key form the relevant E-IPFS object.

E-IPFS system store the contents by encrypting the original content. Therefore, E-IPFS contents cannot perform the search queries. In other words, it cannot search the data by the traditional searching mechanisms. But for the future, E-IPFS further extend to perform search queries by introducing the Homomorphic encryption [35] which can use for perform search queries on the encrypted data.

References

[1] M. Steichen, B. Fiz Pontiveros, R. Norvill, W. Shbair, and R. State, “Blockchain-Based, Decentralized Access Control for IPFS,” Jul. 2018, doi: 10.1109/Cybermatics_2018.2018.00253.

[2] “Home | Ethereum,” *ethereum.org*. https://ethereum.org (accessed Aug. 18, 2019).

[3] “IPFS Draft-3 IPFS - Content Addressed, Versioned, P2P File System (DRAFT 3).” [Online]. Available: https://ipfs.io/ipfs/QmR7GSQM93Cx5eAg6a6yRzNde1FQv7uL6X1o4k7zrJa3LX/ipfs.draft3.pdf.

[4] K. Nadiminti, M. Assuncao, and R. Buyya, “Distributed Systems and Recent Innovations: Challenges and Benefits,” *InfoNet Magazine*, vol. 16, Jan. 2006.

[5] “Comparison - Centralized, Decentralized and Distributed Systems,” *GeeksforGeeks*, Dec. 24, 2018. https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/ (accessed Oct. 20, 2019).

[6] “Where did peer-to-peer network users share which files during 2017?,” *tecxipio*. https://www.tecxipio.com/single-post/file-sharing-in-peer-to-peer-networks-2017 (accessed Oct. 20, 2019).

[7] “BitTorrent_(BTT)_White_Paper_v0.8.7_Feb_2019.pdf.” Accessed: May 16, 2020. [Online]. Available: https://www.bittorrent.com/btt/btt-docs/BitTorrent_(BTT)_White_Paper_v0.8.7_Feb_2019.pdf.

[8] “(PDF) The Bittorrent P2P File-Sharing System: Measurements and Analysis,” *ResearchGate*. https://www.researchgate.net/publication/221160480_The_Bittorrent_P2P_File-Sharing_System_Measurements_and_Analysis (accessed Sep. 16, 2019).

[9] P. Labs, “IPFS is the Distributed Web,” *IPFS*. https://ipfs.io/ (accessed Jul. 10, 2019).

[10] “What is Dropbox - Features Overview.” https://www.dropbox.com/features (accessed Sep. 08, 2019).

[11] “BitTorrent | computing | Britannica.” https://www.britannica.com/technology/BitTorrent (accessed Oct. 20, 2019).

[12] C. Hoffman, “How Does BitTorrent Work?,” *How-To Geek*. https://www.howtogeek.com/141257/htg-explains-how-does-bittorrent-work/ (accessed Oct. 21, 2019).

[13] “Apache Hadoop 3.2.1 – HDFS Architecture.” https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html (accessed Sep. 16, 2019).

[14] “HDFS Architecture Guide.” https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html (accessed Sep. 23, 2019).

[15] U. Fazil, “IPFS: A Distributed File Store,” *Medium*, Feb. 06, 2019. https://medium.com/block360-labs/ipfs-a-distributed-file-store-533cda4c6047 (accessed Sep. 14, 2019).

[16] A. Tenorio-Fornés, “Open Peer-to-Peer Systems over Blockchain and IPFS: an Agent Oriented Framework,” Accessed: Jul. 25, 2019. [Online]. Available: https://www.academia.edu/36647705/Open_Peer-to-Peer_Systems_over_Blockchain_and_IPFS_an_Agent_Oriented_Framework.

[17] I. Journal, “IRJET- DISTRIBUTED DECENTRALIZED DATA STORAGE USING IPFS,” Accessed: Jul. 21, 2019. [Online]. Available: https://www.academia.edu/39649480/IRJET-_DISTRIBUTED_DECENTRALIZED_DATA_STORAGE_USING_IPFS.
[18] IPSE Team, “IPSE: A Search Engine Based on IPFS,” [Online]. Available: https://ipfssearch.io/IPSE-whitepaper-en.pdf.
[19] K. Chandrasekaran and M. V. Thomas, “Distributed Access Control in Cloud Computing Systems,” in *Encyclopedia of Cloud Computing*, John Wiley & Sons, Ltd, 2016, pp. 417–432.
[20] D. Team, “Hadoop HDFS Architecture Explanation and Assumptions,” *DataFlair*, Jun. 10, 2017. https://data-flair.training/blogs/hadoop-hdfs-architecture/ (accessed Sep. 23, 2019).
[21] “XtreemFS - Fault-Tolerant Distributed File System.” http://www.xtreemfs.org/ (accessed Sep. 23, 2019).
[22] M. Ober, “The IPFS Cloud,” *Medium*, Dec. 04, 2018. https://medium.com/pinata/the-ipfs-cloud-352ecaa3ba76 (accessed Oct. 19, 2019).
[23] C. Laybats and L. Tredinnick, “Information security,” *Business Information Review*, vol. 33, pp. 76–80, Jun. 2016, doi: 10.1177/0266382116653061.
[24] Casper Lassenius, Timo Soininen, “Methodology workshop 26.11.2001 Casper Lassenius, Timo Soininen, Jari Vanhanen.” .
[25] “https://www.microsoft.com/en-us/p/windows-10-pro/df77x4d43rkt?activetab=pivot%3aoverviewtab.” https://www.microsoft.com/en-us/p/windows-10-pro/df77x4d43rkt?activetab=pivot%3aoverviewtab (accessed Mar. 20, 2020).
[26] “JS IPFS,” *JS IPFS*. https://js.ipfs.io/ (accessed Mar. 20, 2020).
[27] “ipfs/js-ipfs,” Jun. 20, 2020. https://github.com/ipfs/js-ipfs (accessed Mar. 20, 2020).
[28] “Download - Sublime Text.” https://www.sublimetext.com/3 (accessed Jan. 19, 2020).
[29] “Microsoft Excel, Spreadsheet Software, Excel Free Trial.” https://www.microsoft.com/en-us/microsoft-365/excel (accessed Jan. 20, 2020).
[30] R. L. Rivest, A. Shamir, and L. Adleman, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” p. 15.
[31] N. I. of S. and Technology, “Advanced Encryption Standard (AES),” U.S. Department of Commerce, Federal Information Processing Standard (FIPS) 197, Nov. 2001. doi: https://doi.org/10.6028/NIST.FIPS.197.
[32] “Wayback Machine,” Mar. 06, 2016. https://web.archive.org/web/20160306104007/http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf (accessed Jan. 21, 2020).
[33] E. Barker and A. Roginsky, “Transitioning the use of cryptographic algorithms and key lengths,” National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-131Ar2, Mar. 2019. doi: 10.6028/NIST.SP.800-131Ar2.
[34] “IPNS.” https://docs.ipfs.io/concepts/ipns/ (accessed Jun. 14, 2020).
[35] F. Armknecht *et al.*, “A Guide to Fully Homomorphic Encryption,” 1192, 2015. Accessed: Jan. 21, 2019. [Online]. Available: https://eprint.iacr.org/2015/1192.

Appendices

Appendix A

Time takes to generate RSA key pair and share public key in IPFS

Key no	Execution Time	IPFS hash path of the public key
1	1470	QmRcuRiunCzvHVMAcS5PiAaJCiHaFC2s4rJ3aQibSEHySp
2	1131	QmQVB9g7UDHbN8ifGymPCPPxmVnsRLEjzskF4LA2c1JZZZ
3	849	QmUqrNYXXwNeawJ47g4kEigdz8VDyypSHbNqdHC5TWGyFe
4	1497	QmVpPVDvJBHHGhVs3nrU5LPbzXMJH6StzUGyEJB3ssiwLv
5	1285	QmdzEL5sBn5P1y5x2exDHMQDzSazkdkoiXd3GVfY8VtS2L
6	1408	Qmcqnx5dzWNj46YcimXgnxgEcj9bnf6ZvL1hc5M3vPC3Pk
7	1029	QmWXSDpcSXU5npX2VNfK3i1Nk2XG4NJnA2JQTXDRQZTTJA
8	934	QmeZoJVj6HEyZyuCw66hEQBkbvWmAPzbs3PS8BpQkM7LTp
9	1208	QmNtHnDt2Qg8jPZRLqyPf4EseYXQvZjLvyLhDiYYtuf7jN
10	1463	QmeKBmQc3UHmzWTpwTQYGyfiBrQcDmYmAdNHBvSbSZ6RH6
11	983	QmZcP4cihyEWgv3hoWSt7FsZtKUC6N1yUosXJ4GFWunUED
12	1582	QmNemGxq3JxxthQGGYZCutoZyAMrp2Jqfhw97nSNE2TyYA
13	1401	QmXPnJiG5hfy6K4cr3GQLEszB74sNu2wue3ZAaTKadd5Ri
14	1060	QmV6f7uyZkPyaeuWcGgE2izC1Ju8jzCqScuSKbs4UVeCEq
15	1005	QmWguDi6NyQZ57xMTokpEUTDBVaCA9xDdmZ1KnNBWueWBe
16	861	QmeXE1z7a49SGwFVbH82CwULw2EdwnoF1XGm6Y1bGRRpBk
17	996	QmaXjQdjg6XbTEEa23YjmhAsLzxsiCxzr8Gigz1Azc8R83
18	807	Qmeps2BPmkgDf49sVpNXwpZabJs6dVfT4yLfoUakW44JPR
19	1074	QmaxJaZLLLv8Cs1iVVPUmKQDuPU8CsBiEbxWp8NznQd6V1
20	819	QmcF9GCkVRt5BJLxASjdjB3PQrmehFgbLVkFfX94KbNQJQ
21	1007	QmVHvQFW8jPYpZaHe3DAYt1DX1nYdN4xtFNXr2AGU81oz7
22	1318	QmaPm2PgmbU6CgSxZsF4HsYsLfgfHsm2Zz78kHbohzNa6T
23	804	QmdUpGqzR47TdDMd3mXC9Rqci6ENVYsq8w5f7xNdd7iEV2
24	1107	QmcDhKZacKrsWHY3oQCitx6kiNjwphWNyAz586mRLRQ5Ff
25	936	Qme575ZgEnUVchgMHaJxaVprWkeM8UKHVxM1PnaiVqbzRv
26	732	QmXYTmpwgtQwwtmjsobqgvEc1PAS3PcwifNJiQTVWFVFan
27	951	QmYwd3S9jPfy28qsngTfDdxrJUrWMHuvMb4d6iMfuN9RpU
28	1063	QmTvjRDG1U3Hc1EjZZswmXrKmt4NHDcFz4sEBVgaQu6cjF
29	1250	QmT8A9wZEFzadjCz9B8RqWbhMyhgoBTxgTz6nWTqapcDe9
30	718	QmaUqMkt7bQkkjY9gbnkoAt3GQ2d1tT3We5WqJRqjMNm4e
31	686	QmbN3JDgHVbBLjLmAD1eeTyz63dYJt1TePF9a3PbrR32jg
32	763	QmUvdnSrHKEaCBhXCPRbjtjxTSMBRvcaS55pq5Tdm6W8JL
33	1331	QmX5oD1ZYhS83ZvzEQBBUMG9ZaLZKKUD1etcvNcRKGxSu4
34	741	QmNqY2Vg6fwjkvs5ggUp2tnZ64M8LAsH922ML7DW1a7Ggs
35	897	QmbemX12A17LnPoEj8beAbiWXvNU1TcqnBgQikDKQSJvR3
36	1007	QmTdNUp1HxkjFLtYDGtTkgpgYFRfC2vUjZpvNa2aMcgH8x

37	1141	QmY99QwvYR8T7FKqGuwgn8Sqf1utSjBQhWRGsM822D5QCv
38	913	QmPjLhw43AKqFazMooVcRG5Ai1sc7vZ8NwwH8oGSEiXzmG
39	673	Qme3S8WwetvUuWFYtmBGitNmRLpFAHTEHUbsE1keKcJ9GG
40	1239	QmPb5PCKUsmGwBoUeCiZpmWnkHsTXnSddzqDewqHckkoi3
41	1078	QmaV3T871fmuwc99ewrXknTnuyWCMR36JqPgW88Q6F7XVJ
42	837	QmZVxSt6BddxTwYXobv2Ch83fSQ5fN3JZ3mKe7ssE7Mm85
43	1315	Qma6DWSdmJqZw8CMosDEtKo67eTwkM6vgSTfz3dvBwd1QX
44	751	QmXG18pAiHNfQz5CsQuzwAfqtFsaw57wkzr91gdHvKhcNJ
45	1031	Qmdork2m3BpbA9YwnresEjWRE8upQBqNvBmXjiHL5ppNbV
46	1575	QmU3FmHWGEGfXvEL8LepTxenqG2HFg6XC6G8RxnjDxd8id
47	953	QmYv4di4CTBuV85s24oGySQhh3bdpZbyJznEML5iQfZYEa
48	852	QmPT5NGPXdVeixxZQg3NEAzwYAHJfhp2PZuMzdNbdeMVuq
49	891	QmcJBGnVy7sJ69oaVWArvRmxVpjhppTgwMAuG7nCHwc2LT
50	716	QmQxqHfXbj4T8GRAML2YXZ1Nqr8wD1gLm1oQwcAZfhMj2d
51	1014	QmbWYcNFhZruWrSb8MPQVYy5YNYf32sgbkLhYy4FvU21mX
52	1082	QmXXRV1yx6uFfk135qRpkoRRr5yqC2kQjVeUFMdiRvXXpp
53	898	QmQFqoXdntMaXRqG5ixvQ83khDUJCCvo2CVNsEGBq24XVD
54	1207	QmaJbrQcJL22BGGvkoe4rM4d3Ki33yTL6Fcc5k1zA81QYB
55	1350	Qma19MPLdJnZhvhwo9Sb87fsrCBkQTYtaduZzDiFK7UeHj
56	1484	QmXp5C9b3ZZbFBZ6SzJxB3pnmkzbJY5GXLXbfaGfB2kUEu
57	711	QmZTuVtRN9pTBSyDgyB3jBGjjxevCqznLcwY7aCub39ZHe
58	1054	QmS1x8FhRFs2SFTZMAyCJQv4ggu6GEFpQYxz7yRGcYZaY6
59	682	QmNm2mSqm6922F4spk7qkkcqceH3Q89tNnm4rzxLC2GtgV
60	974	QmNSACYbH6LKcoTQXCVWcfQgVuVzgh5HcutMqCPwydP2Js
61	774	QmXfZhdM5Ra7XnH4cJgp9E68BPVio6L3RzEZg8CZeTeS2N
62	1430	QmQVMqhMNMEmNYAcTWWKNdHqcanm8yeMLS15jcT4gkP9Rt
63	1047	QmVTYR6bAL62YbJubavui9ut2xrambNw7Javm8zmdza9hU
64	1284	QmSMuZuBTsDi7VCqb1T1V8VYP5118RwLj75EPZgkMhfPoy
65	1117	QmX7fbGcC8NjZTvuRT16orC5aprZJ3enA3WvfREMeDaiNM
66	1371	QmWYL35RSELVFaot7m9TC8bZcgwHmtVcbbY1ZvBCccfCmM
67	973	QmVbuGDUbs8EdnWGVwHfzFMRkEdxtZvEKyJaq2yG8cmY8b
68	808	QmWrbk73HwaQCNtudzej6A3hEhhYXG5cFg3zAadS7g8T15
69	1118	QmPuzzaEZLPgMde9zgmq5AAYsEsYvnfjVim112GM94DU7H
70	910	QmSHAE3kWRhrspv79JJBPTbFyij6MQvXJSMNniqjtKkEPA
71	1252	QmaS8VaLTecN1z12VP2HBDdrnGKWdRDxPrsyVNqpaQVP2J
72	959	QmUtnNdgde1y17kznJSJmSmszzSQMnfQvd5oopfNyKy4Jx
73	1052	Qmf7apHa16wTEfSaMumXjLLgjkatBX3F8NTGEwkaDFvhTe
74	1202	QmQVQ5UFntuN7dLfSjdruy7bBC6ZCsExjbkAyBueV8bBjH
75	685	QmWkiC3fbGEBKxfBW9YAKbjMrV1q5JNh21HH1akBoRjGsY
76	1063	QmQt4qmdjGQeq56hYS7FB7iWpFUhem6BhUonmT4SLH8WQC
77	896	QmYmbzcytkwz2JV7X1LigsuQQU6B5eA9qfPsf6DBwjrWHE
78	1370	QmUY3YgbLwUoTazBgj5soBtxQwicjMMcuxPVrdc8HTGFYm
79	771	QmNrbM1VHfM7JW48XKpHPfYR3xfCbCbN8pKWMxFPmTnRxv
80	1317	QmUWpy7m4bFXUvhgUvWU7t3HJccAH473GY41RWMWUK3NDf
81	1019	QmebMyTrHBh1CV7xtRDX5sFJ6qwkpmeiZU4cgbzxizfCcU

82	819	QmacuSgb891SXn5n1P2fUM3bckmHmJ55rWrtEnGBVfQ13d
83	786	QmTZ1PsHUYMGCGp46QfBrttLfTtxryJGzRYZ72AaP6gFVx
84	1161	QmdTKnmDD4CYq65x335HdbeFRwhcJb38CWcCaiDN7eUu2w
85	1116	QmdUTRegPzFTW7eGp94ZrxubxunZLudn1Bgx4UTbNdEXXJ
86	1005	QmcvzfYd8wchMv7qWoeqwb2EGWKNUXSuswrsoJVSSnuLzj
87	737	QmQoknxccWggKk4C4fDGVKpXnRCCnAnS6aiig1CxPj67py
88	1418	QmUuntfnhgz13vp6shdNwsdyXJgiMLpSZRxMp9dBEjZcwn
89	718	QmYP1necx3s3bNiiLn4HhbUqCyqv8UaihvjxeZbtXpyatQ
90	1396	QmWMuUBxtr1TDcRCYCQNLTj23vToHq19H1mBgEdcXxSe4b
91	729	QmQqkjmV5RguJYyH3Y73BBD6QtbaEtvMYigMCHWt5TMi5G
92	662	QmSprdkfPEB1oNUyQFEns3XrPmWTPThhFJprcMY3CiJAuc
93	1018	QmSd8GYvovhu4qhNiu1CEbJc6cKn2jSiq3VkjQmq5chwSy
94	1085	QmYcVSjvR66xANmKnwa2v1yVcaSybUbRfKVjisnfnkt3Ew
95	1172	QmP4NV6MG87cSZ7UyRB4WBrdUzfgfQNh2iDNnyrQizkJy8
96	905	QmPC7fp5E3HMjMSMoA9yaYBTABqEhps8T9fzMio3Ba7hsm
97	1416	QmdPvU12FqZ363QoQ3ctKj9ULEXw9rzHRFdX3gKoDJdd3o
98	776	QmRAgkXUEi7k5bp2B9vXtnLYRGXe6c9EQJEChrfUH5qVdd
99	1184	Qmf3e2bHNC1DFg5Dg13ZJBsB5DmVNpUZ7WxWk6gBuruZtx
100	839	QmZ9Y6R1HFyD3BUPhtdLTPNBWeCSArX2TuA74p3r1Vt3qe

Appendix B

Time takes for add the given size data to IPFS and E-IPFS

No	Execution time (ms)					
	256kb		2048kb		10240kb	
	IPFS	E-IPFS	IPFS	E-IPFS	IPFS	E-IPFS
1	1356	1407	1900	3248	4358	5230
2	1127	1132	1787	3177	3871	4645
3	1171	1053	1607	3118	3578	4294
4	1254	88	1613	2687	3676	4411
5	1119	1006	1556	2862	3379	4055
6	981	869	1574	3054	4078	4894
7	1002	917	1518	3222	3918	4702
8	1009	906	1579	3054	7308	8770
9	985	993	1562	3030	3970	4764
10	955	931	1505	3155	3859	4631
11	1083	931	1534	3219	4782	5120
12	1018	928	1810	3358	3838	4606
13	934	1045	1648	6784	3829	4595
14	1050	1020	1496	2566	6392	7670
15	973	951	1738	4787	5722	6866
16	1008	913	1757	2867	4307	8415
17	1024	942	1844	2744	5129	6155
18	1096	1007	1702	2771	4986	5983
19	1082	1466	1806	2740	5125	6120
20	1007	2725	1684	2869	6249	7499
21	1012	1187	1622	3243	5641	6769
22	1040	977	1679	3144	5351	6421
23	962	994	1711	3109	5609	6731
24	1006	1300	2830	3624	7033	8440
25	1004	1321	1714	3088	5040	6048
26	1101	1053	1439	3983	5394	6473
27	1190	1086	1424	2658	4337	5204
28	1032	876	1419	2606	5612	6734
29	1059	886	1591	2573	4156	4987
30	930	988	1585	5264	7970	9564
31	987	955	1547	2866	3843	4612
32	943	820	1403	2374	3663	4396
33	1031	1044	1543	3144	5606	6727
34	1080	1004	1516	2432	4450	5340

35	925	844	1412	2528	4904	5885
36	991	874	1546	3022	4482	5378
37	997	1000	1585	3066	7074	8489
38	925	1054	1522	3226	3595	4314
39	914	995	1686	2749	4853	5824
40	991	968	1745	3475	5064	6077
41	1012	958	1473	2964	5302	6362
42	1003	994	1639	5854	5799	6959
43	1068	1041	1669	2087	4452	5342
44	1108	1328	1740	2231	5338	6406
45	1018	1820	1757	3530	5220	6264
46	1023	1512	1499	2507	4196	5035
47	1068	1231	1612	2238	6154	7385
48	1187	1501	1487	3041	4413	5296
49	1069	1696	1872	3418	3963	4756
50	900	1342	3570	3072	5515	6618
51	996	1182	5029	3956	5198	6238
52	1084	1493	1527	2240	6734	8081
53	1036	1321	1582	2313	4397	5276
54	928	1038	1590	2409	5565	6678
55	1021	1100	1402	2420	4431	5317
56	945	1054	2286	2554	4873	5848
57	959	1665	1759	2966	4531	5437
58	941	1276	1861	3208	4655	5586
59	930	1151	1634	4520	6651	7981
60	954	2038	1727	5906	5379	6455
61	967	2369	1746	4478	4735	5682
62	937	2475	2244	2319	7090	8508
63	1019	2864	1572	3029	4702	5642
64	1057	1377	1681	2125	4712	5654
65	1081	1109	1968	2193	6211	7453
66	1142	906	1640	2267	5276	6331
67	1086	1043	1537	2143	5443	6532
68	1041	1032	1483	3242	4881	5857
69	1070	1035	1532	2220	5389	6467
70	1145	951	1485	3048	5445	6534
71	1019	897	1475	3613	3899	4679
72	1023	1008	1670	4332	4303	5164
73	923	2692	1771	2320	5051	6061
74	994	897	1746	4750	4387	5264
75	916	1073	1900	2876	6237	7484
76	1151	900	1684	2266	6056	7267
77	952	898	2402	2156	5392	6470
78	991	968	4674	2572	5065	6078
79	927	1175	2608	4143	5006	6007

80	1033	1140	1503	2801	6078	7294
81	1019	1085	1841	2917	5890	7068
82	892	1042	1925	3587	4908	5890
83	949	1109	2943	3933	7073	8488
84	1016	1109	1486	4174	7922	9506
85	951	1092	1521	2575	6041	7249
86	1042	1214	1520	2897	4350	5220
87	923	1122	1911	3543	6446	7735
88	1052	1107	2100	2308	5040	6048
89	893	1042	1615	5253	5263	6316
90	904	1111	1584	2966	5192	6230
91	862	1129	1660	1976	6011	7213
92	930	2045	1543	2030	6153	7384
93	989	1823	1615	2931	4526	5431
94	933	1502	1720	2642	4668	5602
95	990	932	1713	2876	4937	5924
96	872	1012	1828	2909	4499	5399
97	1006	1138	1768	4231	4618	5542
98	891	996	1789	2928	6105	7326
99	973	1110	1722	2542	3769	4523
100	955	1666	2756	3581	4117	4940

Appendix C

Time takes to retrieve the given size data from IPFS and E-IPFS

No	Execution time					
	256kb		2048kb		10240kb	
	IPFS	E-IPFS	IPFS	E-IPFS	IPFS	E-IPFS
1	7	206	39	486	480	593
2	4	200	13	411	530	700
3	28	254	13	412	512	635
4	6	297	18	399	444	683
5	42	152	13	398	381	743
6	18	231	12	353	443	807
7	4	171	14	393	433	620
8	5	164	13	366	302	629
9	3	107	14	381	291	685
10	3	207	13	342	322	782
11	10	159	13	333	451	777
12	4	138	24	311	427	712
13	4	120	51	318	330	678
14	6	198	14	352	343	802
15	3	151	17	351	551	678
16	3	187	14	313	292	676
17	4	190	11	408	355	693
18	6	95	13	291	161	656
19	4	134	12	298	304	582
20	4	113	11	288	249	695
21	6	109	20	391	226	546
22	4	137	13	316	192	636
23	4	162	12	314	243	671
24	3	104	12	278	168	622
25	9	112	19	281	179	543
26	3	221	15	316	197	601
27	3	108	15	263	311	697
28	4	122	15	580	188	643
29	36	139	20	463	202	752
30	45	115	12	316	161	807
31	3	103	13	256	363	697
32	4	127	51	275	224	722
33	3	95	14	277	173	608
34	3	174	11	291	260	585
35	2	222	13	281	550	480
36	5	146	12	361	479	628
37	3	119	10	269	515	618

38	4	106	12	263	439	454
39	3	116	11	229	205	573
40	3	121	17	214	137	587
41	3	107	15	285	141	515
42	3	120	15	212	218	490
43	2	82	14	245	203	575
44	3	237	11	244	200	701
45	3	93	22	232	171	705
46	5	121	14	260	246	634
47	3	106	13	231	152	651
48	3	126	39	244	186	548
49	3	257	15	225	243	611
50	7	358	13	256	265	404
51	7	283	14	188	167	544
52	7	323	42	243	92	689
53	3	158	11	216	57	473
54	5	193	12	253	56	427
55	3	333	141	251	215	552
56	3	251	13	235	54	552
57	4	202	12	209	87	542
58	6	203	13	225	52	515
59	4	129	12	216	94	441
60	4	162	12	256	83	641
61	4	191	12	228	102	387
62	6	125	13	228	77	531
63	4	157	22	196	79	620
64	4	143	14	216	56	508
65	5	152	15	223	90	576
66	4	352	16	221	85	585
67	3	227	15	219	97	438
68	4	161	14	198	51	552
69	4	109	16	191	48	514
70	5	158	13	214	83	475
71	4	107	20	209	51	470
72	5	126	15	175	50	532
73	5	87	15	169	52	341
74	4	154	15	167	98	453
75	4	110	12	198	50	481
76	5	102	15	152	49	522
77	5	94	61	171	49	510
78	4	158	14	185	50	432
79	5	107	14	162	60	435
80	5	203	15	176	55	443
81	4	120	12	164	75	375
82	4	127	57	184	100	537

83	3	93	13	164	79	532
84	5	127	11	326	86	542
85	4	119	11	297	77	421
86	4	119	13	368	57	374
87	3	115	52	238	95	397
88	5	139	11	207	121	517
89	5	123	11	157	49	375
90	4	121	11	164	80	505
91	3	114	14	173	90	518
92	4	113	56	381	82	378
93	4	117	12	286	117	424
94	4	106	10	143	81	397
95	11	125	12	173	81	441
96	6	114	13	177	82	462
97	4	117	52	179	104	457
98	4	93	11	156	80	475
99	6	118	11	165	54	499
100	4	121	11	181	51	549

Appendix D

Avarage time takes to add the given size data to IPFS and E-IPFS

No	Sample data size (kB)	Execution time (ms)	
		IPFS	E-IPFS
1	100	590	1252
2	200	407	1160
3	300	644	992
4	400	626	962
5	500	531	1172
6	600	681	1078
7	700	697	1151
8	800	721	1763
9	900	690	1280
10	1000	798	1375
11	1100	652	1485
12	1200	1100	1587
13	1300	1374	1528
14	1400	885	1937
15	1500	973	1672
16	1600	1121	3285
17	1700	942	2719
18	1800	1053	2941
19	1900	953	2950
20	2000	1191	3965
21	2100	1195	2414
22	2200	1113	4354
23	2300	1140	1918
24	2400	1191	2616
25	2500	1484	2158
26	2600	1247	2762
27	2700	1173	2440
28	2800	1374	3110
29	2900	1281	4548
30	3000	1212	2384
31	3100	1364	3638
32	3200	1439	2438
33	3300	1152	3642
34	3400	1346	2693
35	3500	2241	3652
36	3600	1541	4981
37	3700	1696	3693
38	3800	1798	5058

39	3900	1580	6701
40	4000	1819	6293
41	4100	1477	6992
42	4200	1417	6264
43	4300	1682	3910
44	4400	2176	3667
45	4500	2175	3868
46	4600	2477	4666
47	4700	1700	5678
48	4800	3609	5008
49	4900	1919	4786
50	5000	1956	5400
51	5100	3813	6984
52	5200	3618	4324
53	5300	3602	5198
54	5400	2545	9525
55	5500	2149	3636
56	5600	2277	3707
57	5700	2276	4411
58	5800	2333	5577
59	5900	2373	4707
60	6000	2398	6496
61	6100	2161	9495
62	6200	4438	7118
63	6300	2538	6448
64	6400	4264	6191
65	6500	3421	5066
66	6600	2527	7591
67	6700	2443	8689
68	6800	2684	7605
69	6900	2735	8688
70	7000	2997	9546
71	7100	3024	8919
72	7200	3417	10066
73	7300	3082	9384
74	7400	4306	7825
75	7500	2844	9615
76	7600	3123	8465
77	7700	3052	7888
78	7800	4467	7991
79	7900	2994	5665
80	8000	3250	10154
81	8100	4213	9026
82	8200	4205	7090
83	8300	3826	10229

84	8400	3380	11289
85	8500	5866	8681
86	8600	3503	6406
87	8700	3415	5740
88	8800	3595	5752
89	8900	3243	6532
90	9000	2894	8095
91	9100	3964	7445
92	9200	3730	10863
93	9300	3893	8382
94	9400	4229	7524
95	9500	3145	13610
96	9600	3165	12067
97	9700	6126	13126
98	9800	4925	7566
99	9900	3170	8345
100	10000	3050	10609

Appendix E

Time takes to share the 1MB data to different number of receivers in IPFS and E-IPFS

No of receivers	Execution time (s)	
	IPFS	E-IPFS
1	10.457	14.593
2	19.6	31.941
3	25.034	39.897
4	28.458	56.553
5	40.659	81.159
6	44.221	88.869
7	57.873	106.233
8	63.469	126.987
9	75.343	142.338
10	99.592	156.982
11	102.522	167.897
12	118.422	177.214
13	125.214	180.554
14	137.893	193.545
15	140.214	200.414
16	149.873	217.892
17	160.311	224.152
18	169.874	251.452
19	175.142	269.836
20	188.521	280.314
21	197.154	292.145
22	204.155	302.146
23	217.412	315.412
24	229.635	328.542
25	235.568	339.854