USING BLOCKCHAIN TO BUILD DECENTRALIZED ACCESS CONTROL IN A PEER-TO-PEER E-LEARNING PLATFORM

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

Sihua Ma

Permission to Use

ABSTRACT

In the context of E-learning platforms, the amount of research focusing on access control is proliferating. However, research related to the decentralized access control in this field is scarce. To improve such area of research, an innovative model of decentralized access control used to protect the collaborative peer-to-peer E-learning platform has been proposed. In this model, the integrity, authenticity, non-repudiation and traceability of E-learning resources are ensured by using Blockchain platform. Also, RESTful web service and Go/Java programming language will be used as tools to implement this model. A key metric is measured to evaluate the proposed model: *average response time.* To increase the accuracy, some experiments (144) have been carried out. The same experiment is conducted in two comparatively different network *environment*: Local Area Network (LAN) and Cloud Web Service (such as Amazon Web Service). LAN running environment represents the optimal condition while Cloud environment stands for the actual condition in the real world. When the number of clients in my proposed E-learning platform is relatively small (consisting of one to thirty concurrent clients interacting with E-learning resources), the average response time in the LAN environment is much faster (nearly 1.5 times) than that in Cloud environment. Nevertheless, when the number of clients is on a large scale, the difference of average response time between this two environment becomes insignificant. Besides, adding servers in both environments can increase the horizontal scalability. Furthermore, adding servers in Cloud environment can boost the system performance dramatically. However, extending the delay could have an impact on the system performance but negligible.

ACKNOWLEDGEMENTS

Here, I would like to thank all the people, who contributed in various ways to the research work described in this thesis.

First, I would love to show my profound appreciation and thanks to Dr. Ralph Deters, who has supported me during my research. His knowledge and continuous support were important. Next, I am willing to express my appreciation to graduate chair Dr. Julita Vassileva, who also gave me much support. Also, I show my appreciation to my committee members: Dr. Jim Greer and Dr. Chen Li. Besides, I would like to thank Billy Rowluck who trusted and supported me to develop Lessons Basket Application and also gave me encouragement for my thesis study. Furthermore, I like to thank all colleagues of the Multi-User Adaptive Distributed Mobile and Ubiquitous Computing (MADMUC) Lab for their friendship and countless support.

Personally, I would love to thank my family members for their selfless love and support. I would also willing to thank my friend (Uurtsaikh Jamsrandorj) who always encourage and support me.

CONTENTS

ABSTRACT ii

ACKNOWLEDGEMENTS iii

LIST OF TABLES vi

LIST OF FIGURES vii

LIST OF ABBREVIATIONS x

CHAPTER 1 1

1.1 Problem Definition 4

1.2 Research Questions 6

1.3 Research Objectives 6

CHAPTER 2 7

2.1 E-learning Platforms 7

2.1.1 Network Models of E-learning Platforms 8

2.1.2 Components in E-learning Platforms 9

2.1.3 Information Security in E-learning Platforms 11

2.1.4 An E-learning Platform Study: Moodle 12

2.2 Access Control 15

2.2.1 Models of Access Control 17

2.2.2 Decentralized Access Control 22

2.2.3 OAuth2.0 24

2.3 Blockchain 28

2.3.1 Distributed System 29

2.3.2 Bitcoin: A peer-to-peer Electronic Cash System 31

2.3.3 Core Technologies and Concepts of Blockchain 32

2.3.4 Bitcoin Protocol 37

2.3.5 Private and Public Blockchain 38

2.3.6 Applications of Blockchain 39

2.3.7 Summary 40

2.4 RESTful Web Services 41

Chapter 3 43

3.1 System Architecture Overview 43

3.2 System Components 47

3.3 Work Flow and Related Methods of Proposed E-learning Platform 49

3.4 Implementation 52

3.5 Summary 63

CHAPTER 4 64

4.1 Experiment 1: Local area network 66

4.2 Experiment 2: Cloud running environment - AWS 72

4.3 Summary 78

CHAPTER 5 83

5.1 Conclusion 83

5.2 Contributions 84

5.3 Limitation and Future work 84

REFERENCES 86

LIST OF TABLES

Table Number Page

Table 1: Descriptions of experiments 64

Table 2: Hardware specifications for experiment 1 67

Table 3: Instance specifications for experiment 2 74

LIST OF FIGURES

Figure Number Page

Figure 1: 2016 worldwide revenue for self-paced E-learning products and services by region 3

Figure 2: Peer-to-peer network 3

Figure 3: Architecture of centralized server E-learning platforms 4

Figure 4: Architecture of peer-to-peer E-learning platforms 5

Figure 5: Centralized model.. 9

Figure 6: Decentralized model 9

Figure 7: General components in an E-learning platform... 10

Figure 8: Proposed design of E-learning authentication system 12

Figure 9: Fall 2016 global LMS market... 13

Figure 10: Moodle demo page 14

Figure 11: Access control and other security services 16

Figure 12: Access matrix 17

Figure 13: Subject-based relationship table 18

Figure 14: Object-based relationship table 19

Figure 15: RBAC model 21

Figure 16: Relationship among RBAC models... 21

Figure 17: Running process of OAuth2.0 25

Figure 18: Procedures of authorization code grant 26

Figure 19: Procedures of implicit grant 27

Figure 20: Gartner's hype cycle for emerging technologies... 29

Figure 21: Blockchain network view 33

Figure 22: Input of a transaction 33

Figure 23: Components of block headers 34

Figure 24: Procedures of Bitcoin protocol 38

Figure 25: System architecture 45

Figure 26: Detailed system architecture.. 46

Figure 27: E-R model of E-learning platform.. 50

Figure 28: Worldwide smartphone sales to end users by operating system in 1Q17................................. 52

Figure 29: Code snippet for adding lesson.. 54

Figure 30: Screenshot of creating E-learning resource 54

Figure 31: GUI of editing access control system.. 55

Figure 32: Code snippet of generating JSON string 56

Figure 33: Code snippet of assigning lessons 57

Figure 34: Code snippet of publishing item... 58

Figure 35: JSON files stored in Multichain Blockchain platform 58

Figure 36: First screen page of lesson basket 59

Figure 37: Lesson list screen page 61

Figure 38: Code snippet of sending request... 61

Figure 39: Code snippet of receiving request 62

Figure 40: Lesson video screen page 63

Figure 41: Technological stacks 65

Figure 42: Experimental setup for experiment 1 in LAN 67

Figure 43: Average response time in creating the E-learning resource in the LAN without delay.............. 68

Figure 44: Average response time in consuming the E-learning resource in the LAN without delay............69

Figure 45: Average response time in creating the E-learning resource in the LAN with 250ms delay…......69

Figure 46: Average response time in consuming the E-learning resource in the LAN with 250ms delay...70

Figure 47: Average response time in creating the E-learning resource in the LAN with 500ms delay..........71

Figure 48: Average response time in consuming the E-learning resource in the LAN with 500ms delay..72

Figure 49: Experimental setup for experiment 2 on AWS... 73

Figure 50: Average response time in creating the E-learning resource on AWS without delay.....................74

Figure 51: Average response time in consuming the E-learning resource on AWS without delay................75

Figure 52: Average response time in creating the E-learning resource on AWS with 250ms delay.............75

Figure 53: Average response time in consuming the E-learning resource on AWS with 250ms delay..........76

Figure 54: Average response time in creating the E-learning resource on AWS with 500ms delay.............77

Figure 55: Average response time in consuming the E-learning resource on AWS with 500ms delay......77

Figure 56: Average response time comparisons between LAN and AWS in creating the E-learning resource without delay for single server..78

Figure 57: Average response time comparisons between LAN and AWS in consuming the E-learning resource without delay for single server..79

Figure 58: Average response time comparisons between LAN and AWS in creating the E-learning resource without delay for peer-to-peer (3 nodes)..80

Figure 59: Average response time comparisons between LAN and AWS in consuming the E-learning resource without delay for peer-to-peer (3 nodes)..81

LIST OF ABBREVIATIONS

ACL	Access Control List
AWS	Amazon Web Service
BTC	Bitcoin
C/S	Client/Server
DAC	Discretionary Access Control
DBMS	Database Management System
E-cash	Electronic Cash
IETF	Internet Engineering Task Force
IOT	Internet of Things
ISO	International Standard Organization
ISP	Internet Service Provider
JSON	JavaScript Object Notation
LMS	Learning Management System
MAC	Mandatory Access Control
MRM	Master Reference Monitor
OAuth2.0	Open Authorization 2.0
P2P	Peer-to-Peer
PBFT	Practical Byzantine Fault Tolerance
POW	Proof of Work
PTRM	Private Trusted Reference Monitor
RBAC	Role-Based Access Control
REST	Representation State Transfer
URL	Uniform Resource Locator

CHAPTER 1
INTRODUCTION

Over the past two decades, E-learning systems which are often referred to Learning Management Systems (LMS) have been rapidly growing and have dominated the Internet-based education [1]. According to the E-learning market trends and forecast [2], the state of the E-learning market globally continues to shift, grow, and evolve. Figure 1 shows the global revenue for self-paced E-learning products and services by region. Among the mainstream and commonly used E-learning platforms, the majority of them are based on the centralized system and conform to C/S (Client/server) model. However, such kind of system is subject to central server attack. Its fault tolerance is low. Besides, its scalability is low which means it is tough to accommodate the growing number of nodes connecting to the central server. Even worse, the network efficiency of the centralized system is relatively low. It depends on the processing capability of the central server to a great extent. If there are plentiful requests coming into the server at the same time, the latency will be extremely high which means the network efficiency will decrease accordingly. By contrast, decentralized systems release the burden of the central server and take advantages of every participant nodes' processing capability. Meanwhile, the scalability of decentralized systems is comparatively higher than centralized ones. In E-learning platforms, such kinds of superiority could be enlarged. The number of users is always large and there are growing number of users joining the platform. Thus E-learning platforms need high scalability. Meanwhile, there could be a mass of requests coming from different users in a specific time such as class registration or online class hours. Therefore, the central server would suffer from high burden if C/S model is used. Instead, decentralized systems play its full role in this situation. Peer-to-peer network as shown in Figure 2 which is by the decentralized model has been raised. More and more research involves in exploring the model of building collaborative E-learning platforms on a peer-to-peer network in which E-learning resources could be managed and shared among nodes.

When building a decentralized collaborative E-learning platform based on a peer-to-peer network, ensuring the information security has become crucial [3]. To keep the security and privacy of E-learning resources, access control model is introduced. Access control model was conceptualized by Sandhu et al. [4]. It enforces controls into the resources to access after authentication. The purpose of access control is to limit the actions or operations that a legitimate user can perform

[4]. There are three models of access control: Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role-based Access Control (RBAC). Nowadays, there are numerous validation and authorization methods based on the access control models. OAuth has become one of the globally recognized authorization methods on top of access control models. It is not hard to apply access control model into the centralized network but in a peer-to-peer environment, it becomes incredibly complicated and insecure because there is no central authority.

In 2008, a groundbreaking paper *Bitcoin: Peer-to-Peer Electronic Cash System* [5] written under the pseudonym Satoshi Nakamoto was published and attracted a myriad of researcher's attention. In this paper, a distributed consensus mechanism has been proposed which is regarded as the rudiment of the Blockchain technology. In its original release version, it was only used for cryptocurrencies in financial ecosystems known as Blockchain1.0. Later, smart contracts have been introduced in Blockchain2.0. Currently, Blockchain3.0 has moved from building only economic systems to developing various applications in different fields. Even though the legitimacy of the Bitcoin system is controversial, it indeed brings an ingenious solution to the problem of Byzantine and double spend problems in distributed systems. Additionally, when applying Blockchain into building the decentralized access control model in a peer-to-peer E-learning platform, POW (Proof-of-Work) [5] algorithm makes it possible to manage and share resources without a trusted central authority.

In the following parts, problem definition will be explained first in Section 1.1, the related research questions will be listed in Section 1.2 while the research objectives will be shown in Section 1.3.

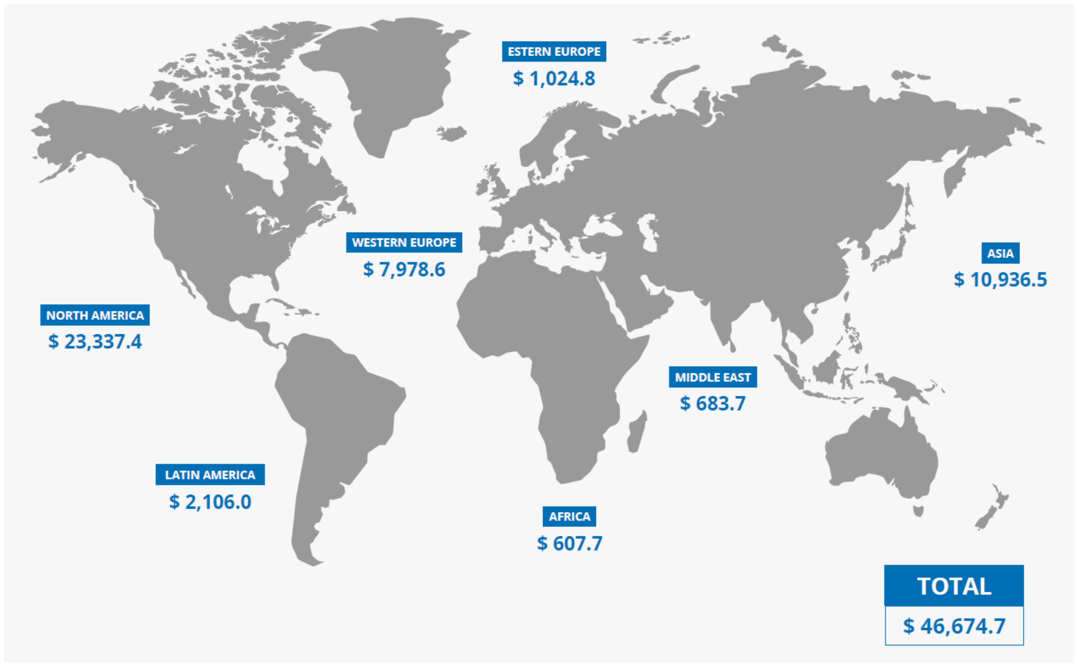


Figure 1: 2016 Worldwide revenue for self-paced E-learning products and services by region (in US$ millions) [2].

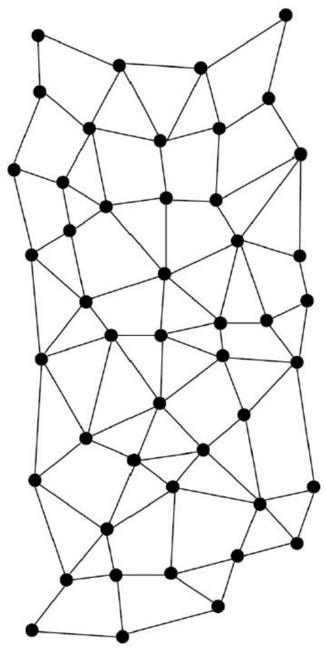


Figure 2: Peer-to-Peer Network

1.1 Problem Definition

Modern E-learning platforms become more and more community-based and collaborative [2] in which E-learning resources are managed and shared within the community. Such kind of dynamically collaborative E-learning platform is in demand for the aboriginal community of northern Saskatchewan since local education resource is scarce. However, traditional centralized server architectures (shown in Figure 3) must be replaced due to the following reasons:

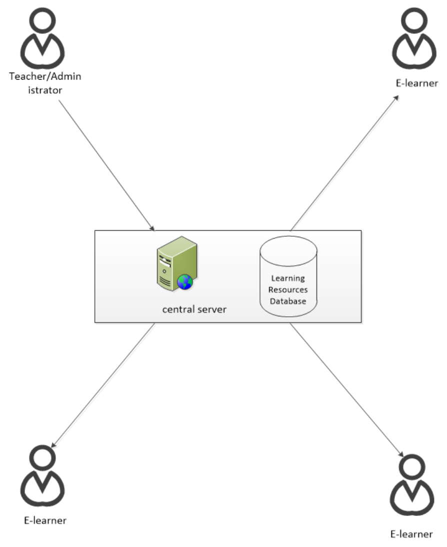


Figure 3: Architecture of centralized server E-learning platforms

a) **Single point of failure-** if the centralized server fails, the whole system will fail.
b) **Low scalability**- it cannot scale with the number of participants growing.
c) **Lack of community-based access control-** there is no access control system which can enable the members of the community manage their E-learning resources dynamically and independently.
d) **Low performance of the platform**- since all the requests need to be approved by the centralized server, the performance of the platform is limited to the capacity of its server.
e) **High cost and burden of the server-** the cost of maintenance and load are always high.
f) **Low robustness**- the robustness of the system is weak.
g) **Limited transparency**- since all the access control permissions are managed by the centralized server, the transparency of the platform is another serious issue.

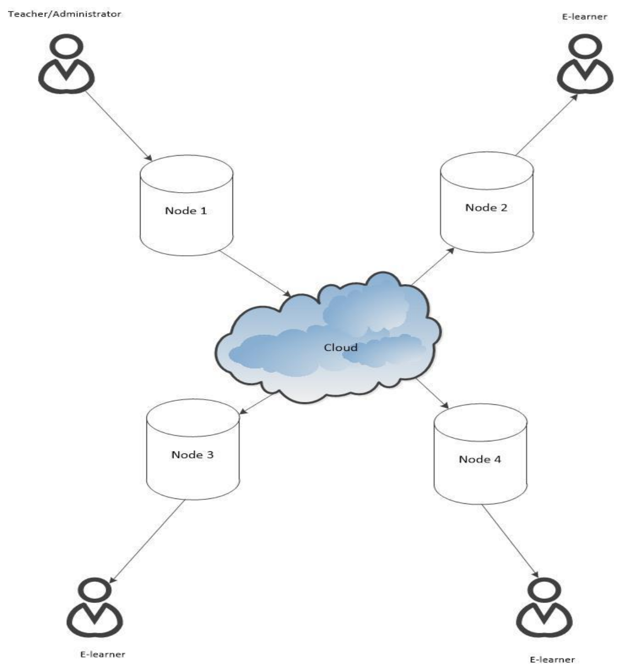


Figure 4: Architecture of peer-to-peer E-learning platforms

Therefore, decentralized access control systems built on peer-to-peer E-learning platforms (shown in Figure 4) are proposed. Traditional access control mechanisms such as DAC, MAC and RBAC are designed for machines under common administrative control and depend on maintaining a centralized database of user identities [7]. They are efficient and secure in the centralized server model. However, they fail to accommodate to rapidly growing users with the size of system enlarging. Even though some research has involved in building decentralized access control systems in various fields [6-10], most of them are proved to have one or more of the following deficiencies:

a) **Vulnerable**
b) **Slow**
c) **Highly complicated**
d) **Low horizontal scalability**

Thus, building a simple and secure decentralized access control system with high performance and horizontal scalability in a peer-to-peer E-learning platform is a significant and challenging issue.

In this research, Blockchain, a peer-to-peer distributed ledger that is cryptographically secure, append-only, immutable and updatable only via consensus or agreement among peers is adopted as a tool to solve the research questions.

1.2 Research Questions

Based on the problem defined and explained above, two questions are listed.

1. How to improve the performance of access control system on a peer-to-peer E-learning platform by the decentralized solution?

2. How to increase the horizontal scalability of decentralized access control system on a peer-to-peer E-learning platform?

1.3 Research Objectives

The objectives of this research are:

1. To build a server-less decentralized access control solution based on autonomous peer-to-peer network model by using Blockchain to solve question 1.

2. To apply Blockchain technology to create a simple and secure decentralized access control system in a peer-to-peer E-learning platform.

3. To build a Blockchain based solution to increase the horizontal scalability of decentralized access control system on a peer-to-peer E-learning platform to solve question 2.

CHAPTER 2
LITERATURE REVIEW

The primary objective of this research is to apply Blockchain technology to build a simple and secure decentralized access control system in a peer-to-peer E-learning platform. From the perspective of both administrators and E-learners, the platform should provide a user-friendly, information security environment. Some research has been conducted related to access control of E-learning platforms but there is limited research focusing on both providing a decentralized solution and highlighting the access control.

This chapter provides a detailed background. In this chapter, all concepts that are closely related to my research are discussed and reviewed. Background to the E-learning, access control, Blockchain and RESTful web services are discussed. Furthermore, the applications of the respective concepts have also been explained.

2.1 E-learning Platforms

According to Asmaa et al. [11], E-learning is one of the fastest growing markets in the virtual world. E-learning platforms are aimed to provide platforms for different users to learn online. Regarding the definition of E-learning platforms, it is a flexible term used to describe the newest method of teaching throughout the online internet technology [11]. It is also called Learning Management System (LMS). It provides suites of tools that support online course creation, maintenance, student enrollment and management, education administration, and student performance reporting [1]. It can be divided into two categories: Open-source platforms and Proprietary solution-based platforms. For Open-source platforms, they are typically built on extensible frameworks that let implementers adjust and modify the systems to suit their specific needs [1]. There are plenty of successful and innovative platforms such as Moodle (www.moodle.org), Sakai (www.sakaiproject.org). In one of the following sections, Moodle would be explained in detail because it is one of the most widely-used E-learning platforms around the world. For proprietary solution-based platforms, they have not been widely adopted. The examples of such kind of platforms are WebCT/Blackboard (www.blackboard.com) and Gradepoint (www.gradepoint.net). Considering the mainstream of E-learning platforms are Open-source ones, this research would mainly focus on Open-source platforms.

Nowadays, E-learning platforms experience two generations. From roughly 1993 to 1999, the first generation of E-learning platform came out. It focused on the delivery and interoperability of content designed for a specific purpose [1]. From roughly 1999 on, the second generation of E-learning platform inherited the first generation's advantages and focused on its failures. It not only addressed sharing contents but also sharing learning objects, sequences of learning objects and learner information [1]. Even though the second generation of E-learning platforms achieves great success in the online Education market and there are plenty of widely-used Open-source E-learning platforms dominating the E-learning industry, most of them are not entirely learner-centric [1] which means that they mostly emphasize on learning administration rather than on the learners. Thus, the next generation of E-learning platform should focus more on the learners.

2.1.1 Network Models of E-learning Platforms

Regarding the majority of E-learning platforms, they are based on the centralized model as shown in Figure 5. There is a centralized server which can handle all the requests from different users. There are three main problems existing in this model:

a) The scalability is low which means that it cannot accommodate the rapidly growing number of nodes connecting the single server.
b) Too much burden will be put on the server. The cost of maintenance would become incredibly high.
c) The network efficiency would become low as the number of connecting nodes is increasing.

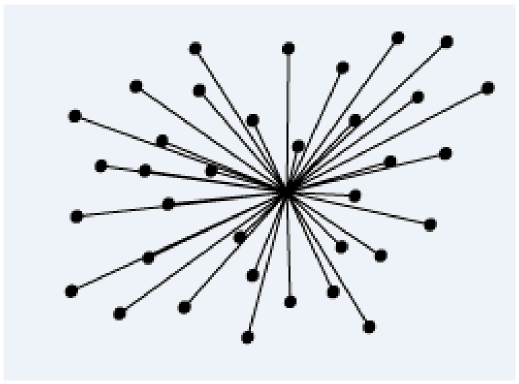


Figure 5: Centralized model

Compared with the centralized model, the decentralized model distributes the central server's burden into the participant nodes as shown in Figure 6. It overcomes the disadvantages of the centralized model. To make learners feel supported by others and contact mutually, a peer-to-peer (decentralized) network is appropriate.

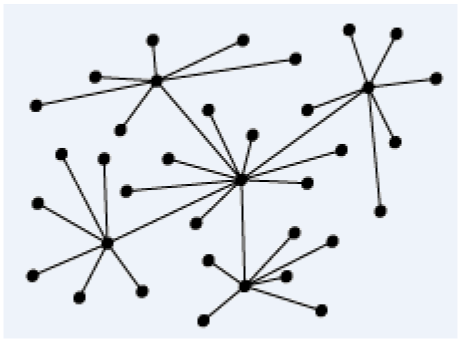


Figure 6: Decentralized model

2.1.2 Components in E-learning Platforms

Network architecture (or model) has been explained in the previous section, another question may arise: What is inside an E-learning platform or what is the components of an E-learning platform. According to Ouadoud et al. [12], general components of an E-learning platform could be shown in Figure 7. In a complicated E-learning platform, there are five roles: **Coordinator**, **Learner**, **Tutor**, **Teacher and Administrator**.

a) Coordinator guarantees the normal operation of the whole platform.
b) The learner is the central role in this platform who can download Educational Resources from the Computer System.
c) Tutor monitors and supports each Learner by communicating with learners when they are in trouble.
d) The teacher creates and manages the Education Resources.
e) The administrator takes charge of maintaining and customizing the platform.

In this research, components and roles of the proposed E-learning platform would be covered in System Architecture part in Chapter 3.

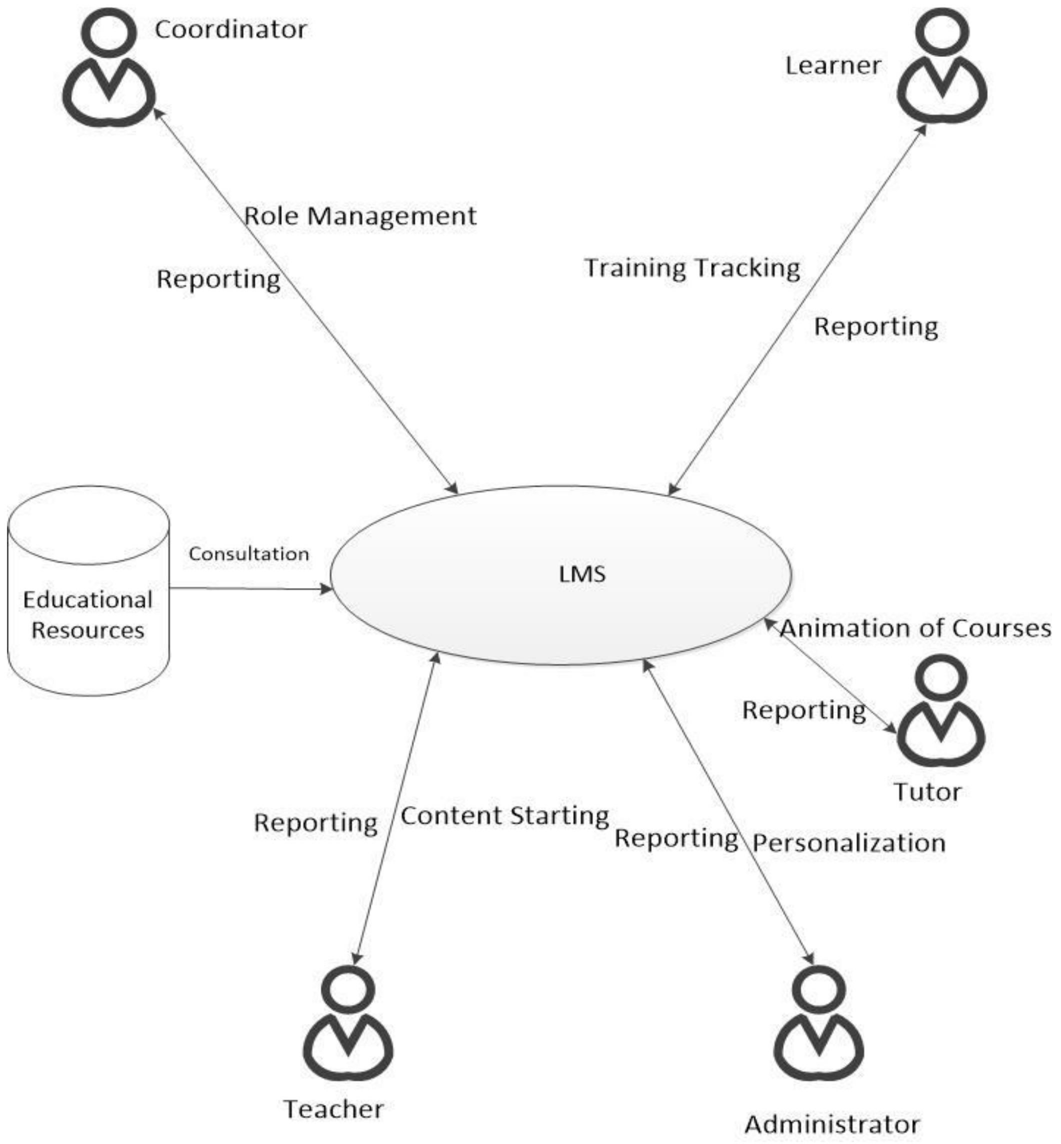


Figure 7: General components in an E-learning platform [12]

2.1.3 Information Security in E-learning Platforms

E-learning platforms, in essence, provide an Education platform in the Internet context. The internet has become the venue for a new set of illegal activities [13], so E-learning platforms are exposed in such insecure environment. Regarding E-learning platforms in Online Education market, most of them have valued course development and delivery [13] while little consideration to the privacy and security has been taken. Although some of the Open-source E-learning platforms have realized the importance of the security and they have been improving it by releasing different versions continually, meeting the increasingly strict security requirements needs more explorations and efforts. Previous research has indicated that information security in an E-learning platform is an extremely complicated problem. Accordingly, there are plenty of research exploring the efficient way of meeting the security requirements to solve this problem. However, the majority of them have more or fewer vulnerabilities causing various security issues.

Information security in E-learning is a broad issue mainly including authentication, encryption system and access control. In this research, access control would be discussed emphatically.

According to Kang et al. [14], authentication is very crucial in E-learning platform because user information is needed to be secured. In a broad sense, authentication refers to the process of verifying the truth of a property from a piece of data claimed true by an entity. In an E-learning platform, user authentication ensures the identity of the user against impersonation or fabrication [14]. It is aimed at preventing unauthorized access by malicious users [15]. Recently, there are a growing number of papers focusing on this issue. Regarding the methods of user authentication, they have been classified into three categories by Asha et al. [15].

a) Methods based on human memory like a password.
b) Methods based on physical devices such as IC card.
c) Methods based on biometrics like a fingerprint, eye recognition.

In the methods above, the first two are vulnerable to forgetfulness and losses. Only the last category is reliable and safe. As far as biometrics user authentication is concerned, Kang and Kim [14] have proposed a module called Access Learning module in which learners could be authenticated through eye or face recognition detected by digital devices continuously. This real-time monitoring authentication system is running while the session is ongoing [14]. Figure 8 shows the design of this module. It is noteworthy that access control happens during the authentication time when the learner is granted all necessary rights [16]. User authentication is closely related to access control which will be explained in Section 3.2.

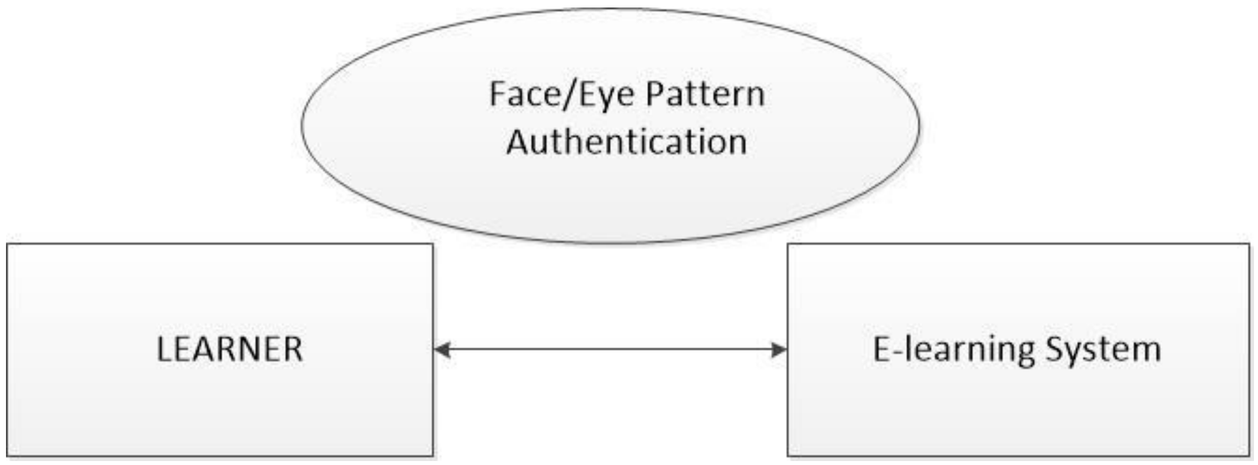


Figure 8: Proposed design of E-learning authentication system [14]

Apart from authentication, the encryption system is another vital issue in an E-learning platform. To ensure the confidentiality, encryption system has been raised. In cryptography, encryption refers to the process of encoding a message or information in such a way that only authorized parties can access it [17]. It is commonly used to protect the confidentiality of Education Resources stored in an E-learning platform. Modern encryption system should provide not only confidentiality protection but also support the following features:

a) Authenticity: the origin of the Education Resource should be traced.
b) Integrity: prove that the content of the Education Resource has not been modified since it was sent.
c) Non-repudiation: the sender of the Education Resource cannot deny sending it.

In an E-learning platform, the information security level largely depends on the performance of the encryption system. Therefore, developing a secure encryption system is indispensable when designing an E-learning system. In addition to confidentiality protection, another role of the encryption system is to protect both learners and administrators privacy. Previous research shows that existing E-learning standards provide some rules for privacy protection but it remains unsatisfactory on several levels [18].

2.1.4 An E-learning Platform Study: Moodle

In the last section of 3.1, an instance of E-learning platform would be introduced. In LMS market, there are numerous successful and widely-used E-learning platforms such as ANGEL, Blackboard, Moodle, Canvas. According to a recent report from Edutechnica [19], Moodle remains the most popular LMS in Canada as shown in Figure 9.

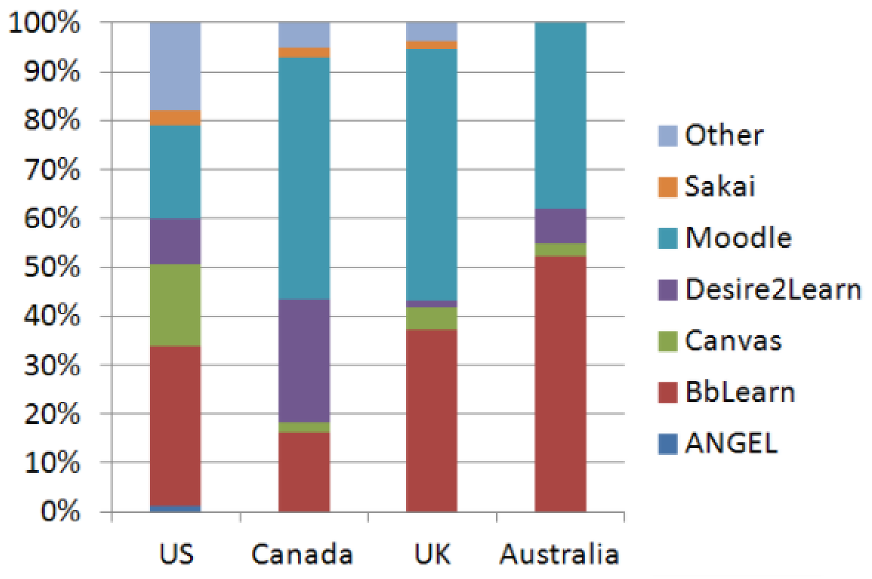
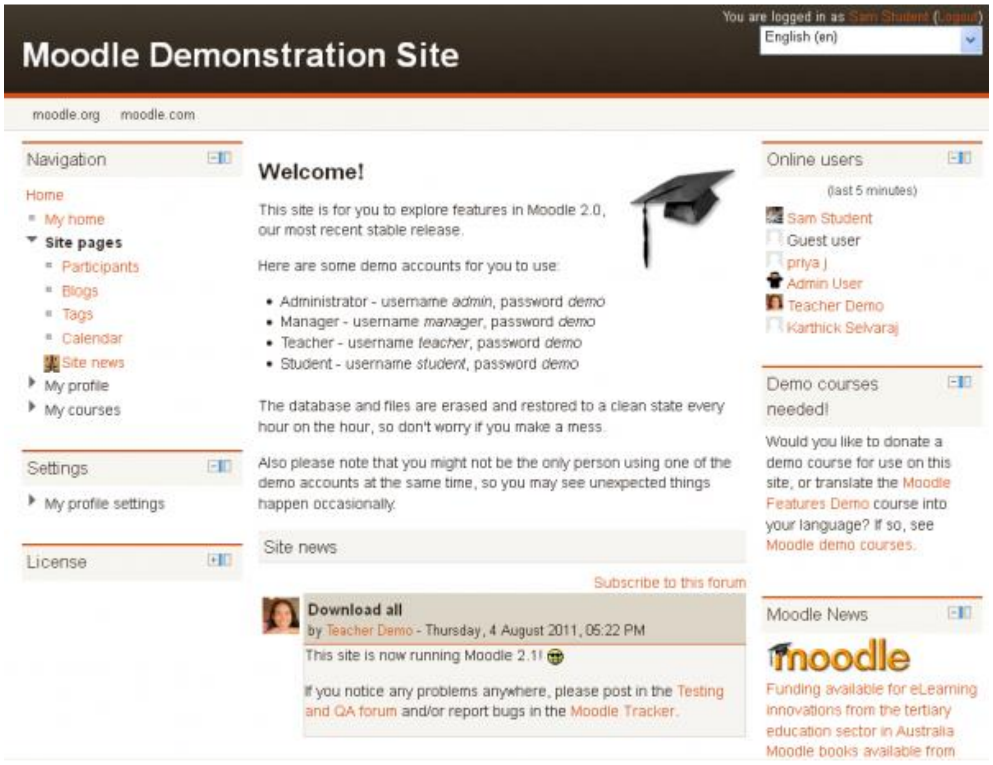


Figure 9: Fall 2016 global LMS market [19]

Moodle which stands for Modular Object-Oriented Dynamic Learning Environment is an Open-source platform aimed at providing the perfect E-learning solution for developers. It was created in 2001. It is a free software (for individual usage) initially initiated by Moodle HQ from Perth and later maintained by a community of users on Moodle. It gives the opportunity of developing Internet-based courses and proper support for security and administration [20]. The source code is written in PHP language and the corresponding database is MySQL and PostgreSQL. Figure 10 shows the demo page of Moodle. Currently, the version of Moodle is 3.3 released on 15 May 2017.


You are logged in as Sam Student (Logout)
English (en)

Moodle Demonstration Site

moodle.org moodle.com

Navigation
Home
- My home
- Site pages
  - Participants
  - Blogs
  - Tags
  - Calendar
  - Site news
- My profile
- My courses

Settings
- My profile settings

License

Welcome!

This site is for you to explore features in Moodle 2.0, our most recent stable release.

Here are some demo accounts for you to use:

- Administrator - username *admin*, password *demo*
- Manager - username *manager*, password *demo*
- Teacher - username *teacher*, password *demo*
- Student - username *student*, password *demo*

The database and files are erased and restored to a clean state every hour on the hour, so don't worry if you make a mess.

Also please note that you might not be the only person using one of the demo accounts at the same time, so you may see unexpected things happen occasionally.

Site news

Subscribe to this forum

Download all
by Teacher Demo - Thursday, 4 August 2011, 05:22 PM
This site is now running Moodle 2.1!

If you notice any problems anywhere, please post in the Testing and QA forum and/or report bugs in the Moodle Tracker.

Online users
(last 5 minutes)
Sam Student
Guest user
priya j
Admin User
Teacher Demo
Karthick Selvaraj

Demo courses needed!
Would you like to donate a demo course for use on this site, or translate the Moodle Features Demo course into your language? If so, see Moodle demo courses.

Moodle News
moodle
Funding available for eLearning innovations from the tertiary education sector in Australia
Moodle books available from

Figure 10: Moodle demo page

Moodle is a robust course management E-learning platform covering the following features: creating courses, assignments, quizzes, documents. There are plenty of modules supporting students and teachers to interact with each other. For example, in chat module, students can chat with a teacher personally or they can also join a group chat with other students.

Although Moodle has achieved great success in Canada, there are some defects affecting its market competitiveness. First and foremost, Moodle is based on the traditional C/S model which means that it has a centralized server. As mentioned in Section 3.1.1, the scalability and stability of such platform are low while the maintenance cost for the centralized server is relatively high. With the number of learners soaring, the network efficiency would become low. Secondly, information security has become a complicated issue since Moodle was released in 2001. Even though Moodle has experienced numerous updating and improvements, some security flaws are being exposed to untrusted internet environment. For example, *brute force attack* [13] is one of the most severe security problems in Moodle. It refers to sending a myriad of requests to the server to guess the right log in the profile of a learner. It is often used by criminals to crack encrypted data or by security, testers to test a website's network security. Another design flaw in Moodle is that it is tough to detect if the file (or Education Resource) has been modified illegally since it was created. According to the researchers, logic vulnerabilities would exist in large-code based E-learning platforms. Regarding Moodle, it has over 200 million lines of codes suffering from several kinds of security Loophole Attacks such as an Object Injection and a double SQL injection [21]. If the hackers want to change the score of the students, they can get authority of an administrator or a teacher by using *brute force attack* or *session hijacking.* Then they could get access to the score files on Database and modify some student's scores by using SQL injection. In this scenario, two fatal problems concerning authority management and access control emerge. Lack of authority management would destroy the security of E-learning platform. An attacker may obtain superior authority through illegal attacks and gain access control to sensitive files. Besides, defectiveness of access control may cause irredeemable consequence.

Leaning on the pros and cons of Moodle, an innovative method will be proposed in this research. In next chapter, a decentralized solution using Blockchain to build E-learning platform will be introduced.

2.2 Access Control

With the development and improvement of Computer Networking and Information Technology [11], a growing number of sophisticated and user-friendly E-learning platforms dominate the education market. Modern E-learning platforms not only support teaching and learning but some intelligence interaction among the collaborative team members [22]. Regarding such a sophisticated E-learning platform, Authority management and data security have become focus issues. Education resources which contain sensitive data such as learner's profiles or lesson information must be protected carefully. To prevent the reveal of unauthorized information or illegal modification of private data, various security services have been applied. According to International Standardization Organization, ISO7498-2 [23] has defined five security services: Authentication, Access control, Data confidentiality, Data integrity and Non-repudiation. In this section, access control would be explained in detail.

Access control is not an isolated concept and it always coexists with other security services including Authentication and Auditing [4]. It enforces controls into the resources to access after authentication. According to Sandhu et al. [4], the purpose of access control is to limit the actions or operations that a legitimate user can perform. As shown in Figure 11, the Reference Monitor who consults an Authorization Database with the purpose of ensuring if the user attempting to do is authorized plays a central role. Security Administrator has set authorization Database by security policies of the organization. It is also worth noting that Auditing happens during the whole process to monitor and keep a record of relevant activities [4].

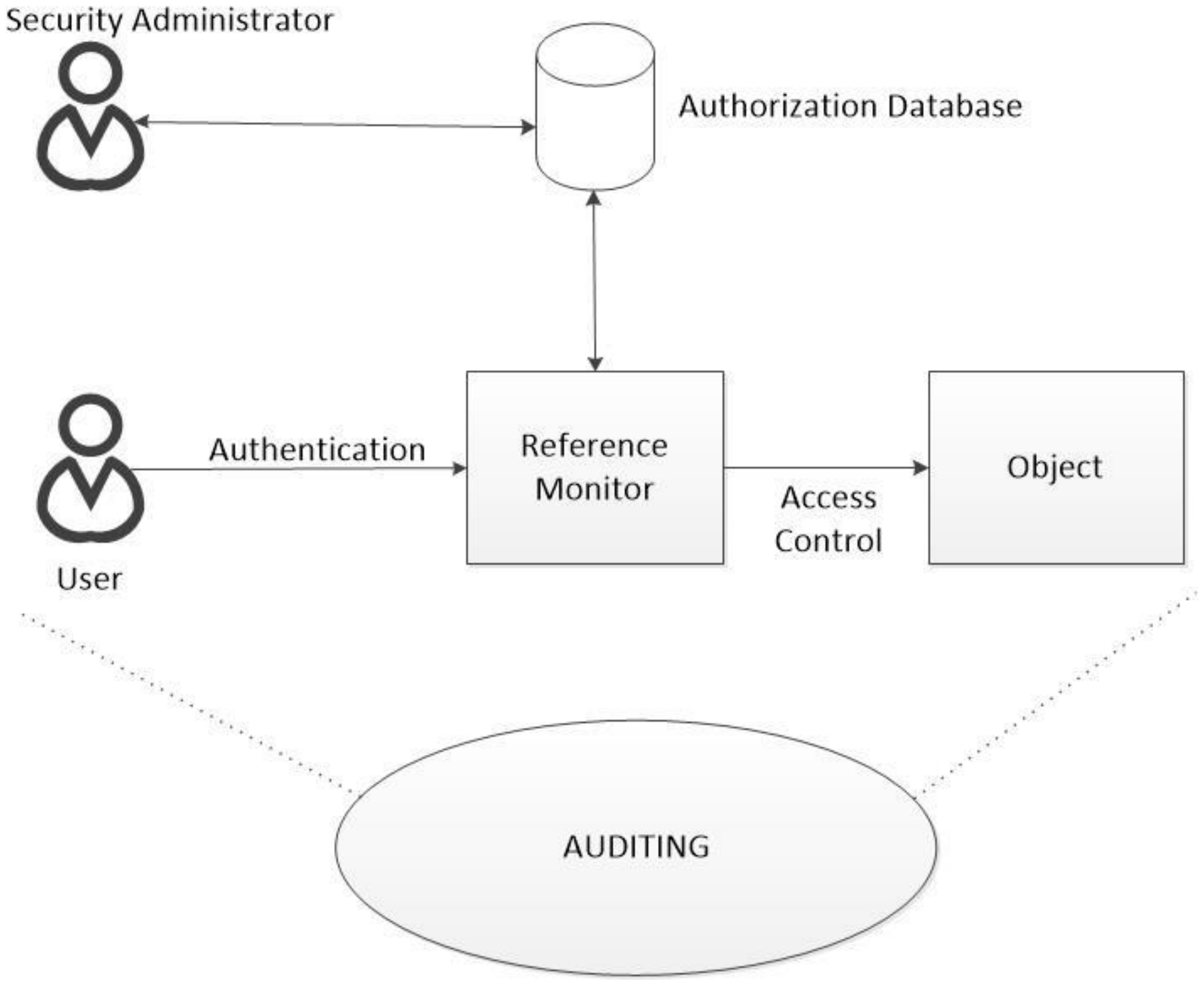


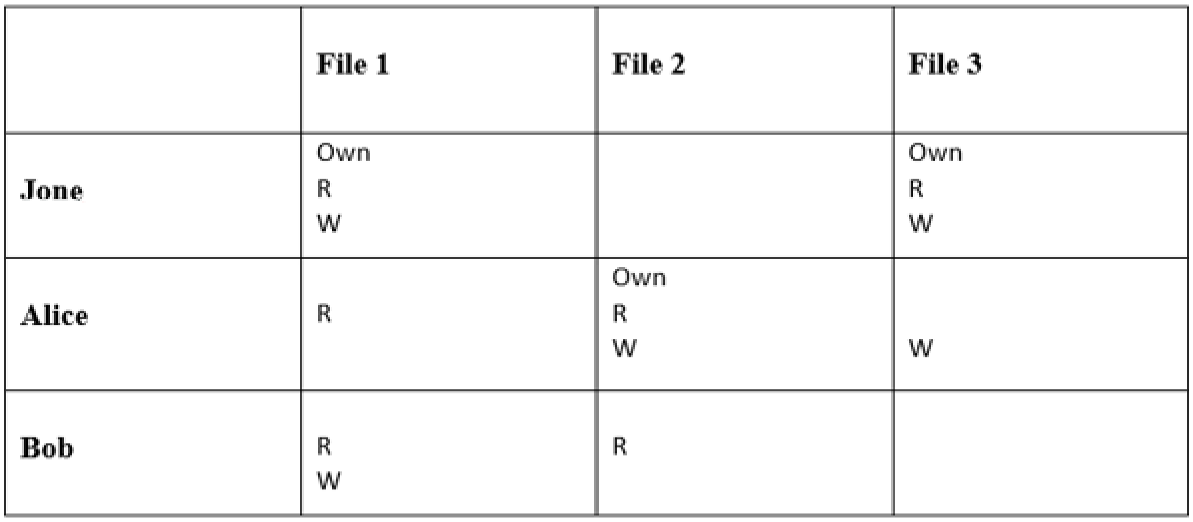
Figure 11: Access control and other security services [4]

In a primary access control system, there are four components as follows:

a) Subject: the user or program initiating the operation of access.
b) Object: accessed resources.
c) Operation: the specific behavior initiated from Subject to Object.

d) Constraint: a set of rules which confirms the access authority.

Based on the components, the Access Matrix has been proposed [4] as shown in Figure 12. The row represents the Subject while the column represents the Object. In each cell of the Matrix, the respective authorities have been listed. There are three kinds of rights: R represents read, W represents write and Own represents ownership of the file. The first two are self-evident and the last one means the subject of the file could grant Read or Write right to other subjects. In the large-scale system, this kind of Matrix would become so complicated. Design of such Access Matrix would become tedious. So ASL (Access Control List) [4] has been proposed to relieve the work.



	File 1	**File 2**	**File 3**
Jone	Own R W		Own R W
Alice	R	Own R W	W
Bob	R W	R	

Figure 12: Access matrix

2.2.1 Models of Access Control

Access control originates from the 1970s and it has been applied to various fields in Information System [24]. The ultimate objective of access control is to solve the security and flexibility of authorization management. To achieve this goal, there are all kinds of models of access control being raised by researchers. Three mainstream models will be explained there-in-after.

Discretionary Access Control (DAC) is a kind of traditional and widely-used access control model which is a means of restricting access to objects based on the identity of subjects and groups to which they belong [25]. It is also called Selective Access Control [26]. In DAC, subjects can independently grant the authorities to any other subject. In some cases, one file could be shared between different subjects. The basic idea of DAC is to build the relationship between Subject and Object. According to McCollum [27] et al., there are two forms of tables reflecting such relationship: Subject-based one and Object-based one. Regarding Subject-based relationship table, a list of objects will be linked to the specific subject as Figure 13 shown. Similarly, Object-based relationship table lists the object in each row linking with respective subjects as Figure 14 shown. Similarly. An advantage of DAC lies in the flexibility of authorization Management which makes it perfect for various kinds of systems. For example, in Windows or UNIX operating systems, DAC has been applied to manage files and mailboxes. However, it is not efficient in distributed systems due to vast memory space for the table and low efficiency of the searching algorithm. Moreover, the security level of DAC is pretty low and it is vulnerable to Trojan Horse [28].

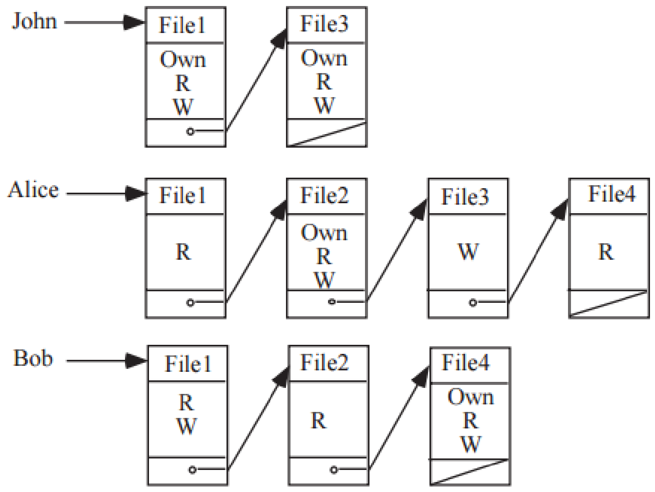


Figure 13: Subject-based relationship table [4]

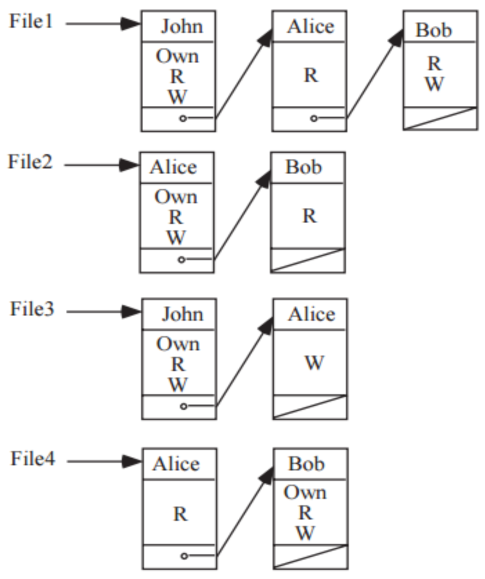


Figure 14: Object-based relationship table [4]

Mandatory Access Control (MAC) is another model in access control and it was represented by the Bell-La Padula model [29]. Every subject and object must be labeled by security level [30]. Here, security level means the trust level of a user (subject) or the level of sensitivity for information (object). For instance, the level of sensitivity for a file could be classified as follows: top secret, secret, confidential or public. The trust level of a subject reflects the degree of trust by Certificate Authority while the level of sensitivity could reflect the security level of the object. According to Sandhu [31], such security levels are partially ordered and from a lattice. To distinguish the security level, the label on the subjects is often called *clearance* while the label on the objects is often called *security classification* [29]. When applying MAC, there are two mandatory rules:

1) No read up: Subject s can read Object o only if $\lambda(s) \geqslant \lambda(o)$
2) No write down: Untrusted Subject s can write Object o only if $\lambda(s) \leqslant \lambda(o)$

Note that whether the subject is trusted is determined by the system which is also called Certificate Authority. It means that the subject could not get access right to the object if the system denies the access by checking the subject's clearance and the object's security classification. This behavior is mandatory and no one can intervene [32]. Thus, MAC ensures the stability of Authority Management and it is resistant to Trojan Horse. Nevertheless, the deficiency of MAC consists in the unexpectedly heavy workload for system administrators on a large scale system. Referring to commercially successful access control systems, DAC and MAC are always combined. In this case, if a subject wants to visit an object, it must pass the check of both DAC and MAC. MAC makes up the lacks of DAC and provides a second shield for the system. One of the famous models based on such combination is the Chinese Wall Security Model [33].

Role-based Access Control (RBAC) stems from multi-user and multi-application online systems in the 1970s [34]. Here, "Role" originates from the concept of System Group in Unix and Permission Group in Database Management System (DBMS). A great deal of research focuses on developing access control models based on RBAC. Sandhu et al. [34] proposed a series of models called RBAC and these models gain a lot improvement in their later research [35]. RBAC covers plenty of issues such as roles, role hierarchies, role permissions, constraints. According to Sandhu [34], a role can be defined as a set of actions and responsibilities associated with a particular working activity. As Figure 15 shown, role serves as the bridge between User and Permission. In MAC and DAC, users are assigned with specific permissions. By contrast, in RBAC, users are assigned to specific roles and each role is assigned to specific permissions. As a whole, RBAC has five following features:

1) The role is the central part of RBAC: the role assigned to a user determines the permissions and the operations for this user.
2) Role hierarchies: a natural means for structuring roles to reflect an organization's lines of authority and responsibilities [34], "parent" roles include all the permissions of "children" roles.
3) Least privilege: only those permissions required to perform the task are assigned to the role
4) Separation of duties: mutually exclusive roles must be invoked to perform a task.
5) Role capacity: assign the capacity when creating a new role.

According to Sandhu [34] et al., constraints are a powerful mechanism for laying out higher-level organizational policy. After introducing essential components in RBAC, a family of RBAC reference models should be stated. $RBAC_0$ is the base model among a series of RBAC models which states the minimum requirement to meet RBAC model. $RBAC_1$ and $RBAC_2$ both inherit from $RBAC_0$ and they have added their features respectively. To be specific, $RBAC_1$ adds role hierarchies while $RBAC_2$ adds constraints. It is worth noting that $RBAC_1$ and $RBAC_2$ are incomparable but $RBAC_3$ consolidates their advantages. The relationship among RBAC family reference models can be shown in Figure 16. In Sandhu et al.'s later research [36], a standard called NIST was proposed to give directions to develop access control management on RBAC models.

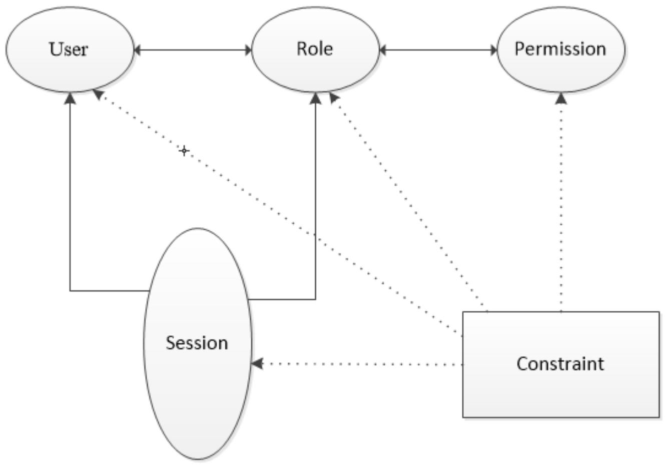


Figure 15: RBAC model

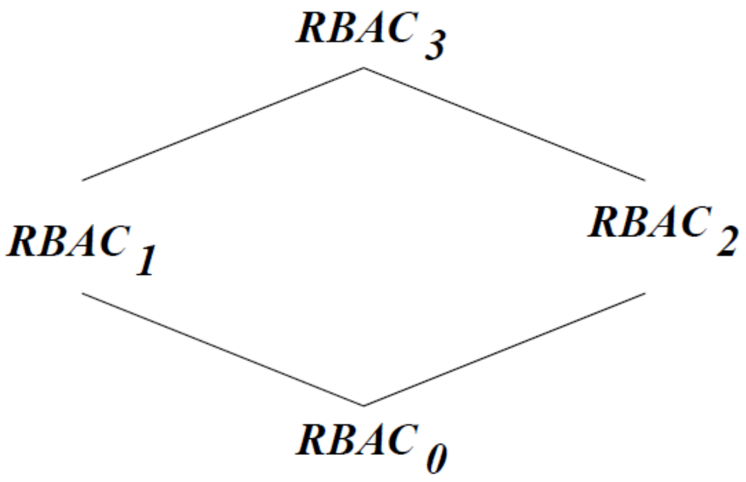


Figure 16: Relationship among RBAC models [4]

In summary, there are three primary access control models and they have their suitable places. The authorization management in DAC is flexible and the subject determines it. However, the security level is low and it is vulnerable to Trojan Horse. For most pint-sized systems, it is appropriate to apply DAC. The security level in MAC is high. It is immune to Trojan Horse. However, the workload to implement it is too heavy. For most military systems, MAC is the best choice. RBAC combines the advantages of both DAC and MAC. Meanwhile, it overcomes the disadvantages of DAC and MAC by adding least privilege, separation of duties and role capability. Nevertheless, the fatal weakness in RBAC lies in the fact that a user could be assigned to multiple roles and the permissions granted to this user are the union set of the roles. It will cause the problem of excess permissions [37].

2.2.2 Decentralized Access Control

The field of access control covers solutions to the problems of authorization, validation and authentication. In the previous section, fundamental issues about traditional access control have been introduced and three kinds of access control models have been listed to solve these three problems. However, traditional access control mechanisms are designed for machines under common administrative control and depend on maintaining a centralized database of user identities [7]. They fail to accommodate to rapidly growing users with the size of system enlarging. As mentioned in section 3.1, E-learning platform is growing fast and the number of users is increasing dramatically. For this reason, traditional access control models could not adapt to such development tendency. In the last decade, mounting research has focused on proposing access control mechanisms based on a decentralized model. Balasubramanian et al. [6] have proposed a framework for decentralized authorization for physical access control. Miltchev et al. [7] provided a survey of decentralized access control mechanisms in the large-scale distributed file systems and analyzed the experimental result. Ruj et al. [8] proposed a novel decentralized access control scheme for securing data stored in clouds supporting anonymous authentication. Ouaddah et al. [9] have provided a broad review of different access control solutions in IOT (Internet of Things) based on the decentralized model. However, most of them are proved to have one or more of the following deficiencies: vulnerable, slow, so complicated or with low horizontal scalability. In brief, decentralized access control is still in the pilot stage which has a bright future to apply to various kinds of fields.

As mentioned before, traditional access control models can be applied to C/S applications in which both protect sensitive objects and reference monitor are stored at the server side [37]. However, it is requisite to enforce access control mechanisms at the client side because it can protect sensitive data [38]. Sandhu et al. [38] proposed an access control mechanism for the P2P environment supported by trusted computing technologies. But according to Han et al. [10], the architecture proposed by Sandhu et al. [38] requires too many trusted components. So a trusted decentralized access control framework called TDAC has been raised [38]. There are two features in TDAC framework:

1) The server's sensitive data is stored in server side while the user's private data is stored in Client side.
2) Two reference monitors have been introduced: one is the privately trusted reference monitor (PTRM) running on the Client side is the master reference monitor (MRM) running on the server side.

Considering trust issue is out of domain in this research, TDAC will not be explained in further detail. Based on the concept of making access control policies on the client side, Balasubramanian et al. [6] proposed a physical access control framework. In this framework, the significant feature consists in supporting the evaluation of policies dependent upon dynamically varying context. It can be used in the following scenario: using smart cards to get access to the room. Furthermore, in traditional implementation of this scenario, when a user shows a card to the reader (here is the card reader in the door), the reader will deliver the card information to the central controller and wait for reply directing whether or not to allow access. In this situation, both reader and access card have no processing ability but the central controller is a high-powered device with fail-over capabilities whose cost is enormous. By contrast, in the framework proposed by Balasubramanian et al. [6], state of the art lies in replacing the central controller with a network of smaller controllers and introducing per-use active devices with processing power and memory. A context modeling mechanism was proposed in which controllers dynamically maintain the context information. In this situation, when a user inserts the card into the card reader, the controller and the smart card come together to execute the policies stored in the card and make the decision based on the context information. The framework proposed by Balasubramanian et al. [6] has relieved the central server's stress. Such information-context based mechanism could maintain the access control dynamically in the decentralized environment. In the next section, an efficient mechanism of access control called OAuth2.0 would be introduced.

2.2.3 OAuth2.0

Previous research shows that the essential problem of Open-source E-learning platform lies in the validation and authorization of users [39]. OAuth is a globally recognized authorization method. The marked feature of OAuth is that users could apply to visit their protected resources without the need to enter their names and passwords in the third-party application [40]. OAuth1.0 was settled in IETF (Internet Engineering Task Force) whose reference number is RFC5894. It indicates that OAuth has become Internet Standard Protocol. In October 2012, OAuth2.0 (the latest version) whose reference number is RFC6749 had been released officially. Compared with OAuth1.0, its operation of validation and authorization is more straightforward and safer. The focus of OAuth2.0 is to simplify the work of developing Client and it can be applied to all sorts of applications including Web, desktop, mobile, etc. It is worth noting that OAuth2.0 has no backward compatibility even though it is the next version of OAuth1.0. A lot of research [39,41] has indicated the bright future of OAuth2.0. According to Ziqing [39], OAuth2.0 will become a standard protocol in the Open-source platform and with the development of OAuth2.0, it will become a protocol family to solve authorization problems in different contexts.

Before understanding the work process of OAuth2.0, it is necessary to introduce the related terms:

1) Authorization server: dealing with authorization issues owned by ISP (Internet Service Provider)
2) Resource server: the server used to store the resource which is submitted by users and ISP also owns it
3) Client: the third-party application requesting to access the protected resources
4) Resource Owner: the owner of the protected resources
5) Protected Resource: the restricted accessed resources controlled by OAuth2.0
6) Authorization Code
7) Refresh Token
8) Access Token

In a broad sense, the solution raised by OAuth2.0 is to separate users from Client by adding an authorization layer. The third-party Client should communicate with authorization layer and acquire the Token (Refresh Token and Access Token) first. Then ISP would send the corresponding user's resources according to the validity and the extent of competence of the Token. Figure 17 shows the specific running process of OAuth2.0. There are six steps:

1) Step A: Client sends a request for authorization to Resource Owner (like a user).
2) Step B: Resource Owner sends a response to Authorization Grant back to Client.
3) Step C: Client sends Authorization Grant to the Authorization server to apply for Access Token.
4) Step D: Authorization server attests the authenticity of Authorization Grant sent from Client and if it is authentic, Authorization server will send Access Token to Client
5) Step E: Client requests the protected resource by sending Access Token
6) Step F: Resource server verifies the authenticity of Access Token and if it is true, then sends back to the protected resource

```
+--------+                               +---------------+
|        |--(A)- Authorization Request ->|   Resource    |
|        |                               |     Owner     |
|        |<-(B)-- Authorization Grant ---|               |
|        |                               +---------------+
|        |
|        |                               +---------------+
|        |--(C)-- Authorization Grant -->| Authorization |
| Client |                               |     Server    |
|        |<-(D)----- Access Token -------|               |
|        |                               +---------------+
|        |
|        |                               +---------------+
|        |--(E)----- Access Token ------>|    Resource   |
|        |                               |     Server    |
|        |<-(F)--- Protected Resource ---|               |
+--------+                               +---------------+
```

Figure 17: Running process of OAuth2.0

Among the steps mentioned above, Step B is pivotal. In other words, how the user grants authentication to Client has become the critical point. OAuth2.0 defines four methods of authorization grant: *authorization code grant*, *implicit grant*, *resource owner password credentials grant* and *client credentials grant*.

Authorization code grant owns the most integrated functions as well as the most rigorous authorization process [42]. The most prominent feature of this kind of grant lies in introducing User Agent to communicate with the Authorization server. The procedures of the authorization code grant have been shown in Figure 18. There are five steps in this figure:

1) Step A: Client redirects the user to the Authorization server.
2) Step B: Resource Owner chooses whether to grant authentication to Client.
3) Step C: if Resource Owner grants authorization, Authorization server will redirect the user to the redirection URI and attach authorization code.
4) Step D: Client receives an authorization code and attaches receiving redirection URI, then sends back to the Authorization server.
5) Step E: Authorization verifies the authenticity of the authorization code and redirection URI then sends Access Token and Refresh Token back to Client.

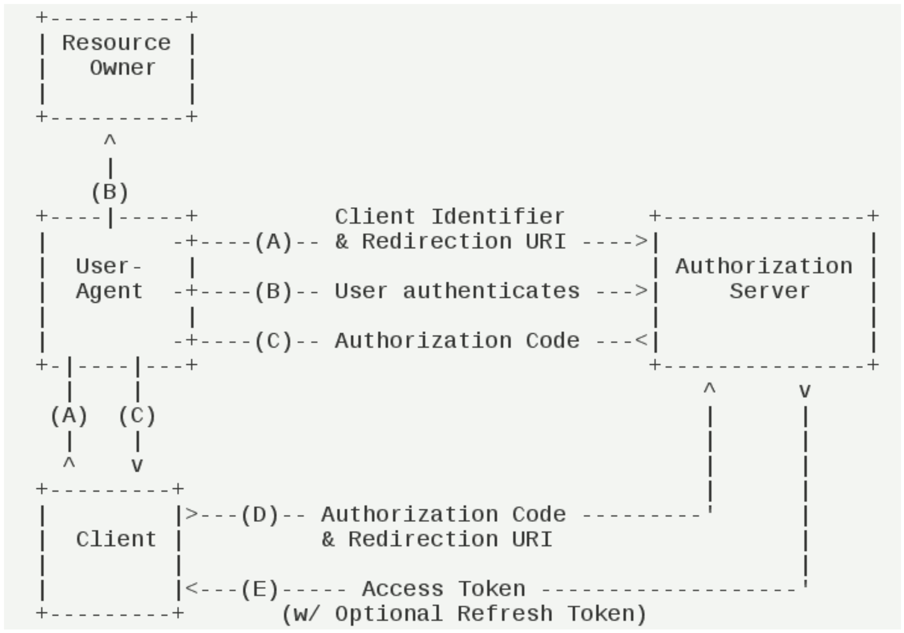
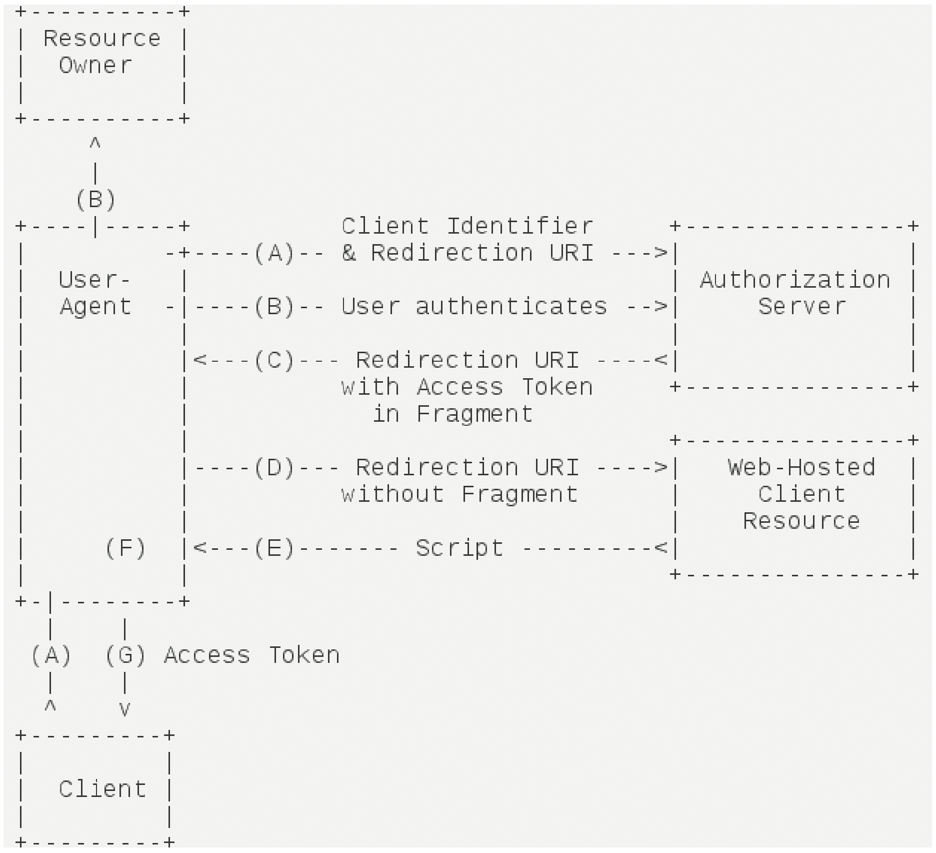


Figure 18: Procedures of authorization code grant

Compared with authorization code grant, implicit grant skips the process of acquiring authorization code and all the processes implement in Browser [43]. The detailed processes can be demonstrated in Figure 19.

1) Step A: Client redirects the user to the Authorization server.
2) Step B: Resource Owner chooses whether to grant authentication to Client.
3) Step C: if Resource Owner grants authorization, Authorization server will redirect the user to the redirection URI and attach Access Token in Hash Fragment.
4) Step D: Browser sends a request to Web-Hosted Client Source without Hash Fragment.

5) Step E: Web-Hosted Client Resource responds with a web page in which Client could get Access Token encrypted in Hash Fragment.
6) Step F: Browser executes the script in the above web page and extracts the Access Token.
7) Step G: Browser sends the Access Token back to Client.



```
+----------+
| Resource |
|  Owner   |
|          |
+----------+
     ^
     |
    (B)
+----|-----+          Client Identifier     +---------------+
|         -+----(A)-- & Redirection URI --->|               |
|  User-   |                                | Authorization |
|  Agent  -|----(B)-- User authenticates -->|     Server    |
|          |                                |               |
|          |<---(C)--- Redirection URI ----<|               |
|          |          with Access Token     +---------------+
|          |            in Fragment
|          |                                +---------------+
|          |----(D)--- Redirection URI ---->|   Web-Hosted  |
|          |          without Fragment      |     Client    |
|          |                                |    Resource   |
|     (F)  |<---(E)------- Script ---------<|               |
|          |                                +---------------+
+-|--------+
  |    |
 (A)  (G) Access Token
  |    |
  ^    v
+---------+
|         |
|  Client |
|         |
+---------+
```

Figure 19: Procedures of implicit grant

Resource owner password credential refers to the fact that the user provides his/her account name and password to the third-party Client directly. Then Client uses such information to acquire authentication grant from ISP. This kind of authorization grant has lowest security level [43] and it should not be considered unless there is a healthy trust relationship between users and Client.

The last authorization grant type is called client credentials grant. It is noteworthy that in this type, Client requests authorization grant in the name of its own instead of the user.

Another critical issue concerning OAuth2.0 is the mechanism of refreshing Access Token. In the original versions of OAuth, the validation of Access Token could be one year or even longer which brings numerous potential safety hazards [43]. In OAuth2.0, the validity of Access Token has been shortening sharply (generally within 24 hours) and the concept of Refresh Token was proposed. Usually, Refresh Token has an extended validity and it is optional. When the Access Token expires, Resource server will notify Client and at this time, Client will send its credential (client_id and client_secret) and Refresh Token to the Authorization server to request a new Access Token. With the help of this mechanism, OAuth2.0 restricts the validity of Access Token [44].

In conclusion, OAuth2.0 provides a safe and straightforward solution to the problem of validation and authorization of users in the Open-source platform including E-learning platform. Nowadays, there are increasingly growing number of companies supporting OAuth2.0 in their market applications such as Facebook, Twitter, App Store.

2.3 Blockchain

In 2008, a groundbreaking paper *Bitcoin: Peer-to-Peer Electronic Cash System* [5] written under the pseudonym Satoshi Nakamoto was published and attracted a myriad of researcher’s attention. In his later paper [45], a peer-to-peer electronic cash system was proposed and the term chain of blocks was introduced which is regarded as the prototype of Blockchain. According to Gartner’s 2016 Hype Cycle for Emerging Technologies graph [46] shown below, the technology of Blockchain is at the peak of inflated expectation (as of July 2016) and expected to keep mainstream momentum in the next 2 to 5 years. Interest in Blockchain has soared in the past years [47] and currently, it is being studied by the largest global companies and organizations with millions of dollars [47] invested for the sake of adopting and experimenting with this technology. In this section, the theoretical foundations of distributed system will be described first, then the Bitcoin protocol [5] which is considered as the precursor of Blockchain technology will be explained in detail, and then the related topics about Blockchain will be listed, finally. The related applications in this field will be enumerated. This is a logical way of fully understanding Blockchain technology. The knowledge explained in this section will be applied to solve questions raised in Chapter 1.

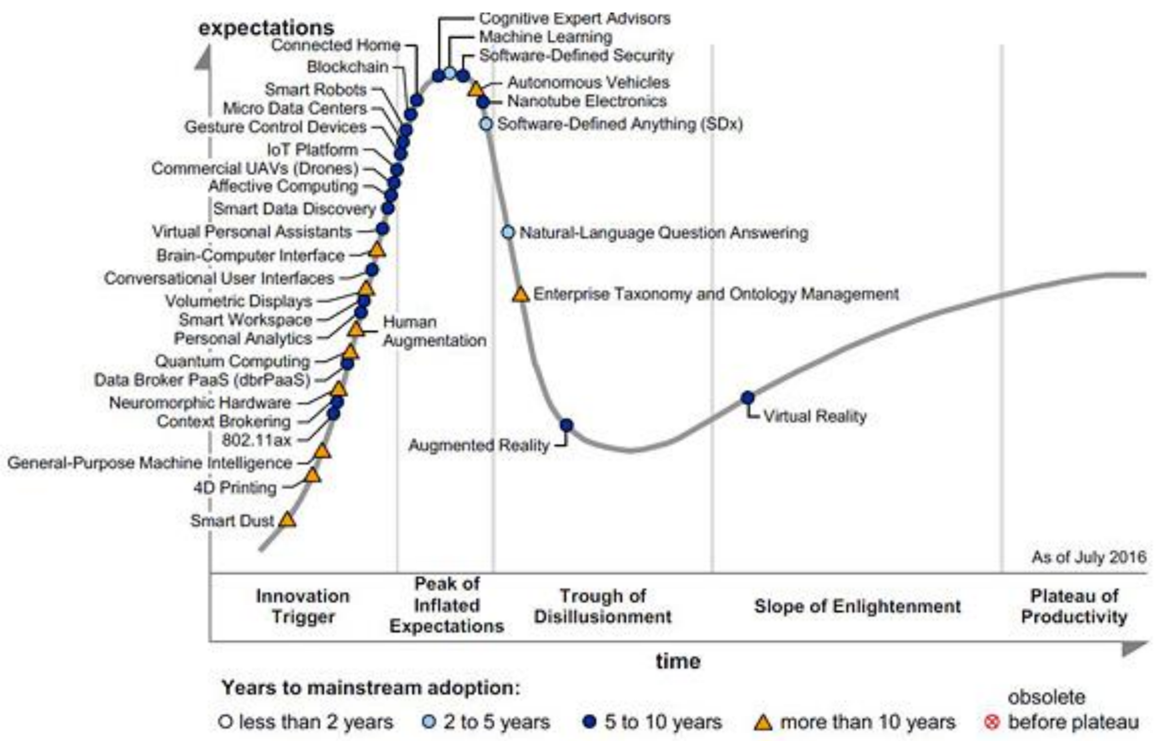


Figure 20: Gartner's hype cycle for emerging technologies [46]

2.3.1 Distributed System

Because Blockchain in its core is a decentralized distributed system, understanding distributed systems are requisite. According to Tanenbaum et al. [48], distributed systems are a computing paradigm whereby two or more nodes cooperate with each other in a coordinated model to achieve a common outcome. From the end user’s perspective, it is a holistic logical platform. A peer-to-peer network [49] is a representative distributed network.

A distributed system consists of numerous nodes. A node can be defined as an individual user in the system [48]. In the practical distributed system, nodes can be honest, faulty or malicious. A node that can execute arbitrary behavior is referred to Byzantine node [50]. Such kind of irrational behavior may cause a potential threat to the whole system.

While designing a distributed system, the primary challenge lies in the coordination and fault tolerance [47] between nodes. Fault tolerance denotes the distributed system should tolerate the situation in which some of the nodes become faulty and should run flawlessly to achieve the desired result. This issue has been a field of active area or research and a lot of mechanisms and algorithms [51,52] have been proposed to solve the problem. However, flaws are existing in these proposed solutions more or less. According to CAP theorem [53] which is also known as Brewer's theorem and later proved by Seth G. et al. [54], any distributed system cannot have Consistency, Availability and Partition Tolerance simultaneously.

1) Consistency: a property which makes sure that all nodes in a distributed system maintain a single latest copy of data.
2) Availability: the system runs without any failure when there are plenty of requests and responds between different nodes.
3) Partition tolerance: ensures the system operates correctly even when a cluster of nodes fail.

In brief, it is impossible to implement these three features altogether in a distributed system. Therefore, current distributed systems may consider one of the features mentioned above in preference of the other two.

The critical process to overcome the fault tolerance and coordination challenge is to solve the problem of distributed consensus [47]. In a centralized system, such problem does not even exist because there is a central server in charge of supervising and controlling the communication in the system. However, in a distributed system, when multiple nodes have to reach agreement on something, it becomes tough. Distributed consensus originates from Byzantine Generals problem [55] raised by Lamport et al. in 1982. Such problem can be described as follows: many army generals who were leading the army of Byzantine in different part planned to attack or retreat from a city. The only way to communicate with them is resorting to messengers. Besides, they need to reach consensus at the same time. In this situation, betrayers may exist in these generals and they may make a misleading message. Therefore, it is necessary to find a reliable mechanism which can ensure the consensus even in the presence of betrayers. As an analogy, in a distributed system, generals can be seen as nodes, betrayers can be considered as Byzantine nodes explained before and messengers can be regarded as the communication channels between different nodes. In 1999, Castro et al. [56] presented the Practical Byzantine Fault Tolerance (PBFT) algorithm and later in 2009, the first implementation of Proof of Work(POW) algorithm was made with the advent of Bitcoin system. POW will be explained in detail in the following part.

2.3.2 Bitcoin: A peer-to-peer Electronic Cash System

Over the past decades, there are growing number of bottom-up, grassroots applications aimed at solving technical problems in a cooperative, distributed manner [57]. In most cases, the practical solutions will come out soon after the thought for a specific application appears. However, there is an exception: digital money or also called electronic cash (e-cash). In the early 1980s, e-cash protocols appeared based on a model proposed by Chaum [58]. However, after almost a quarter of a century, the fully distributed solution was proposed. Regarding the history of e-cash, it has experienced two periods. Early attempts to establish e-cash systems require a central authority [57], in this period, approaches like B-money [59], Karma [60], PROW [61] and BitGold [62] were raised to eliminate the central bank. Even though these methods make some contributions (such as Proof-of-Work, blind signature) and pave the way for later approaches, they all rely on a centralized trusted authority. The critical challenge in the approaches mentioned above is how to solve double spend and Byzantine General problems. In a distributed e-cash system, double spend refers to the scenario that someone could issue two transactions in parallel [57] which means the same currency could be transferred to different recipients. It can be avoided by issuing a unique serial number through the supervision of the central bank. However, it is trivial to implement in a distributed environment. As mentioned in the previous section, Byzantine nodes may exist in a distributed environment and they will launch a malicious attack. Thus, it will be very likely to cause the problem of double spend if there are Byzantine nodes in a distributed e-cash system. For the sake of solving this problem, Quorum systems [63] were raised. Quorum systems are similar to voting systems in an election in which the majority of people can represent the whole community. However, such systems might be vulnerable to the Sybil attack [64]. The double spend and Byzantine General problem were not solved elegantly until the Bitcoin system [5] appeared in 2008. The Bitcoin system learned from the contributions of previous research [59-62] and employed the pragmatic POW algorithm to solve the problems. To fully understand the mechanisms used in the Bitcoin system, some topics related to Blockchain should be introduced first. So in Section 3.3.3, core technologies and concepts about Blockchain will be explained first and in Section 3.3.4, the Bitcoin protocol would be reviewed in detail.

2.3.3 Core Technologies and Concepts of Blockchain

Blockchain was created with the advent of the Bitcoin system [5] in 2008 as a portion of the Bitcoin protocol and it was implemented entirely in later 2009. According to Imran [47], Blockchain can be defined as a peer-to-peer distributed ledger that is cryptographically secure, append-only, immutable and updatable only via consensus or agreement among peers. As shown in Figure 21, Blockchain is built on a peer-to-peer network. There are five components constituting Blockchain: transactions, blocks, mining, consensus and smart contract.

Transaction is the basic unit in Blockchain which means a transfer value (such as Bitcoin) from the sender address to the recipient address. Here, the address is always derived from the public key by hashing it with SHA-256 first and RIPEMD-160 subsequently, prepending a version number and a checksum for error detection [57]. It is recommended to use a new key and address in each transaction in case of a comparison-based attack on signatures [65] and tracking of coins flow [66]. Transactions are depicted by a simple stack-based language called *script*. In this research, only the macroscopic view of Blockchain will be involved because there is no need to develop Blockchain but resort to some marketable Blockchain platforms (such as MultiChian or BigchainDB). So script language will not be explained in detail. The most ingenious part in designing transaction of Blockchain is to overcome the double spend problem elegantly. The critical components of a transaction are a hash value as the transaction identifier and a list of inputs and outputs [57]. One input should reference different outputs as shown in Figure 22. In other words, if the prevOut on the top is the same as the prevOut on the bottom, the double spend problem will occur and should be forbidden.

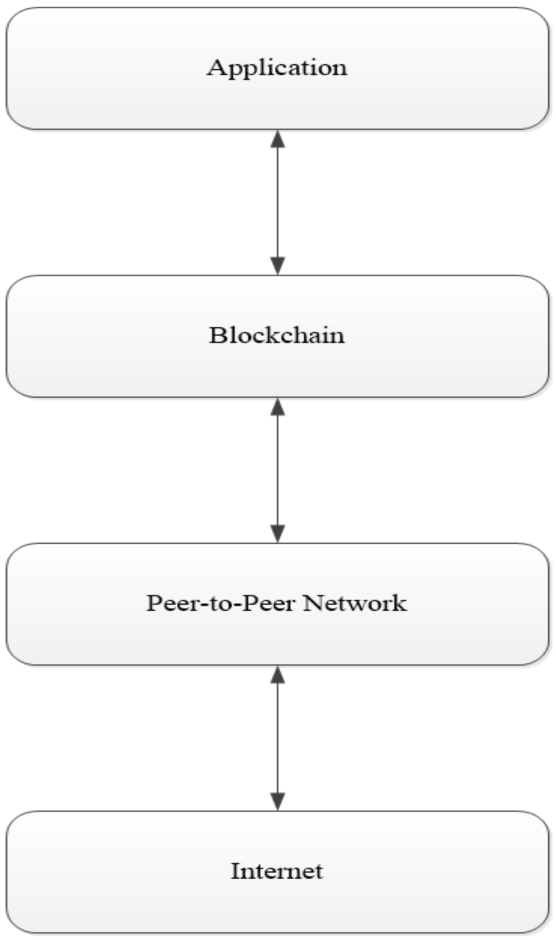


Figure 21: Blockchain network view

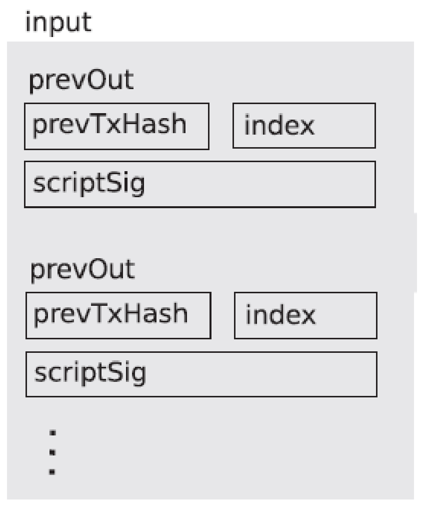


Figure 22: Input of a transaction

A block is a basic composition of Blockchain. From a micro perspective, Blockchain is a chain of blocks where each block is linked to its previous block by reference to the previous block header's hash [47]. There are three parts in one block: block header, transaction counter and transactions. Transaction counter refers to the total number of transactions in this block while transactions mean all transactions. Block header (as shown in Figure 23) is a complex structure including version, previous block hash, Merkle root, timestamp, difficulty target and nonce. Version dictates the block validation rules [47]. Previous block hash is a double SHA256 hash of previous block's header. Merkle root is a double SHA256 hash of the Merkle tree of all transactions. Timestamp contains the approximate creation time of the block in the Unix epoch time format. Difficulty target and nonce are related to mining in Blockchain which will be introduced later. Besides the components of the block, there is one special block called the *Genesis Block* which was created by Satoshi in January 2009. This block is the first block in the Blockchain and has no previous block hash. It is worth noting that such kind of structure in a block make the verification of transaction simple because Merkle hash root could reflect the complete transactions in this block and the tampering in any transaction would result in a different value of the root then would be detected [57]. On average, new blocks are added to the Blockchain every 10 minutes by the process of mining which will be explained in detail subsequently.

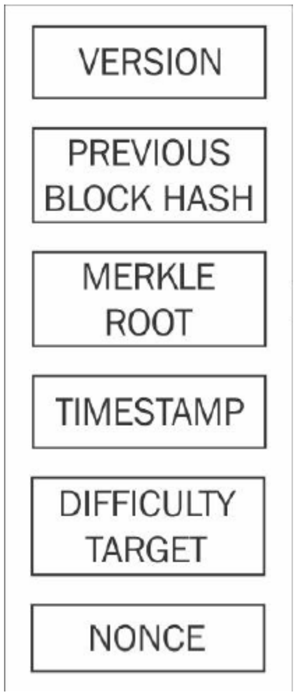


Figure 23: Components of block headers

Mining is one of the most complex mechanisms in the Blockchain technology including transaction validation, block validation, creating a new block, performing POW and fetching reward. Mining is a resource-intensive process and according to [67], in the process of mining, miners are encouraged to solve a mathematical puzzle which requires the consumption of computing power, once solved, the new block of transactions is accepted and committed to the Blockchain, the miner would be rewarded with newly generated coins accordingly. Solving the puzzle is always computationally difficult which requires multiple SHA-256 hash calculations [68] to guess the right number. In each block, there are two components relating to the puzzle: difficulty target and nonce. To be specific, the nonce can be regarded as the final hash number which is randomly chosen according to the current difficulty target. Difficulty target is used to express how difficult to find the nonce and this value is recalculated every 2,016 blocks [5]. Thus, roughly every two weeks (10mins*2016) the target value would be adjusted by the following formula:

$$T = T_{prev} * \frac{t_{actual}}{2016*10\,\text{min}}$$

Here, T_{prev} means the previous difficulty target value while t_{actual} refers to the actual time spanning from the last adjustment time of difficulty target to the current time. Every node in the Blockchain network competes to solve the puzzle and only the first node who calculates the nonce can win the competition. According to [57], this process of mining can be an analogy to “raffle tickets”: the more tickets you buy, the higher chance you will win. However, in the mining process, the tickets would be replaced by computing power which means the more computing power the miner gets, the higher chance it will solve the puzzle. Once the winner finds the nonce, it will send a broadcast to the whole Blockchain nodes with this nonce and let other nodes verify the solution; then the new block would be added to the Blockchain by the winner. There are two resources of rewards through the process of mining: *transaction fee* and *mining* [57]. The transaction fee is the extra service fee for each transaction while mining reward is the prize for the winners who have consumed computing power to calculate the nonce. In the Bitcoin protocol [5], the initial rewards were set to 50 Bitcoin (BTC) and approximately every four years; the reward would be halved. Even though there will be no more BTC minting for mining rewards roughly in 2140, miners can still profit from transaction fees.

The consensus is mostly a distributed concept reflecting the coordination of nodes in a peer-to-peer network [68]. In Blockchain, it provides a means of agreeing to a single version of truth by all peers in the Blockchain network [47]. In recent years, the algorithm of consensus in a distributed system has become active area or research involving proof-based algorithms (Proof of Work [5], Proof of Stake [69], Proof of Elapsed Time [47] and etc.) and Byzantine fault tolerance-based algorithms which employ the voting system to solve consensus problems. However, among these algorithms, only Proof-of-Work (POW) has proven to be adequately resistant to Sybil attacks [45,47]. In the field of Blockchain, POW algorithm is used in the block validation within the process of mining. Sometimes, *forks* may occur when two nodes simultaneously announce a valid block [47] or the Bitcoin protocol has updated. In the first case, the forks generate naturally and it is an undesirable situation in which the Bitcoin protocol employs POW to solve the forks problem. According to [5], mining is continued on the "longest" fork which involves the highest volume of computing power. In other words, only one chain (the most extended chain) can be chosen as a result of POW consensus algorithm which is also called *main chain*. The other divergent chains are called *orphaned chains* which will be discarded. Nevertheless, there is a potential threat existing in this situation: if the adversary (or Byzantine node) collects over 50% computing power, then they can change the blocks of transactions on their own will. This is also called 51% attacks and [71] has proposed an innovative way to prevent it. In the second case, the forks generate with the version updating and it can be solved by upgrading the version of Blockchain.

Before introducing the smart contracts, there is a need to explain the tires of Blockchain. According to [72], Blockchain can be categorized into three tires:

1) Blockchain1.0: it was only used for cryptocurrencies and can be regarded as the initial version of Blockchain.
2) Blockchain2.0: it can be used in financial services and smart contracts were introduced.
3) Blockchain3.0: it can be used beyond financial services including government, health, media, art and education.

It is clear that smart contracts were added in Blockchain2.0. Smart contracts is not a new concept but with the appearance of Blockchain, it has become a hotspot of research. The terminology of smart contracts was first theorized by Nick [73] in the early 1990s. But not until the Bitcoin protocol was implemented in 2009, did researchers realize the real potential and benefits of them.

The most attracting part in smart contracts lies in the scenario that nodes do not necessarily trust each other and they do not need a trusted intermediary when transferring values between each other in a trustless peer-to-peer network. Until now, there is no admitted definition of smart contracts. According to [47], smart contracts can be defined as a secure and unstoppable computer program representing an agreement that is automatically executable and enforceable. From this point of view, smart contracts are computer programs and the code interprets them. Like contracts, they encompass the agreement between two parties. Smart contracts will be executed automatically when some conditions are triggered. They are enforceable which means that they should be executed by measurements of controls without any mediation. In Blockchain, real smart contracts should be only conducted with the help of code. In other words, the code is law [47]. Another critical issue in smart contracts is carefully designing the contracts to make them unstoppable and secure. The smart contracts should be fault tolerated and they should be executed in reasonable amounts of time [47]. In this research, smart contracts will not be overextended. [74-75] have explained the future use of smart contracts in Blockchain and both of them show full of expectation to the promising future in this field.

2.3.4 Bitcoin Protocol

Until now, most important parts of the Blockchain technology have been involved. However, there is one problem to be solved: How is the Bitcoin protocol working as a whole? Fractions of Blockchain have been explained above in detail and subsequently, they should be assembled to get a full understand of the Blockchain technology. Assuming User A needs to do a Bitcoin transfer with User B, there are six processes according to the Bitcoin protocol [5].

1) Transaction: User A initiates the transaction by using the Bitcoin wallet.
2) Broadcast: User A announces the transaction in Bitcoin peer-to-peer network.
3) Blockchain: the new block waits to be added to the distributed ledger Blockchain.
4) Proof of Work: miners compete for calculating the nonce by running multiple Hash functions by the current difficulty.
5) Transaction Verification: miners verify the transaction and bundle it into a new block.
6) Confirmations: User B waits for confirmations and then confirm the transaction.

Figure 24 explains the above processes. It is worth noting that the initial step of starting a transaction through Bitcoin protocol is to install a Bitcoin wallet application. The wallet application always consists of a pair of public/private key to identify the user. Currently, there are numerous Bitcoin wallet applications sharing the market such as Blockchain, Bitcoin Wallet, etc. Another notable issue is in the process of Confirmation; double spend problem will become rarer with the number of confirmations increasing. Thus, there is a delay between the success of the transaction and the time when the new block is added to the Blockchain.

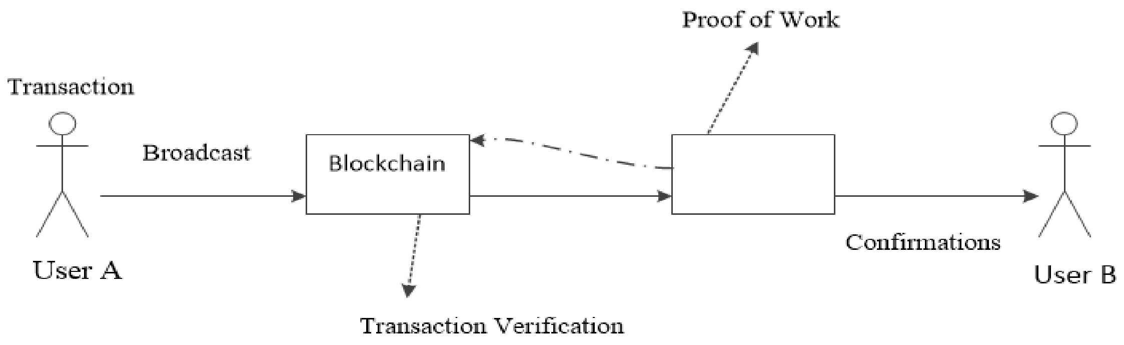


Figure 24: Procedures of bitcoin protocol

2.3.5 Private and Public Blockchain

Concerning the classifications of Blockchain, there are multiple standards [47]. One of the most widely used methods is to sort Blockchain into three categories: public Blockchain, private Blockchain and semi-private Blockchain.

As the name suggests, public Blockchain is the ledgers publicly open for anyone who wants to participate. It is also called permission-less ledgers which are maintained by all users. According to [47], a copy of the ledger will be kept on local nodes and distributed consensus mechanism described in Section 3.3.3 will be applied to reach a joint decision about the final state of the Blockchain. There are numerous applications based on public Blockchains such as Bitcoin system [5], Ethereum [76], etc. Bitcoin system has been explained in the previous sections and Ethereum will be described in the next section.

Compared with public Blockchain, the private one is only open to a crowd of individuals or specific companies. In most cases, they refuse to share the ledger publicly so they just maintain the ledger among themselves. According to [77], private Blockchain is still in its nascent stage and the number of companies working in this area would expect to be multi-fold in 2018. Currently, Multichain (http://www.Multichain.com) is a popular and straightforward platform for creating and deploying private Blockchains.

The concept of semi-private Blockchain is disputable [77] which is accepted by the vast majority of people. It is also named hybrid Blockchain combing part of the public Blockchain with part of the private Blockchain. The public part is controlled by a group of individuals while the public part is open to the public.

2.3.6 Applications of Blockchain

Considering in this research, Multichain will be employed as a platform to improve the information security of E-learning system. Thus, only two representative Blockchain platforms would be explained in this section: Ethereum [76] and Multichain.

In November 2013, Ethereum [76] was conceptualized by Buterin. The primary contribution is implementing smart contracts (mentioned in Section 3.3.3) for Blockchain and decentralized applications by developing a Turing-complete language [78]. Furthermore, this is the most significant progress of Bitcoin system to Ethereum: In Bitcoin system explained in the previous section, the scripting language is insufficient just allowing essential and requisite operations. According to the yellow paper [79], it defines Ethereum as follows:

Ethereum is a project which attempts to build the generalized technology; technology on which all transaction-based state machine concepts may be built. Moreover, it aims to provide to the end-developer a tightly integrated end-to-end system for building software on a hitherto unexplored compute paradigm in the mainstream: a trustful object messaging compute framework.

The concept is complicated enough. However, for developers, in simple terms, it is a platform that enables them to develop decentralized applications running on the Blockchain [78]. Based on this concept, various applications have been developed by using different programming languages. Go-etheruem (https://ethereum.github.io/go-ethereum) was developed by Golang and it is an entirely open-source and licensed Etheruem platform under the GNU LPGL v3. Parity (https://parity.io) is another Etheruem platform which was developed by Rust. The first release of Ethereum was called *Frontier* and the current version was known as *homestead release*. Next generation of Etheruem called *serenity* is aimed to simplify and improve the performance of the protocol. Proof-of-Stake algorithm [80] would be employed and it will also realize Web3.0 concept in which semantic and intelligent web technology will be implemented. According to [78], Bitcoin system and Etheruem platform are probably two representers of the Blockchain technology. From the evolution of these two emerging technologies, it is not hard to infer that there is a new trend from building only economic systems on top of the Blockchain technology to developing various applications in different fields.

Multichain is a powerful platform for developers to create and deploy private Blockchain. According to [81], the goal of Multichain is to overcome the problems of mining, privacy and openness in the deployment of Blockchain technology by integrating management of user permissions and providing required controls in an easy-to-use package. The core contributions for Multichain are in three aspects [81]: Firstly, it guarantees that only chosen participants could see Blockchain activities; Secondly, required controls are introduced to restrict the related transactions; Last but not least, mining can continuously proceed without the control of POW algorithm. In this research, Multichain will be considered as a tool to create and deploy my private Blockchain to keep the authenticity, integrity and non-repudiation of the E-learning resource. So there is no need to show more details about how it works. Further topics can be found in [81].

2.3.7 Summary

The Blockchain technology brings an innovative mechanism to solve the distributed consensus problem in a decentralized system. There are numerous benefits of such technology:

1) Decentralization: there is no need to introduce a trusted third-party intermediary. Instead, a distributed consensus mechanism will be applied.
2) Transparency: everyone can see what happens in the Blockchain which allows the system to be more transparent and trustworthy.
3) Immutability: it becomes almost impossible (except for 51% attacks) to modify the transaction in the Blockchain.

4) Highly-secure: all transactions recorded on the Blockchain are encoded and provide integrity validation.
5) Efficiency: in the area of finance, Blockchain makes the process of transactions fast and straightforward.

Like every innovative technology, Blockchain also has several deficiencies. In reality, much effort is being made to overcome the weaknesses to build a robust system [82]. According to [82], a selection of the most sensitive challenges have been listed below:

1) Scalability: with the growth of the number of nodes, the Blockchain becomes bulky.
2) Vulnerability: Blockchain is susceptible to 51% attacks.
3) Deficiency of POW algorithm: the most significant controversy of POW algorithm lies in energy wasting for meaningless hashing to compete with others [5].
4) The tendency to centralization: there is a trend that miners prefer to group in a mining pool to ensure steady income which causes the centralization of Blockchain [83].
5) Privacy leaking: according to [84], Blockchain could not ensure the *transaction privacy* due to the values and balances for each user exposed publicly.

As mentioned before, Blockchain is still quite young and prone to emerging problems. Numerous solutions have been proposed such as storage optimization of Blockchain [85], advances on consensus protocols [80, 86-87], etc. Nowadays, considerable research has been involved in applying the Blockchain technology to various kinds of field. In this research, Multichain will be explored to apply to E-learning industry. A detailed description of the platform will be shown in Chapter 3.

2.4 RESTful Web Services

RESTful web services were initially introduced by Fielding et al. [88] in 2000. It has become a dominant framework for building various kinds of systems. In essence, REST(Representation State Transfer) is a simple, lightweight framework style and it can send data via HTTP request [89]. It has made a set of principles and constraints while designing a framework. Furthermore, RESTful represents the framework following the principles mentioned above. According to [88], there are four principles when designing RESTful Web Services:

1) URI (Unique Resource Identifier): each resource should be assigned to a unique ID.

2) Use HTTP methods: it has a one-to-one mapping with HTTP methods (GET, POST, PUT and DELETE).
3) Multiple representations of resource: it should provide different representations according to Client’s need.
4) Statelessness: the request from Client should be complete, stateless and independent in the context of the program.

In recent years, there are a rapidly growing number of companies investing in developing RESTful API (Application Interface) including Google, Twitter and Amazon. In E-learning platform, Moodle has provided its RESTful API for developers to build an easy-to-use and functional E-learning platform rapidly. In this research, the proposed platform will apply RESTful web services through using Beego (https://Beego.me) framework which will be explained in the next chapter.

Chapter 3
SYSTEM ARCHITECTURE

In this chapter, the detailed system architecture and the related methods will be described. In section 3.1, a system architecture overview will be explained; in Section 3.2, the components in the system will be represented; in Section 3.3, the workflow will be described in detail; in Section 3.4, implementation and related code snippets will be described and Section 3.5 is the summary and answering the research questions raised in Chapter 1.

3.1 System Architecture Overview

Based on Figure 3 and 4 (Chapter 1), the proposed E-learning platform is aiming at solving the problems raised in Section 2.2. There are two problems:

1) Problem 1: How to improve the performance of access control system on a peer-to-peer E-learning platform by the decentralized solution?
2) Problem 2: How to increase the horizontal scalability of decentralized access control systems on a peer-to-peer E-learning platform?

To solve the questions, Blockchain technology is introduced. As Section 3.3 stated, Blockchain brings the consensus mechanism through POW algorithm to solve Byzantine General problem in a peer-to-peer network. Meanwhile, through the private chain ledger, sensitive data could be tracked in a secure and immutable way. In other words, the authenticity, integrity, non-repudiation, as well as traceability of the data, could be ensured by checking the private ledger. Thus, Blockchain could be employed as a tool to provide secure decentralized access control. Besides, with the help of JSON format, all interactive data of decentralized access control could be tracked simply.

It is noteworthy that in my proposed E-learning platform, the roles have simplified into two categories since the system will become so complicated if more roles are selected. Besides, to make the platform portable, a Mobile Application on Android platform is chosen.

Figure 25 shows the proposed architectural design of my E-learning platform. To create decentralized access control system, several changes have been made. Firstly, there is no central server in this platform: each E-learner or teacher could manage his/her

own E-learning resources independently which means nodes in E-learning platforms could be managed autonomously. Secondly, because storing or retrieving bulky E-learning resources in a peer-to-peer E-learning platform may extend the latency between two nodes, only URL (Uniform Resource Locator) is stored in Blockchain and the bulky E-learning resources are stored in local Database (MySQL Database). Finally, Blockchain could provide secure and traceable protection in building the decentralized access control model. There are four main security features of protection by using Blockchain:

1) Authenticity: the original initiator of the lesson could be traced through Blockchain.
2) Integrity: the lesson file cannot be modified through the process of transferring since Blockchain could check the integrity of the lesson file.
3) Non-repudiation: every transaction can be seen through Blockchain and if someone changes the file, it can be detected by checking the transactions in Blockchain.
4) Traceability: every transaction could be tracked through checking private Blockchain ledger.

Since an E-learning platform includes various functions as stated in Section 3.1, the workload would be too heavy if the proposed architecture implements all of these functions. In my proposed E-learning platform architecture, only the process of creation and consumption of the E-learning resources will be focused. As shown in Figure 26, the role of Teacher/Administrator is granted the permission of creating the E-learning resources through a Course Editing interface which is implemented by HTML/CSS/ANGULAR JS while the role of E-learner is granted the permission of consuming the specified E-learning resources through a Mobile Application written in Java language on Android platform. Another noticeable feature of my proposed architecture is that the front end does not interact with Local E-learning database directly. Instead, Golang framework E-learning Restful API serves as the middleware. The main reason to choose Golang rather than other programming languages lies in its high performance of concurrent and distributed computing. It is worth noting that the detailed evaluation of the performance and horizontal scalability will be shown in the next chapter.

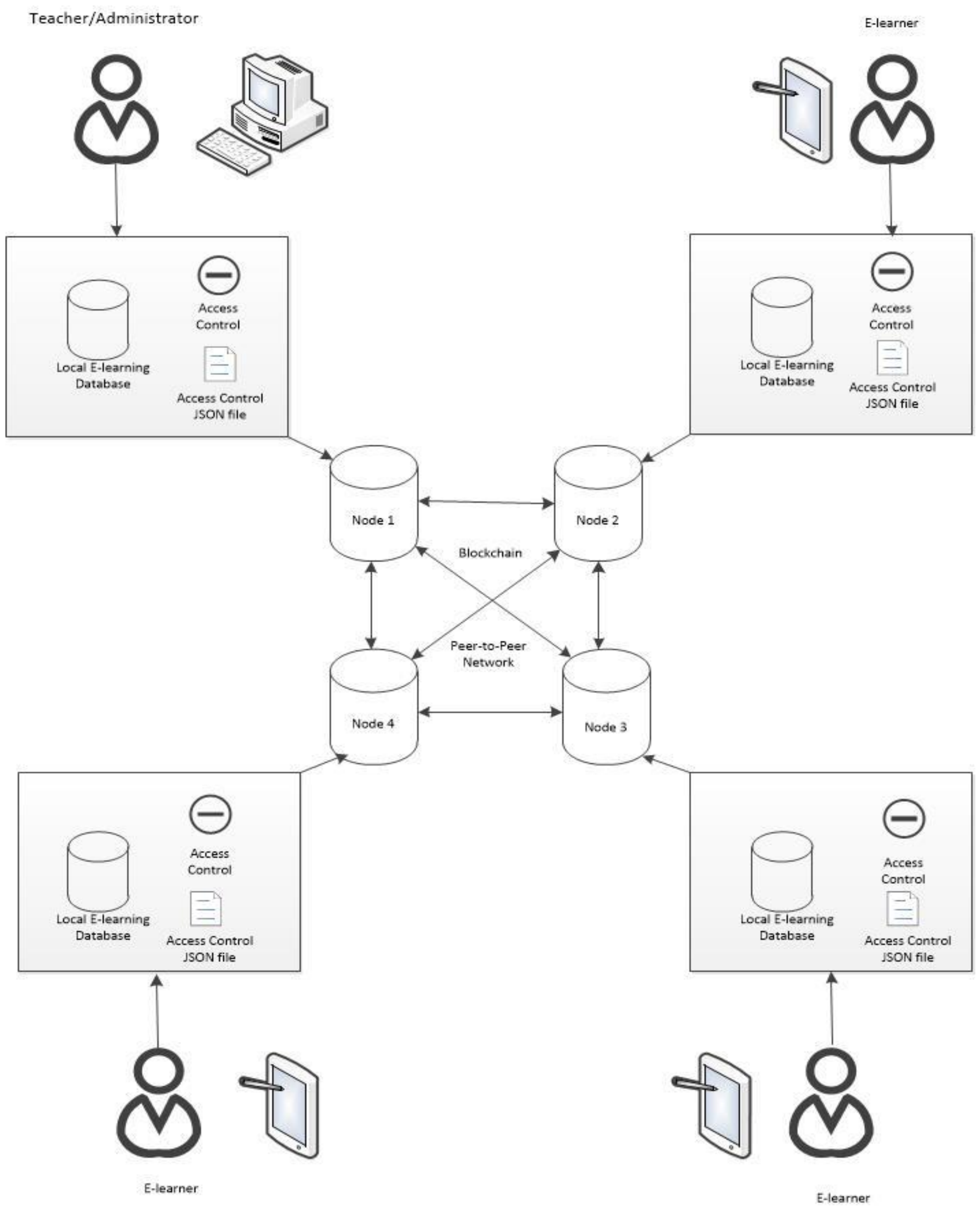


Figure 25: System architecture

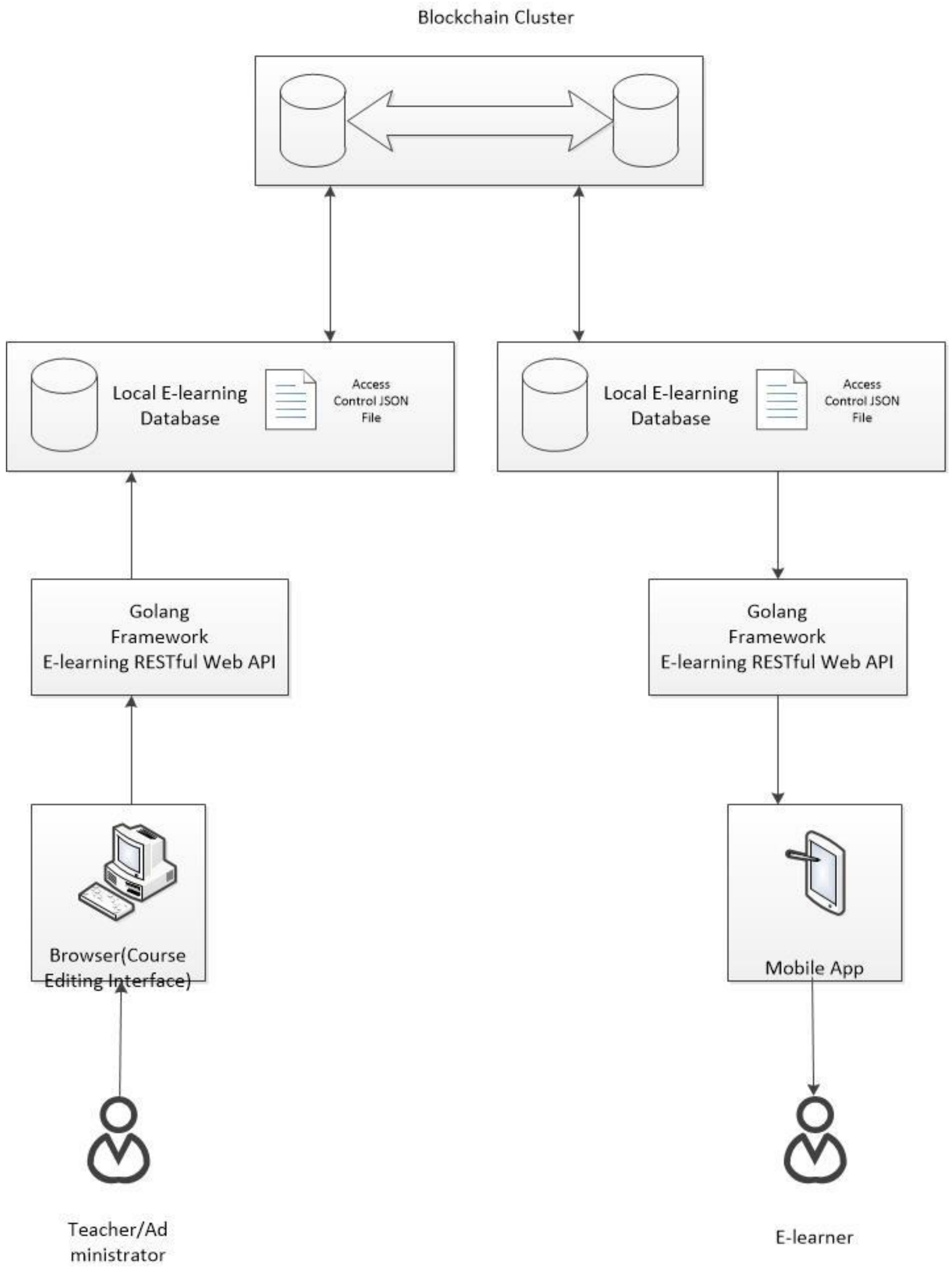

Blockchain Cluster
Local E-learning Database
Access Control JSON File
Local E-learning Database
Access Control JSON File
Golang Framework E-learning RESTful Web API
Golang Framework E-learning RESTful Web API
Browser(Course Editing Interface)
Mobile App
Teacher/Administrator
E-learner

Figure 26: Detailed system architecture

3.2 System Components

As shown in Figure 25 and 26, ten components are constituting the proposed E-learning platform:

i. **Teachers/Administrator**: The role of Teachers/Administrator is to edit lessons and after editing lessons, they can assign the lesson.

ii. **Students**: Students are required to watch lessons through a Mobile Application (Lessons Basket).

iii. **Course Editing Tool**: As the name suggests, it is used to edit the course and add multimedia such as videos, pictures and audios of the course. The GUI (Graphic User Interface) should be user-friendly and simple. In my proposed E-learning platform, it will be implemented by using AngularJS, CSS (Cascade Style Sheet) and HTML5.

iv. **E-learning RESTful Web API**: In my E-learning platform, it is in charge of communicating with Client sides(Course Editing Tool and Mobile Application). It is implemented on top of Beego API framework which is a powerful and easy-to-use Go framework to help to build API by using Golang (https://golang.org).

v. **Mobile Application**: It is a Mobile Application providing a platform for students to see the lesson list and watch the lesson video. It will be implemented on Android platform by using Java.

vi. **Local Learning Resource Database**: It is a database used to store E-learning resources including student's profile, student's scores and course information. It will be implemented in MySQL database.

vii. **Access Control JSON File**: It is implemented by JSON (JavaScript Object Notation). the reason why I choose JSON format as access control file lies in its lightweight and simplicity. The main role of such file is to check whether the user has been granted the respective permission to operate the E-learning resource.

viii. **Blockchain Cluster**: In my proposed architecture, I plan to use the Multichain Blockchain technology, a private peer-to-peer distributed ledger and database that is cryptographically secure, append-only, immutable and updatable only via consensus or agreement among peers.

ix. **Network** – Communication is crucial to every E-learning platform in the process of delivering content. In my proposed architecture, HTTP protocol will be used for the communication among the components.

x. **Cloud Web Service** – In order to simulate the actual network in reality, Cloud Web Service has been chosen. To evaluate the performance of my platform, both LAN and Cloud Web Service are used to measure the difference between the optimal condition and the realistic condition. Amazon Web Service is one of the most commonly used Cloud Web Service and in my proposed platform, it is selected.

3.3 Work Flow and Related Methods of Proposed E-learning Platform

Figure 26 shows the rough procedures of creating and consuming E-learning resources in my proposed E-learning platform. There are four main processes being explained as follows.

Creating the E-learning resource

The first step in my proposed E-learning platform is to create the E-learning resource. As Figure 26 shown, the bulky E-learning resource should be stored in local E-learning database which is implemented by MySQL database. Figure 27 shows the complete E-R model of my proposed E-learning platform including future work. For creating the E-learning resource, the table of lessons should be filled. The columns in this table are explained as follows:

1) Id: the primary key of the lesson.
2) Description: the description of the course to explain what the course is.
3) Title: the title of the course.
4) Grade: the grade of this course belongs to such as Grade 3, Grade 4 and etc.
5) Viewcount: the column used to record how many E-learners have consumed this resource.
6) Ratingcount: the column used to record the rating of this resource.
7) Admin_id: reflects the teacher identifier which is a foreign key reference to Admin table.
8) Created_date: the column used to record the created date of the resource.
9) Video_url: represents the video stored in the resource.
10) Image_url: represents the image stored in the resource.
11) Time: set the total time of finishing the quiz after consuming the E-learning resource
12) Assigned: represents whether the resource is published or just stored in local database.
13) Notes: represents the extra notes of the resource.

After filling in the Lesson table through GUI, the E-learning resource will be created and stored in the teacher’s local database. However, at this moment, the students could not see the E-learning resource until the teacher presses the assign button. As explained in the last section, Golang Web framework serves as Middleware. The latency should be low in this process since all the work runs locally through HTTP protocol.

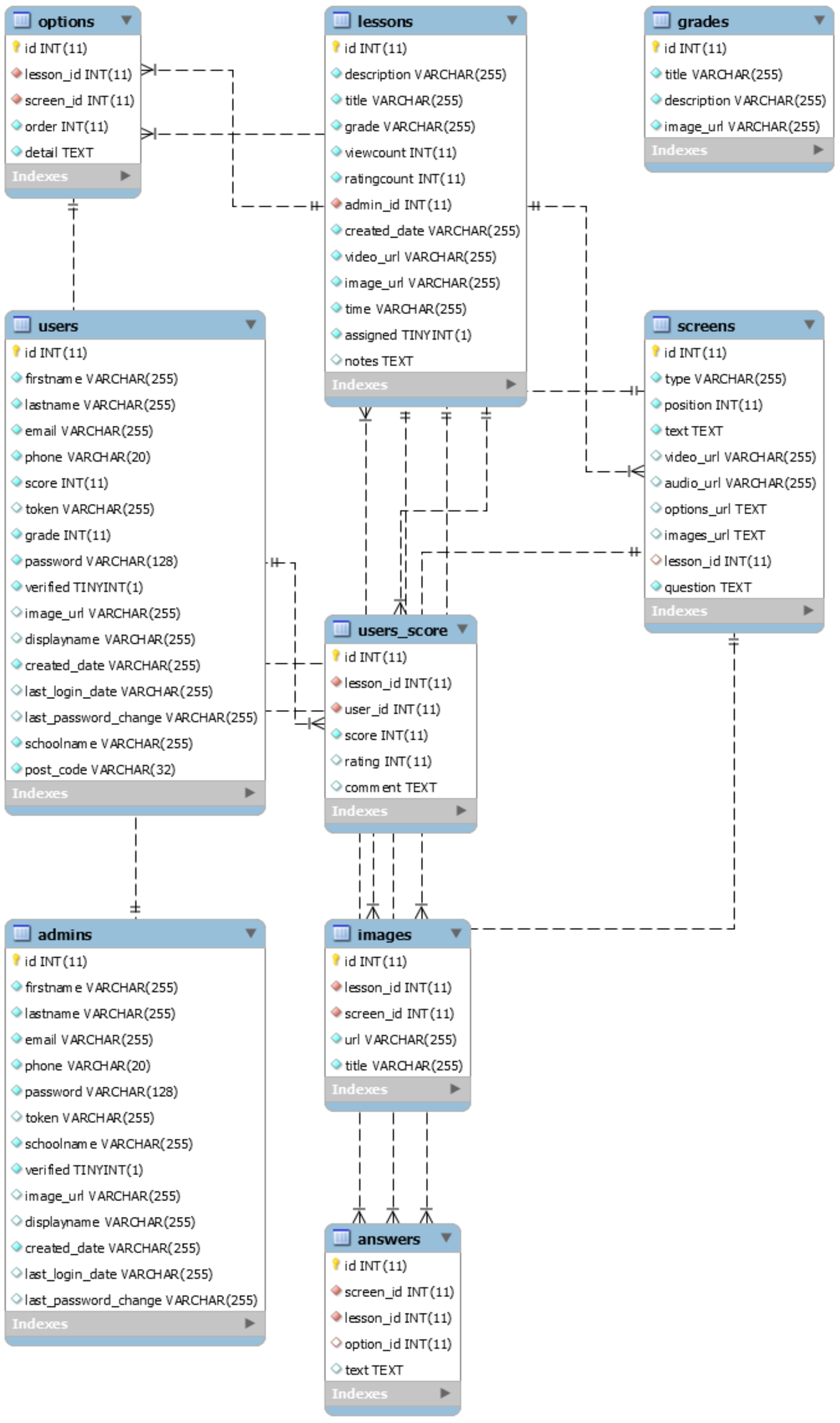


Figure 27: E-R model of E-learning platform

Granting access rights and assigning the E-learning resource

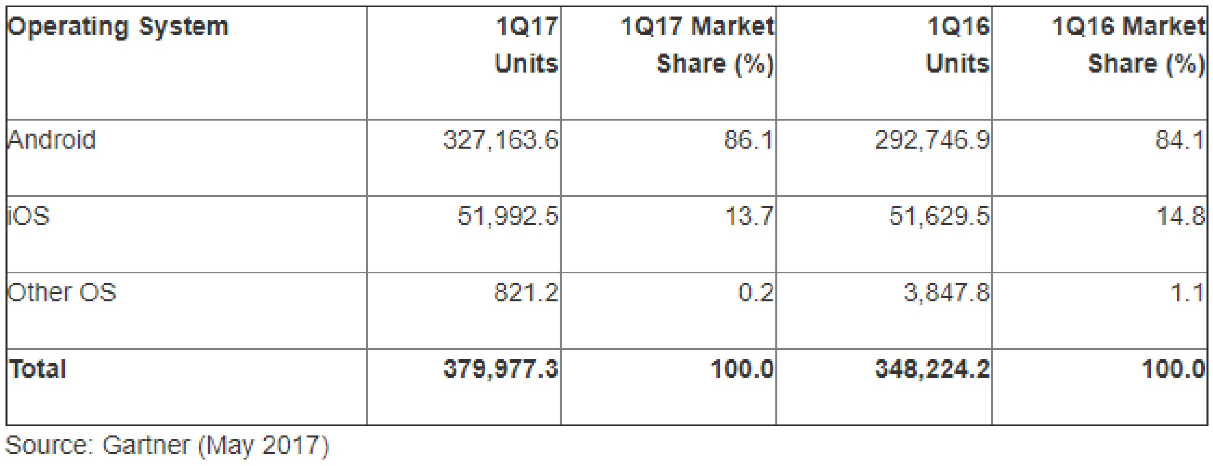
The access rights will be written in the format of JSON. The essence of the JSON file is the key/value pair which makes it a simple and easy-to-use format. It is also implemented through GUI. This file rules the respective permissions of different E-learners. Before assigning the E-learning resource, such file will be stored locally. After assigning the E-learning resource, the access control file along with the resource information will be transferred and propagated in a peer-to-peer network with a distributed consensus mechanism in Blockchain. Considering the visibility and permission of the E-learning resource is limited to a specific group, Multichain, a private Blockchain, is applied. As explained in Section 3.3, POW algorithm could maintain a consistent and traceable access control file within participant nodes in the peer-to-peer E-learning platform. The security of such architecture would improve to a large extent by using Blockchain. Meanwhile, the cost of the whole platform would lessen by eliminating the high-performance central server.

Updating the private ledger through Multichain Blockchain peer-to-peer network

As explained in Section 3.3, the ledger is maintained in a distributed consensus mechanism through POW algorithm and it is immutable and cryptographic. So any malicious attempt to modify the ledger will be refused unless launching a 51% attack. In terms of E-learning resources, the security and privacy should be considered as priorities. For this reason, Blockchain is an optimal tool to provide such kind of protection. When the E-learning resource is assigned, the ledger should be updated and new block may be created. With the purpose of decreasing the capacity of the ledger, only URL is stored instead of the bulky E-learning resource.

Consuming the E-learning resource through a Mobile Application

As Figure 26 shown, the E-learner could consume the E-learning resource created by the teacher. Before accessing the E-learning resource, the platform will check whether this E-learner has the respective permission to read this resource through the JSON file. If there is no reading permission, the mobile application should refuse to load the resource by popping up a friendly hint. It is also worth noting that the Golang framework also serves as the middleware. Additionally, the mobile application is visualized by using Java language in Android platform. The reason why I choose Android platform lies in its globally predominant market share in Smartphone Sales according to Figure 28 [90].



Operating System	1Q17 Units	1Q17 Market Share (%)	1Q16 Units	1Q16 Market Share (%)
Android	327,163.6	86.1	292,746.9	84.1
iOS	51,992.5	13.7	51,629.5	14.8
Other OS	821.2	0.2	3,847.8	1.1
Total	**379,977.3**	**100.0**	**348,224.2**	**100.0**

Source: Gartner (May 2017)

Figure 28: Worldwide smartphone sales to end users by operating system in 1Q17 (Thousands of Units) [90]

The workflow of my proposed E-learning platform has been described in detail. Blockchain makes the platform distributed in a secure and innovative way. Compared with the traditional central server E-learning platform shown in Figure 3, the latency may extend a little when there are a few clients connecting with the platform but the security and scalability should increase a lot. Besides, the maintenance cost of building such platform should drop. In the following section, the detailed implementation of my proposed platform will be explained and related programming code as well as screenshot will be also included.

3.4 Implementation

As there are four main steps being explained in the last section, the implementation will also be divided into these four stages. Before explaining the detailed implementation of every step, it is necessary to introduce the techniques used in the implementation period of my proposed E-learning platform.

1) HTML/CSS/JavaScript/JQuery: front end programming language and style designing language.
2) AngularJS: asynchronous front-end framework providing an easy and fast model.
3) MySQL: SQL management database used to develop my local database.

4) Golang: a back-end programming language developed by Google providing high performance concurrent and distributed computing.
5) Beego framework: a Golang web framework available at https://Beego.me/.
6) Restful Web Service: a simple, lightweight framework style and it is able to send data via HTTP request [89].
7) Java : an ubiquitous programming language.
8) Android Studio: an Android development tool maintained by Google.
9) Multichain platform: a private Blockchain platform.
10) MyEclipse: a Java development tool.

Creating the E-learning resource

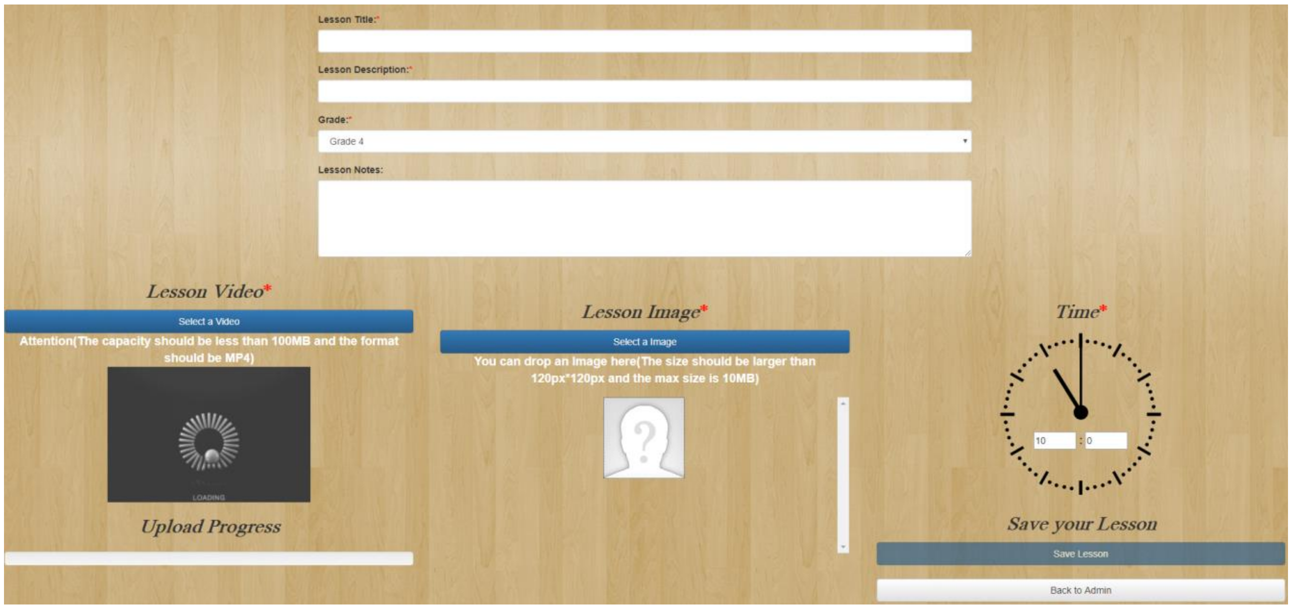
As stated in the last section, Golang Web API framework is used as a middleware. Since Beego is one of the most popular Web API frameworks, it is selected in my proposed platform. The core URL in this stage is shown as follows:

Beego.Router("/v1/lessonsadmin/:admin_id([0-9]+)",controllers.LessonsController{}, "post:AddLesson")

The function of this line of code is to receive a post request sending from the Course Editing Tool and store the posted E-learning resource information into local MySQL database. Meanwhile, all the bulky media files (such as videos and images) will also be stored locally. The core function of this step is called AddLesson and it is listed in Figure 29. The interface of editing the E-learning resource is captured in Figure 30. It is noteworthy that E-learning resources and lessons are interchangeable in this thesis.

```go
// Add Lesson
func (c *LessonsController) AddLesson() {
	adminId := c.Ctx.Input.Param(":admin_id")
	aid, _ := strconv.Atoi(adminId)
	var v models.Lessons
	if err := json.Unmarshal(c.Ctx.Input.RequestBody, &v); err == nil {
		v.AdminId = &models.Admins{
			Id: aid,
		}
		//create date
		v.CreatedDate = tools.GenerateTime()
		if _, err := models.AddLessons(&v); err == nil {
			c.Ctx.Output.SetStatus(201)
			c.Data["json"] = v
		} else {
			jsonresult := &JsonResult{
				Result: "Upload fail!",
			}
			c.Data["json"] = jsonresult
		}
	} else {
		jsonresult := &JsonResult{
			Result: "Upload fail!",
		}
		c.Data["json"] = jsonresult
	}
	c.ServeJSON()
}
```

Figure 29: Code snippet for adding lesson



Lesson Title:*

Lesson Description:*

Grade:* Grade 4

Lesson Notes:

*Lesson Video**

Select a Video

Attention(The capacity should be less than 100MB and the format should be MP4)

*Lesson Image**

Select a Image

You can drop an Image here(The size should be larger than 120px*120px and the max size is 10MB)

*Time**

10 : 0

Upload Progress

Save your Lesson

Save Lesson

Back to Admin

Figure 30: Screenshot of creating E-learning resource

Granting access rights and assigning the E-learning resource

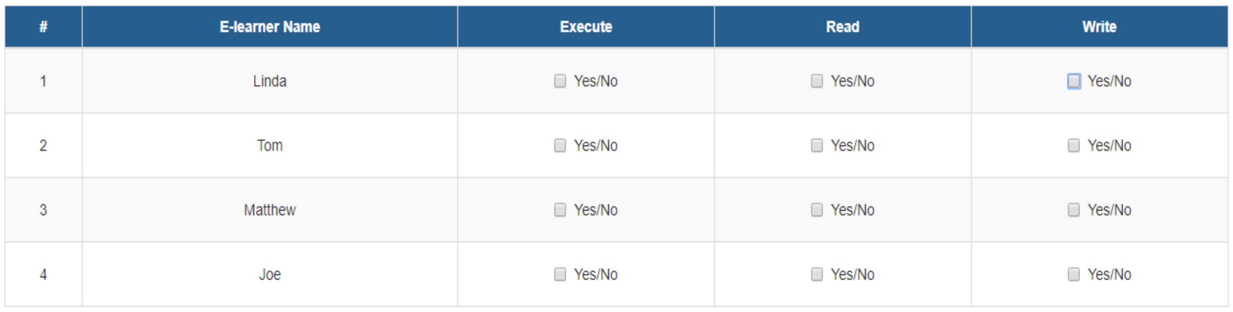
There are two main tasks in this procedure: one is generating the access control JSON file and the other is assigning the course from local Database to the private Multichain platform. Inspired by the role permission management in Operating System, there are three permissions being allocated to E-learners from high digit to low digit:

1) execute: E-learner could see this E-learning resource if it is true
2) read: E-learner could read and consume this E-learning resource if it is true
3) write: E-learner could update this E-learning resource if it is true

In order to make the JSON file simple, the binary system will be used here. The sample JSON file can be shown as follows:

```
{
        "Linda": 100,
        "Tom": 110,
        "Matthew": 110
}
```

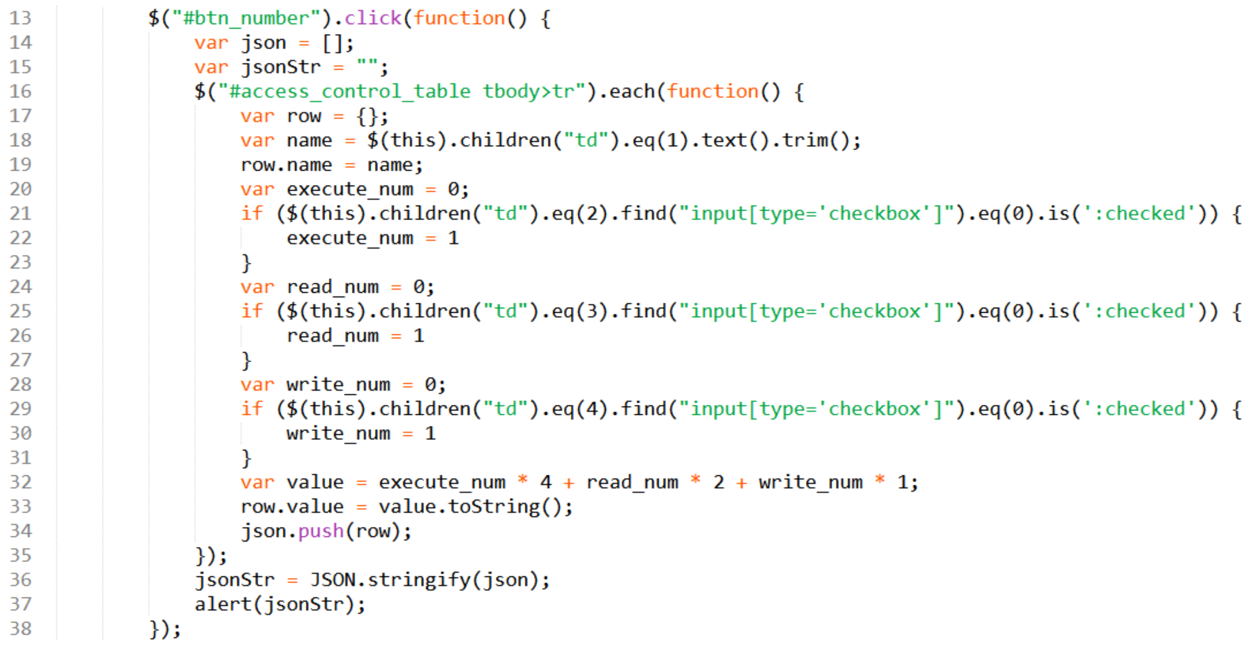
Here, Linda only has execute permission, Tom has execute and read permission while Matthew has execute, read and write permission. In my proposed E-learning platform, there are two steps generating such kind of JSON file. Firstly, it is edited through HTML page as shown in Figure 31.



#	E-learner Name	Execute	Read	Write
1	Linda	☐ Yes/No	☐ Yes/No	☐ Yes/No
2	Tom	☐ Yes/No	☐ Yes/No	☐ Yes/No
3	Matthew	☐ Yes/No	☐ Yes/No	☐ Yes/No
4	Joe	☐ Yes/No	☐ Yes/No	☐ Yes/No

Figure 31: GUI of editing access control system

The second step is to use JavaScript to transfer the table into JSON file. Figure 32 shows the code snippet of generating JSON file in JavaScript and JQuery. In order to calculate conveniently, all binary number is transferred to the respective decimal.



```
$("#btn_number").click(function() {
    var json = [];
    var jsonStr = "";
    $("#access_control_table tbody>tr").each(function() {
        var row = {};
        var name = $(this).children("td").eq(1).text().trim();
        row.name = name;
        var execute_num = 0;
        if ($(this).children("td").eq(2).find("input[type='checkbox']").eq(0).is(':checked')) {
            execute_num = 1
        }
        var read_num = 0;
        if ($(this).children("td").eq(3).find("input[type='checkbox']").eq(0).is(':checked')) {
            read_num = 1
        }
        var write_num = 0;
        if ($(this).children("td").eq(4).find("input[type='checkbox']").eq(0).is(':checked')) {
            write_num = 1
        }
        var value = execute_num * 4 + read_num * 2 + write_num * 1;
        row.value = value.toString();
        json.push(row);
    });
    jsonStr = JSON.stringify(json);
    alert(jsonStr);
});
```

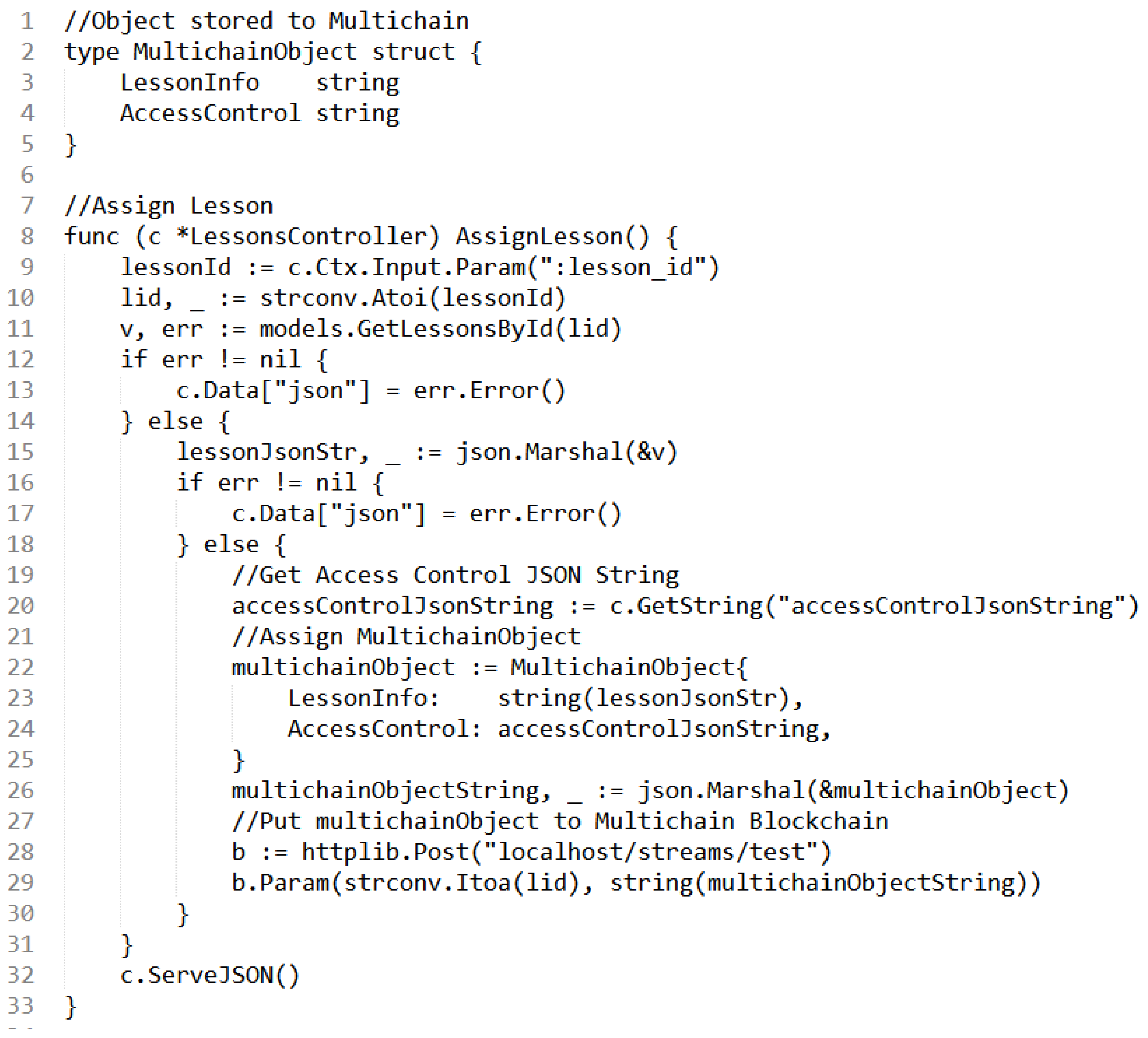
Figure 32: Code snippet of generating JSON string

It is noteworthy that this function also uses JQuery plugin which is a library of JavaScript. After running this function, the JSON string will be stored.

Another task in this process is to assign the E-learning resource. This is implemented by my Beego API:

Beego.Router("/v1/lessonsadmin/:lesson_id([0-9]+)",&controllers.LessonsController{}, "post:AssignLesson")

The core function AssignLesson has been shown in Figure 33. From the view of the teacher, there is no need to concern so much about the processes rather than pressing a button of assigning my lesson. It makes my proposed platform easy to use. By sending admin_id, lesson_id as well as access control JSON string, the Golang framework can query with local E-learning resources database to find the lesson information and then transfer it to the JSON file as well. At last, the framework puts these two JSON files into Multichain Blockchain platform.

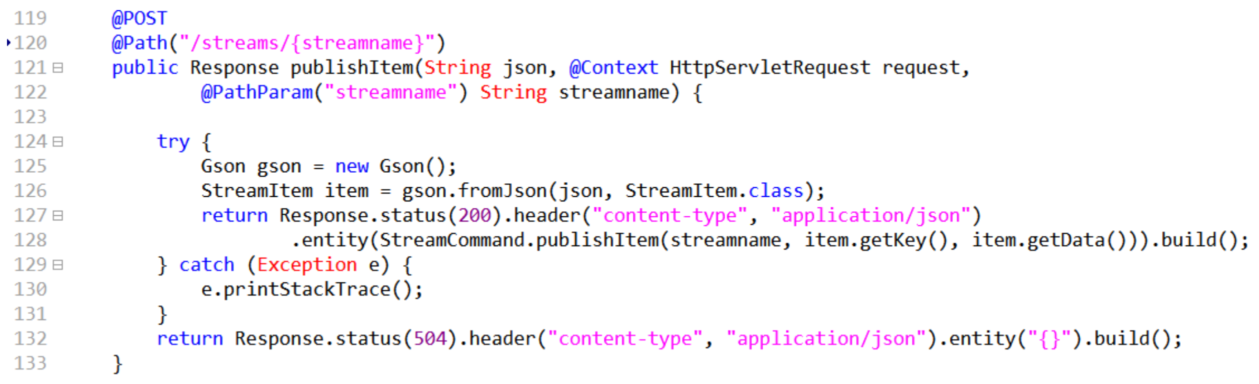


```go
//Object stored to Multichain
type MultichainObject struct {
    LessonInfo    string
    AccessControl string
}

//Assign Lesson
func (c *LessonsController) AssignLesson() {
    lessonId := c.Ctx.Input.Param(":lesson_id")
    lid, _ := strconv.Atoi(lessonId)
    v, err := models.GetLessonsById(lid)
    if err != nil {
        c.Data["json"] = err.Error()
    } else {
        lessonJsonStr, _ := json.Marshal(&v)
        if err != nil {
            c.Data["json"] = err.Error()
        } else {
            //Get Access Control JSON String
            accessControlJsonString := c.GetString("accessControlJsonString")
            //Assign MultichainObject
            multichainObject := MultichainObject{
                LessonInfo:    string(lessonJsonStr),
                AccessControl: accessControlJsonString,
            }
            multichainObjectString, _ := json.Marshal(&multichainObject)
            //Put multichainObject to Multichain Blockchain
            b := httplib.Post("localhost/streams/test")
            b.Param(strconv.Itoa(lid), string(multichainObjectString))
        }
    }
    c.ServeJSON()
}
```

Figure 33: Code snippet of assigning lesson

In this piece of code, httplib is used as a third-party library to send a POST request to specified URL and at the same time, MultichainObjectString is also sent as a parameter. When the Multichain Blockchain client receives the two JSON strings containing the POST parameter, it will push and publish it to the peer-to-peer network. Figure 34 shows the implementation of "/streams/{streamname}" function. It is written in Java language in Eclipse development tool.



```java
@POST
@Path("/streams/{streamname}")
public Response publishItem(String json, @Context HttpServletRequest request,
        @PathParam("streamname") String streamname) {

    try {
        Gson gson = new Gson();
        StreamItem item = gson.fromJson(json, StreamItem.class);
        return Response.status(200).header("content-type", "application/json")
                .entity(StreamCommand.publishItem(streamname, item.getKey(), item.getData())).build();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return Response.status(504).header("content-type", "application/json").entity("{}").build();
}
```

Figure 34: Code snippet of publishing item

Updating the private ledger through Multichain Blockchain peer-to-peer network

With the help of hands-on Multichain API available on https://www.Multichain.com/, such kind of process could be implemented perfectly. As explained in Section 3.3, in essence, Blockchain is a peer-to-peer distributed ledger that is cryptographically secure, append-only, immutable and updatable only via consensus or agreement among peers. Thus, in this process, two JSON files as shown in Figure 35 could be maintained securely by Multichain Blockchain platform.

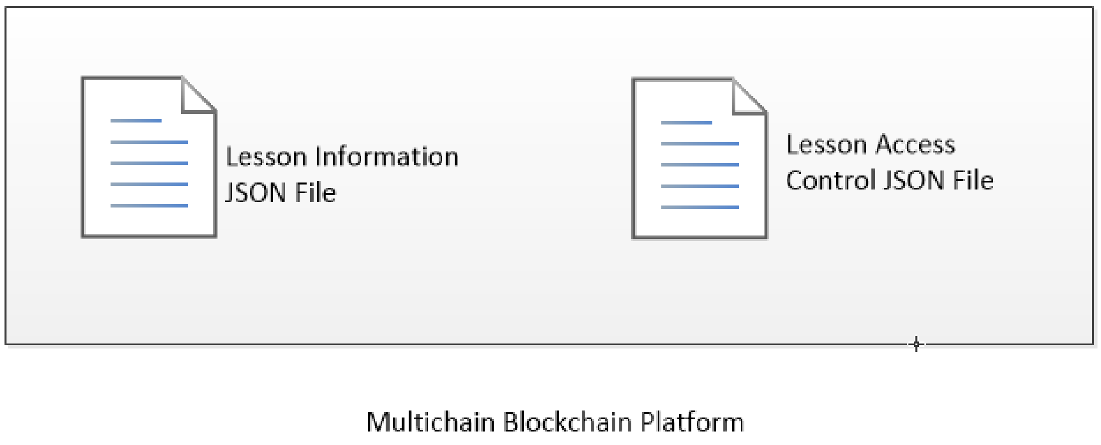

Lesson Information JSON File
Lesson Access Control JSON File
Multichain Blockchain Platform

Figure 35: JSON files stored in Multichain Blockchain platform

Consuming the E-learning resource through a Mobile Application

After assigning the E-learning resource, there will be two JSON files published in my Multichain Blockchain platform. The last process is to consume this E-learning resource. A mobile application on Android platform will be adopted to implement this function. Android Studio is used as a developing tool for building my mobile application called Lesson Basket. It is worth noting that only the model of consuming the E-learning resource will be focused in this part excluding other functions such as login, managing account, finishing quiz and etc. Figure 36 shows the first screen page of Lesson Basket. This application is designed for the aboriginal community in northern Saskatchewan and the first version has been implemented.

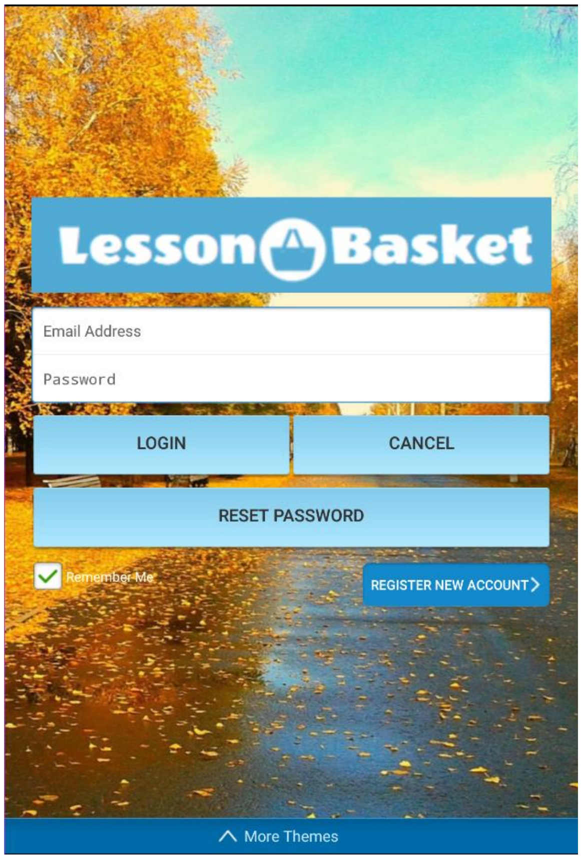


Figure 36: First screen page of Lesson Basket

Every E-learner has a user account and the account information will be stored in local database. As shown in Figure 27, there are eleven columns in Users table:

1) Id: the primary key of the user.
2) Firstname: the first name of the user.
3) Lastname: the last name of the user.
4) Email: the email of the user
5) Phone: the phone number of the user.
6) Score: the average scores of the user.
7) Token: when the E-learner successfully log in or retrieve the E-learning resource, it will be updated.

8) Grade: the grade of the user.
9) Password: the password of the user which is stored with an MD5 encryption algorithm.
10) Verified: a Boolean indicating whether the user has verified the email.
11) Image_url : the URL of user’s head portrait.
12) Displayname: the name used in my proposed E-learning platform which should be unique.
13) Created_date: the date of creation of the account which is automatically generated by Beego framework.
14) Last_login_date: the date of last login time.
15) Last_password_change: the date of last changing the password.
16) Schoolname: the school information of user.
17) Post_code: the post code.

It is noteworthy that displayname is the unique name referencing the name property of learning access control JSON file. Learning from OAuth2.0 explained in Section 3.2, the access token is used when the E-learner successfully login to the application. When the E-learning resource is uploaded to the local database and then published to Multichain Blockchain platform, the E-learner could send a GET request to the Beego framework API with user_id and lesson_id through an Android application. Figure 36 shows the lesson list screen where the E-learner could click the lesson and send the above request.

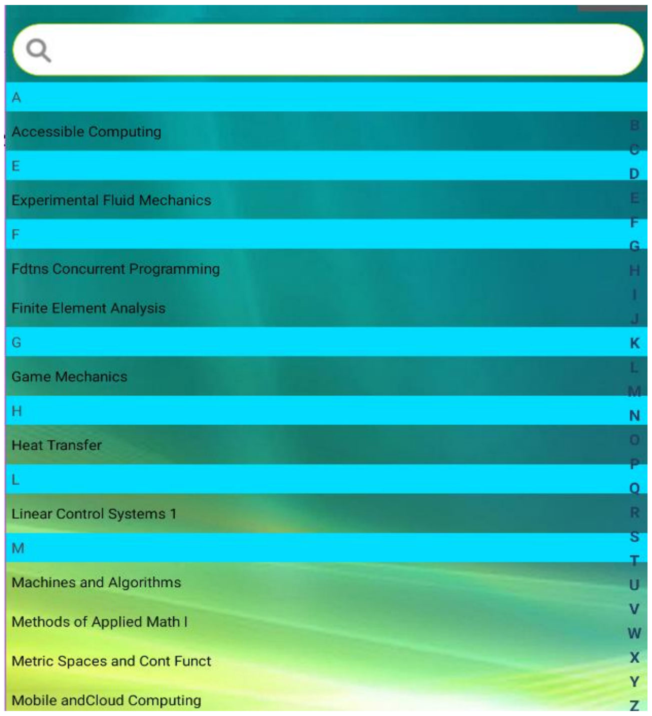
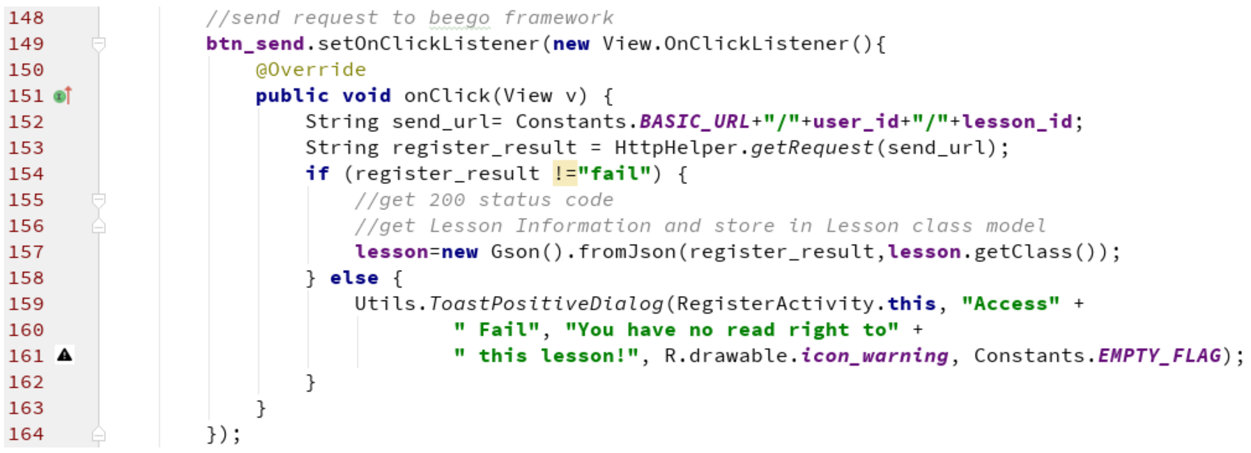

A
Accessible Computing
E
Experimental Fluid Mechanics
F
Fdtns Concurrent Programming
Finite Element Analysis
G
Game Mechanics
H
Heat Transfer
L
Linear Control Systems 1
M
Machines and Algorithms
Methods of Applied Math I
Metric Spaces and Cont Funct
Mobile andCloud Computing
B C D E F G H I J K L M N O P Q R S T U V W X Y Z

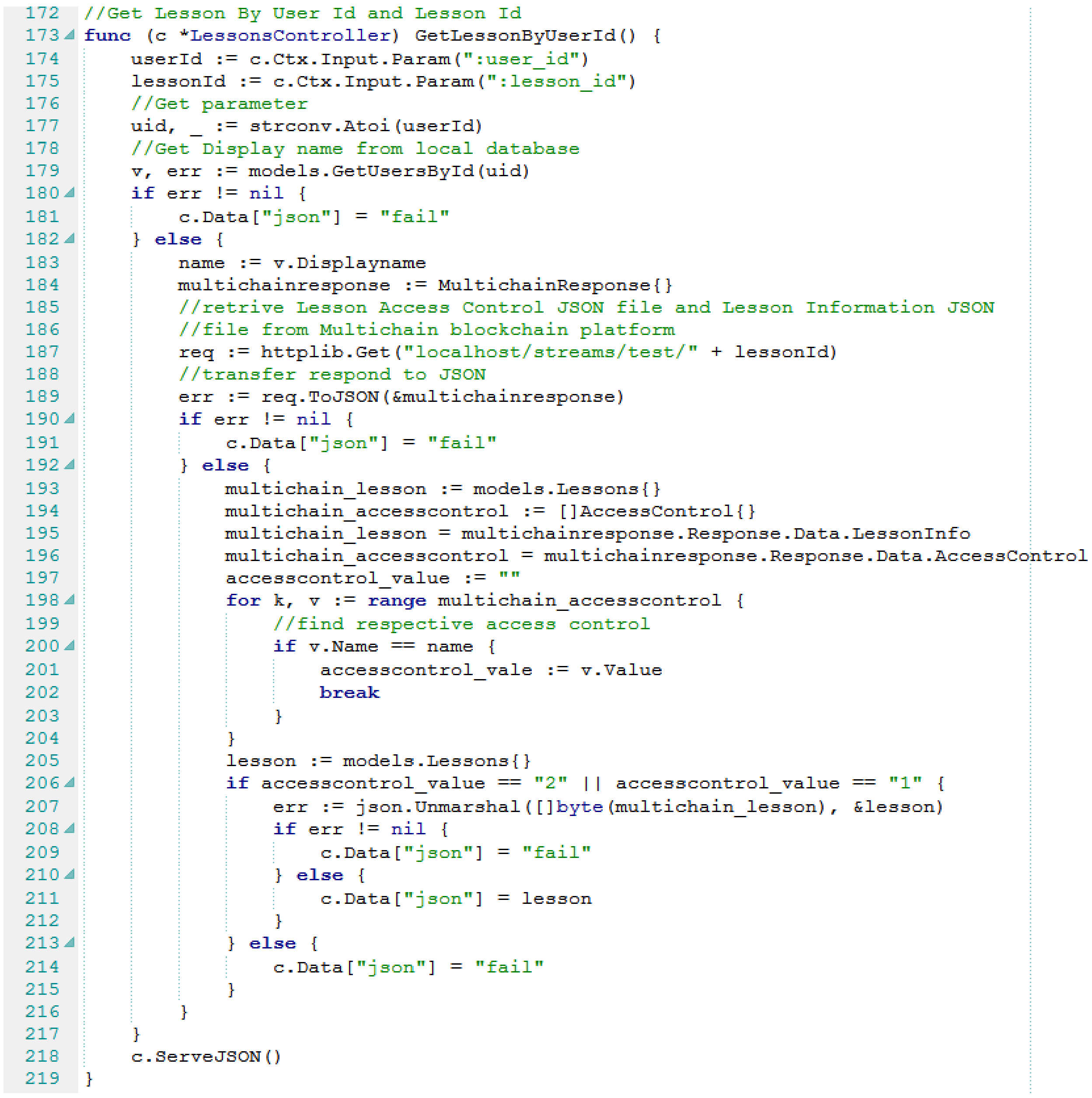
Figure 37: Lesson list screen page

Figure 38 captured the code snippet of sending the request in Java language in Android Studio while Figure 39 captured the code snippet of receiving the request in Beego framework.



```
//send request to beego framework
btn_send.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        String send_url= Constants.BASIC_URL+"/"+user_id+"/"+lesson_id;
        String register_result = HttpHelper.getRequest(send_url);
        if (register_result !="fail") {
            //get 200 status code
            //get Lesson Information and store in Lesson class model
            lesson=new Gson().fromJson(register_result,lesson.getClass());
        } else {
            Utils.ToastPositiveDialog(RegisterActivity.this, "Access" +
                    " Fail", "You have no read right to" +
                    " this lesson!", R.drawable.icon_warning, Constants.EMPTY_FLAG);
        }
    }
});
```

Figure 38: Code snippet of sending request



```go
//Get Lesson By User Id and Lesson Id
func (c *LessonsController) GetLessonByUserId() {
    userId := c.Ctx.Input.Param(":user_id")
    lessonId := c.Ctx.Input.Param(":lesson_id")
    //Get parameter
    uid, _ := strconv.Atoi(userId)
    //Get Display name from local database
    v, err := models.GetUsersById(uid)
    if err != nil {
        c.Data["json"] = "fail"
    } else {
        name := v.Displayname
        multichainresponse := MultichainResponse{}
        //retrive Lesson Access Control JSON file and Lesson Information JSON
        //file from Multichain blockchain platform
        req := httplib.Get("localhost/streams/test/" + lessonId)
        //transfer respond to JSON
        err := req.ToJSON(&multichainresponse)
        if err != nil {
            c.Data["json"] = "fail"
        } else {
            multichain_lesson := models.Lessons{}
            multichain_accesscontrol := []AccessControl{}
            multichain_lesson = multichainresponse.Response.Data.LessonInfo
            multichain_accesscontrol = multichainresponse.Response.Data.AccessControl
            accesscontrol_value := ""
            for k, v := range multichain_accesscontrol {
                //find respective access control
                if v.Name == name {
                    accesscontrol_vale := v.Value
                    break
                }
            }
            lesson := models.Lessons{}
            if accesscontrol_value == "2" || accesscontrol_value == "1" {
                err := json.Unmarshal([]byte(multichain_lesson), &lesson)
                if err != nil {
                    c.Data["json"] = "fail"
                } else {
                    c.Data["json"] = lesson
                }
            } else {
                c.Data["json"] = "fail"
            }
        }
    }
    c.ServeJSON()
}
```

Figure 39: Code snippet of receiving request

If the E-learner has respective reading or writing permission, the value of Access Control JSON file should be 1 (001) or 2 (010). Then the android client should receive the lesson information and show the lesson correspondingly. Otherwise, if the value is other numbers, the Beego framework will return “fail” in JSON format. It is worth noting that in this process, the Beego framework sends a GET request to Multichain Blockchain platform through URL: http://localhost/streams/test/{lesson_id}.

At last, the E-learner could play the lesson video by sending a request to video_url stored and finish the quiz after watching the lesson. The playing lesson interface in Android Platform is shown in Figure 40.

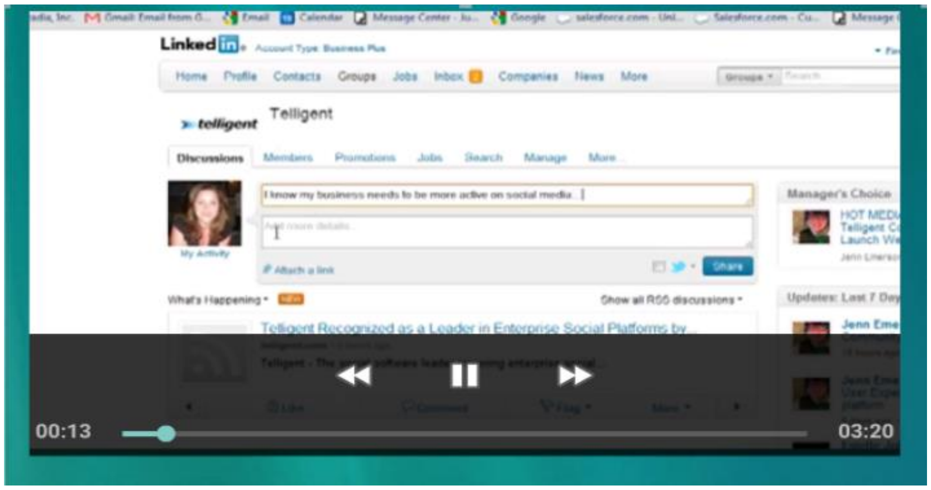


Figure 40: Lesson video screen page

3.5 Summary

In my proposed E-learning platform, research question 1 is solved by eliminating central authority server on Multichain Blockchain platform and all the E-learners constitute a peer-to-peer network. The problem of single point of failure would be solved and the cost of purchasing and maintaining a high-performance will be minimized. Meanwhile, the network efficiency will be increased. Building on top of Multichain Blockchain platform, decentralized access control becomes possible since the access control JSON file is stored separately in each node and maintained by the distributed consensus mechanism in a secure way. Besides, the simplification of JSON format makes the access control file straightforward and easy to understand. It solves research question 2 elegantly. As explained in Section 3.3, Blockchain can be defined as a peer-to-peer distributed ledger that is cryptographically secure, append-only, immutable and updatable only via consensus or agreement among peers. Thus, the authenticity, integrity, non-repudiation as well as traceability of E-learning resource could be ensured. The reason why only two JSON files are stored in Multichain Blockchain platform is that the latency will become unbearably long if more files are published. Besides, the capacity of each node will become bulky. In my proposed E-learning platform, it is significant to keep the privacy and security of access control and lesson information files.

CHAPTER 4
Performance Evaluation

To evaluate the performance (research question 1) of using Blockchain to create a decentralized access control system, the key performance metric: *average response time* has been evaluated in this chapter. The same experiment will be conducted in two different environments: LAN and Amazon Web Service. As explained in the last chapter, there are two main processes should be evaluated: creating the E-learning resource (for AssignLesson function mentioned above) and consuming the E-learning resource (for GetLessonByUserId function mentioned above). Thus, the following four experiments are planned and executed.

Experiments	**Description**
Access control and Blockchain nodes deployed in the process of creating E-learning resource in Local Area Network	Test the key performance metric: average response time on the Local Area Network in creating E-learning resource stage
Access control and Blockchain nodes deployed in the process of consuming E-learning resource in Local Area Network	Test the key performance metric: average response time on the Local Area Network in consuming E-learning resource stage
Access control and Blockchain nodes deployed in the process of creating E-learning resource on Amazon Web Services	Test the key performance metric: average response time on the AWS in creating E-learning resource stage
Access control and Blockchain nodes deployed in the process of consuming E-learning resource on Amazon Web Services	Test the key performance metric: average response time on the AWS in consuming E-learning resource stage

Table 1: Descriptions of experiments

The first research question can be solved by using Multichain Blockchain platform while the second research question can be solved by adding extra servers in a peer-to-peer E-learning platform. Figure 41 shows the technological stacks of the experiment.

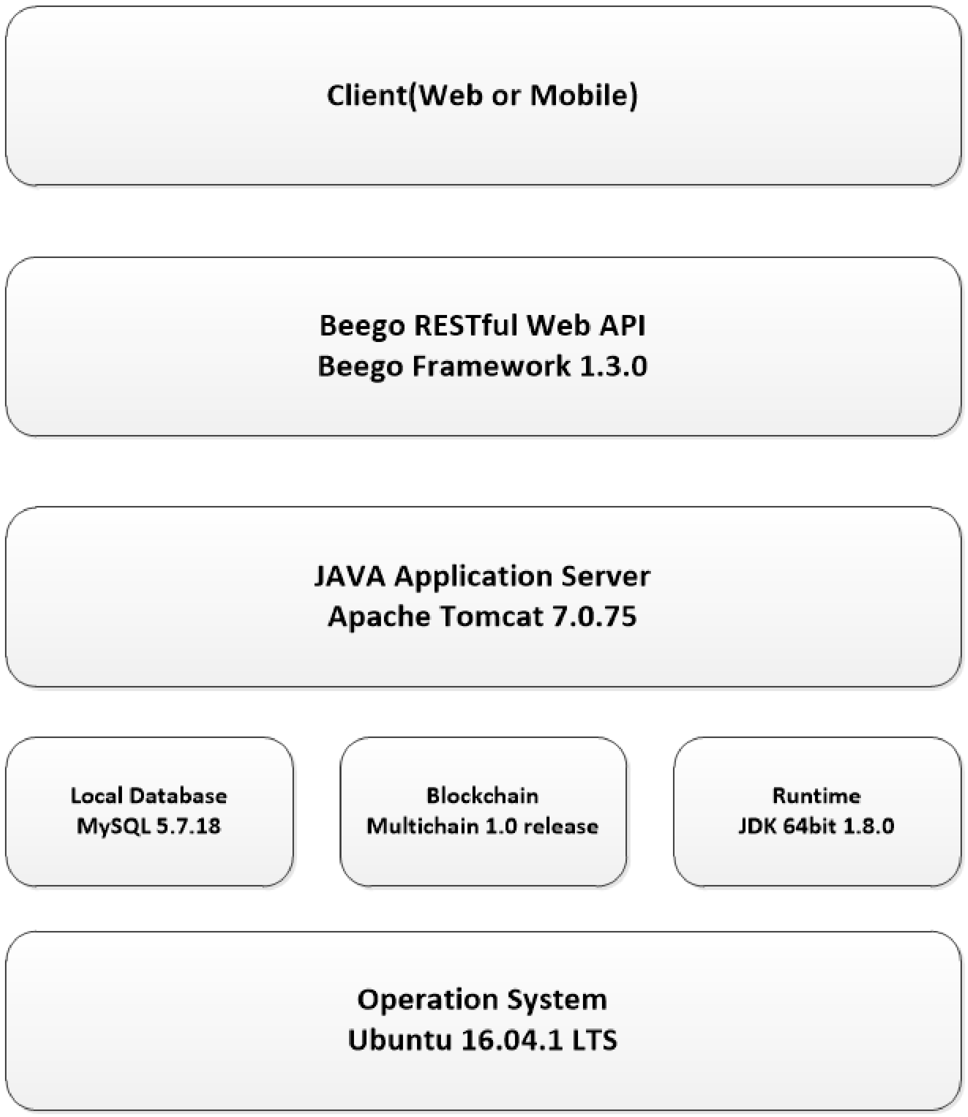


Figure 41: Technological stacks

In terms of the LAN experiment, university's network will be used. JMeter, a performance testing tool, is installed on a Windows machine in order to execute the performance evaluation and record the results.

A number of experiments are carried out to assess the performance of my proposed E-learning platform. The key performance metric: *average response time* which means the time the server takes to respond to a request has been analyzed graphically. In the real world, this is a significant metric since longer response time may passively impact user experience and further cause HTTP request timeout. For the sake of simulating the real condition, the platform is evaluated with 1, 30, 60, 120, 240 and 480 clients sending the requests simultaneously. This reflects how users load affects the performance of E-learning platform. Considering the fact that the delay may exist in the real network, delays of 0ms, 250ms and 500ms are added into the experiment. In other words, the same experiment will be conducted three times (with no delay, 250ms delay as well as 500ms delay). In order to compare the difference between the single server model and the peer-to-peer network model with 3 nodes, in one experiment, both of them are evaluated. It is worth noting that here the single server model represents that there is only one server maintaining the ledger in my Multichain Blockchain platform (which could partially reflect the architecture of central server). By contrast, peer-to-peer network model contains three nodes synchronizing the ledger. Each result graph will show two different curved lines representing these two models.

4.1 Experiment 1: Local area network

The first experiment is conducted in the university's local area network. The setup consists of one Windows machine and three Linux server machines. The Windows machine hosts JMeter testing tool and simulates different amount of clients sending requests. Linux server machine hosts Multichain Blockchain node as a distributed ledger, lesson information JSON files as well as access control JSON files. It is shown in Figure 41.

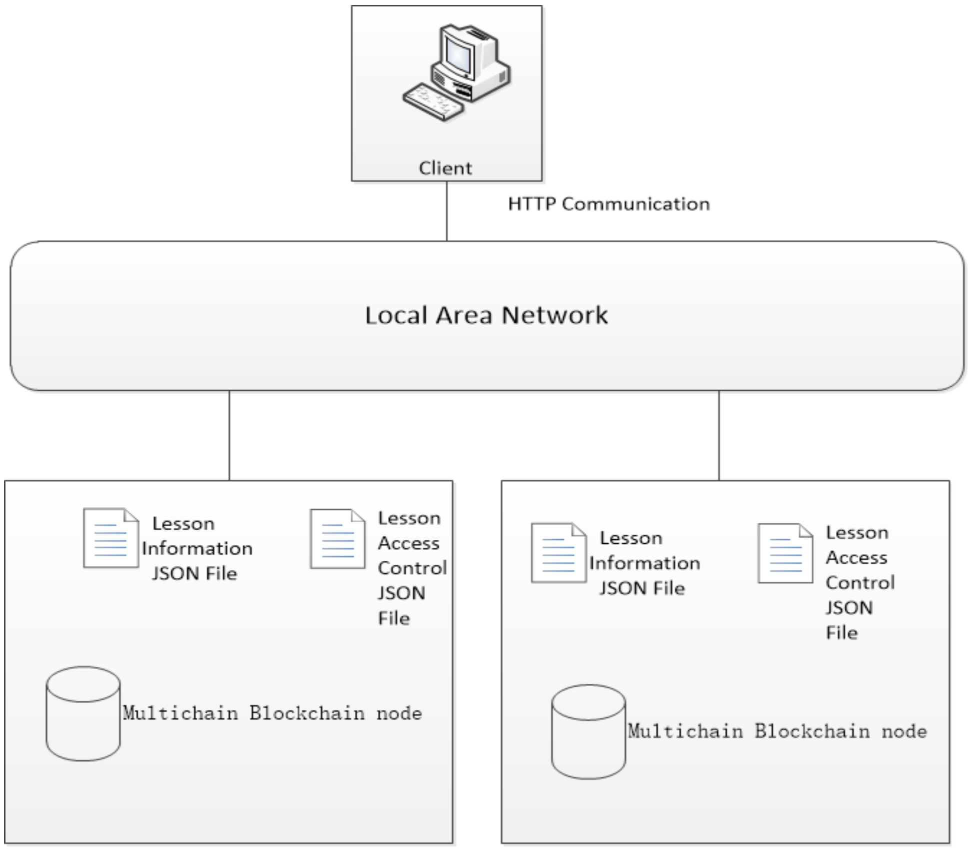

Client
HTTP Communication
Local Area Network
Lesson Information JSON File
Lesson Access Control JSON File
Multichain Blockchain node
Lesson Information JSON File
Lesson Access Control JSON File
Multichain Blockchain node

Figure 42: Experimental setup for Experiment 1 in LAN

Table 2 shows the hardware specification of the setup environment.

Hardware	Specifications
Windows Client	**OS**: Windows 10 **Processor**: Intel(R) Core(TM) i7-4702MQ **Memory**: 8GB RAM
Linux Node 1	**OS**: Ubuntu 16.04 LTS **Processor**: Intel(R) Core i7-6700 **Memory**: 4GB RAM
Linux Node 2	**OS**: Ubuntu 16.04 LTS **Processor**: Intel(R) Core i7-6700 **Memory**: 4GB RAM
Linux Node 3	**OS**: Ubuntu 16.04 LTS **Processor**: Intel(R) Core i7-6700

	Memory: 4GB RAM

Table 2: Hardware specifications for experiment 1

1. Average response time in creating the E-learning resource in the LAN without delay

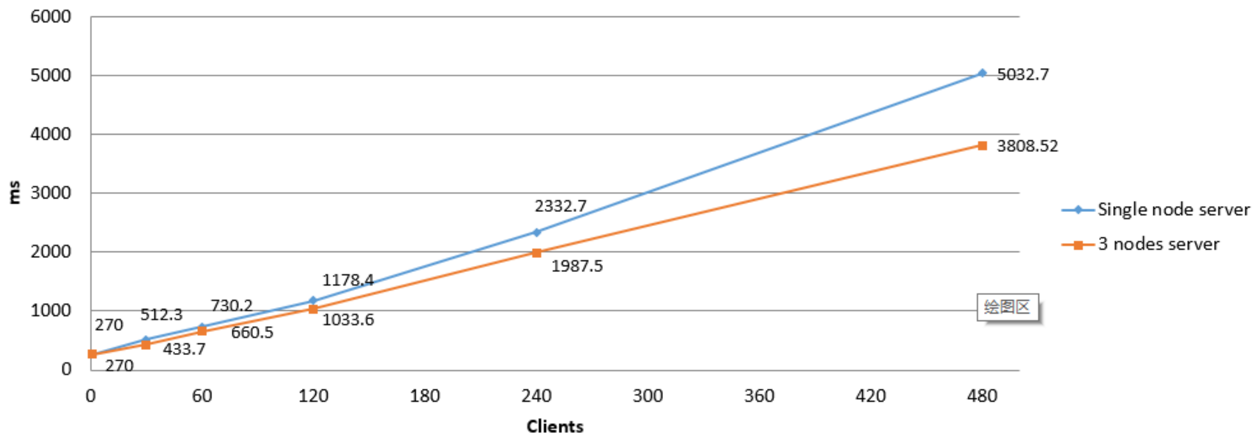


Figure 43: Average response time in creating the E-learning resource in the LAN without delay

In the process of creating the E-learning resource, the average response time in both cases (single server and peer-to-peer network with three nodes) remains the same (270ms) when there is only one client connecting with E-learning platforms. With the number of clients increasing, the gap between the average response time in these two cases dramatically grows. Average response time for single server reaches 5032.7ms when the number of clients is 480 while the average response time for the peer-to-peer model is just 3808.52ms. It clearly demonstrates the horizontal scalability could be extended by increasing the number of servers.

2. Average response time in consuming the E-learning resource in the LAN without delay

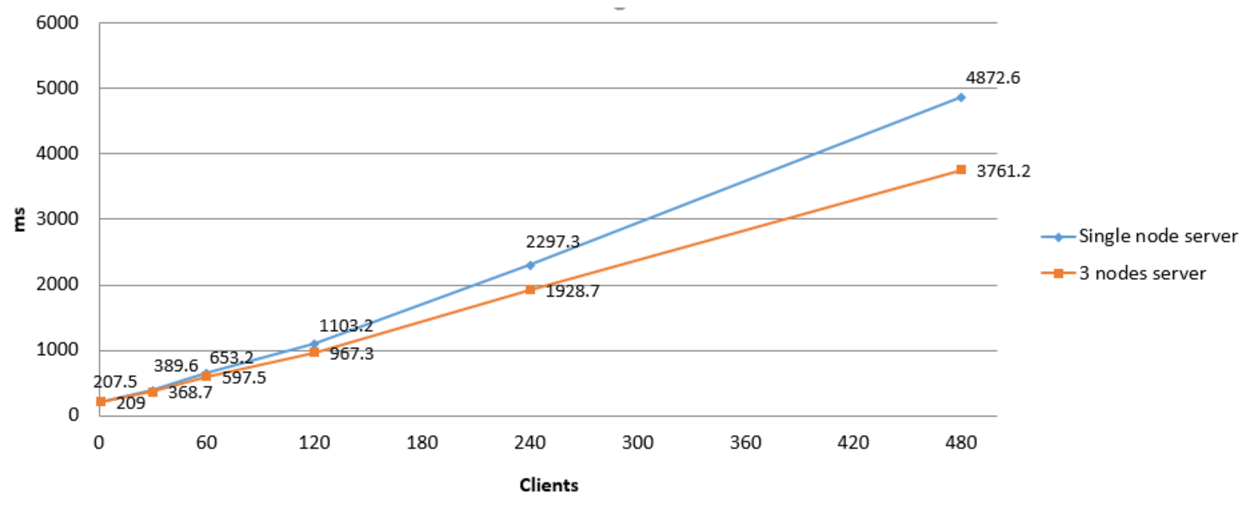


Figure 44: Average response time in consuming the E-learning resource in the LAN without delay

Compared with the process of creating the E-learning resource, the stage of consuming the E-learning resource has lower average response time. When there is only one client connecting to the platform, the average response time for the single server model is 207.5ms versus to 209ms for three nodes model. However, as the number of clients rises, the average response time for the single server is much longer than that for three nodes model. When the number of clients reaches 480, the average response time for the single server (4872.6ms) is almost one fourth more than that for the three nodes server (3761.2ms). This further demonstrates the huge potential of horizontal scalability of the peer-to-peer network by using Blockchain.

3. Average response time in creating the E-learning resource in the LAN with 250ms delay

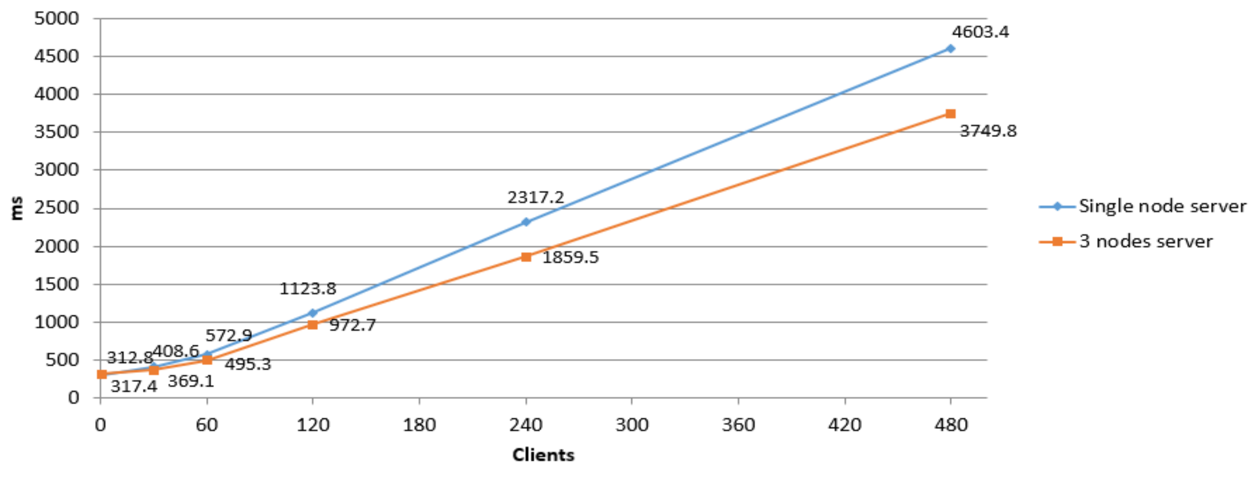


Figure 45: Average response time in creating the E-learning resource in the LAN with 250ms delay

On the whole, the average response time in creating the E-learning resource with 250ms delay reduces slightly compared with that without delay. To be specific, the average response time for the single server is 312.8ms versus to 317.4ms for three nodes network when there is only one client connecting with E-learning platforms. But when the number of clients is 30, the average response time for three nodes network (369.1ms) is less than that for the single server (408.6ms). With the number of clients increasing, the gap between these two values enlarges. It is noteworthy that the average response time for the single server is 4603.4ms, almost 900ms more than that for the three nodes (3947.8ms). In brief, with 250ms delay, the performance of E-learning platforms improves a little bit.

4. Average response time in consuming the E-learning resource in the LAN with 250ms delay

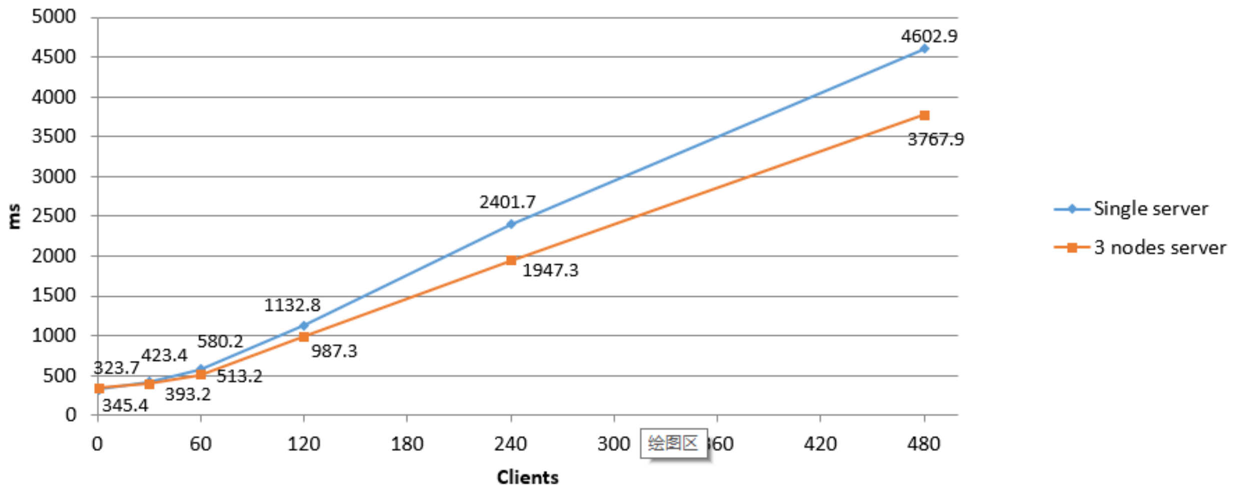


Figure 46: Average response time in consuming the E-learning resource in the LAN with 250ms delay

In the process of consuming the E-learning resource, the average response time for the single server and three nodes is 323.7ms and 345.4, respectively when there is only one client connecting to the platform. With the number of clients increasing, the average response time for three nodes becomes less than that for the single server and the gap becomes widen after the number of clients reaches 30. Similar to the situation of creating the E-learning resource, with 250ms delay, the performance of E-learning platforms also improves a little bit.

5. Average response time in creating the E-learning resource in the LAN with 500ms delay

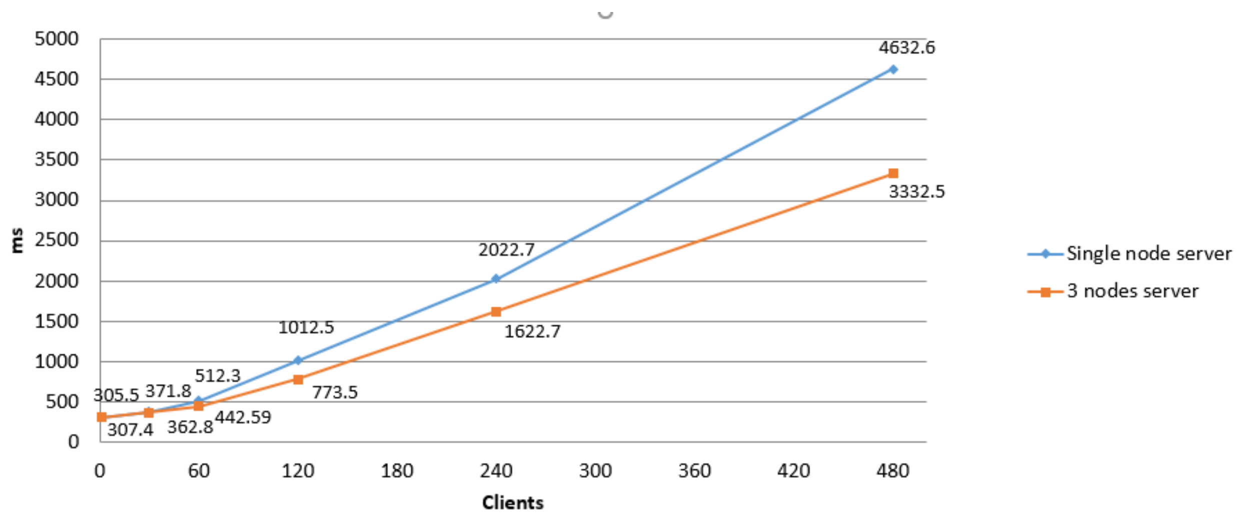


Figure 47: Average response time in creating the E-learning resource in the LAN with 500ms delay

The average response time for the single server over the LAN with 500ms delay is almost identical with that for the three nodes model (305.5ms compared with 307.4ms) when there is only one client. With a load of clients increasing, the three nodes model again outperforms the single server. When the client number reaches 480, the average response time for the single server and three nodes model becomes 4632.6ms and 3332.5ms, respectively. It is worth noting that for the central server, the system performance with 500ms delay has not improved but remains the same compared with that with 250ms delay but improves a bit compared that without delay. Nevertheless, for the three nodes model with 500ms delay, the system performance improves a bit compared with both 0ms delay and 250ms delay.

6. Average response time in consuming the E-learning resource in the LAN with 500ms delay

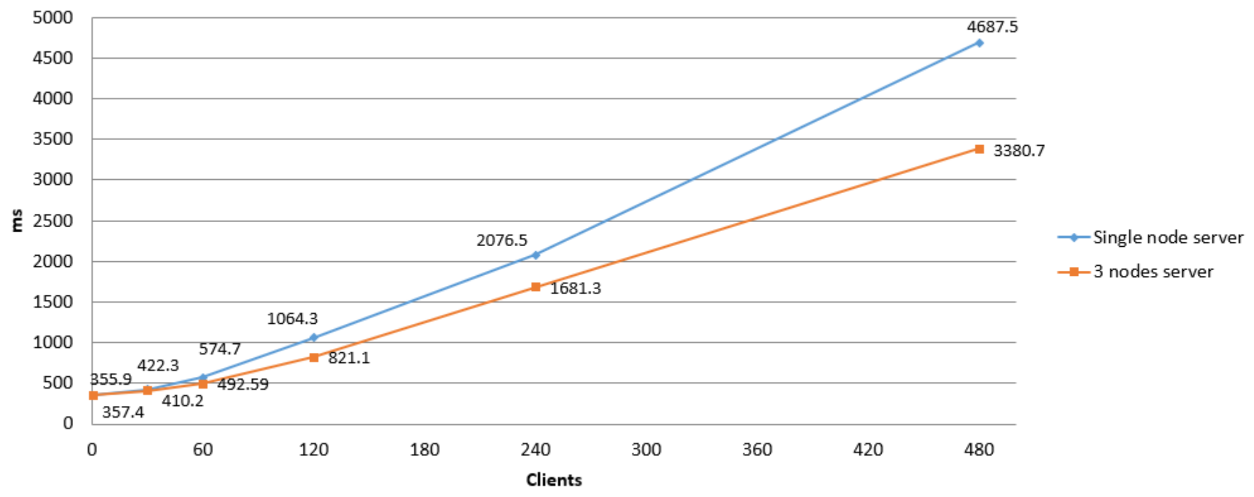


Figure 48: Average response time in consuming the E-learning resource in the LAN with 500ms delay

Overall, the system performance in consuming the E-learning resource with 500ms delay also improves to a small degree. When there is only one client connecting with E-learning platforms, the average response time with 500ms delay is 355.9ms for one server compared with 357.4ms for three nodes model. However, when the number of clients reaches 480, the gap between the central server and three nodes model is almost 1300ms (4687.5 ms and 3380.7ms, respectively). It is worth noting that with the delay increases, the system performance for three nodes model improves more compared with that for the single server model. This also demonstrates the tremendous potential of using Blockchain to build a peer-to-peer E-learning platform since the number of E-learners is always in a large base.

4.2 Experiment 2: Cloud running environment - AWS

The second experiment is conducted on Amazon Web Services. Amazon Web Services offers reliable, scalable, and inexpensive cloud computing services which makes it one of the most popular cloud web services in the world. Figure 42 shows the experimental setup.

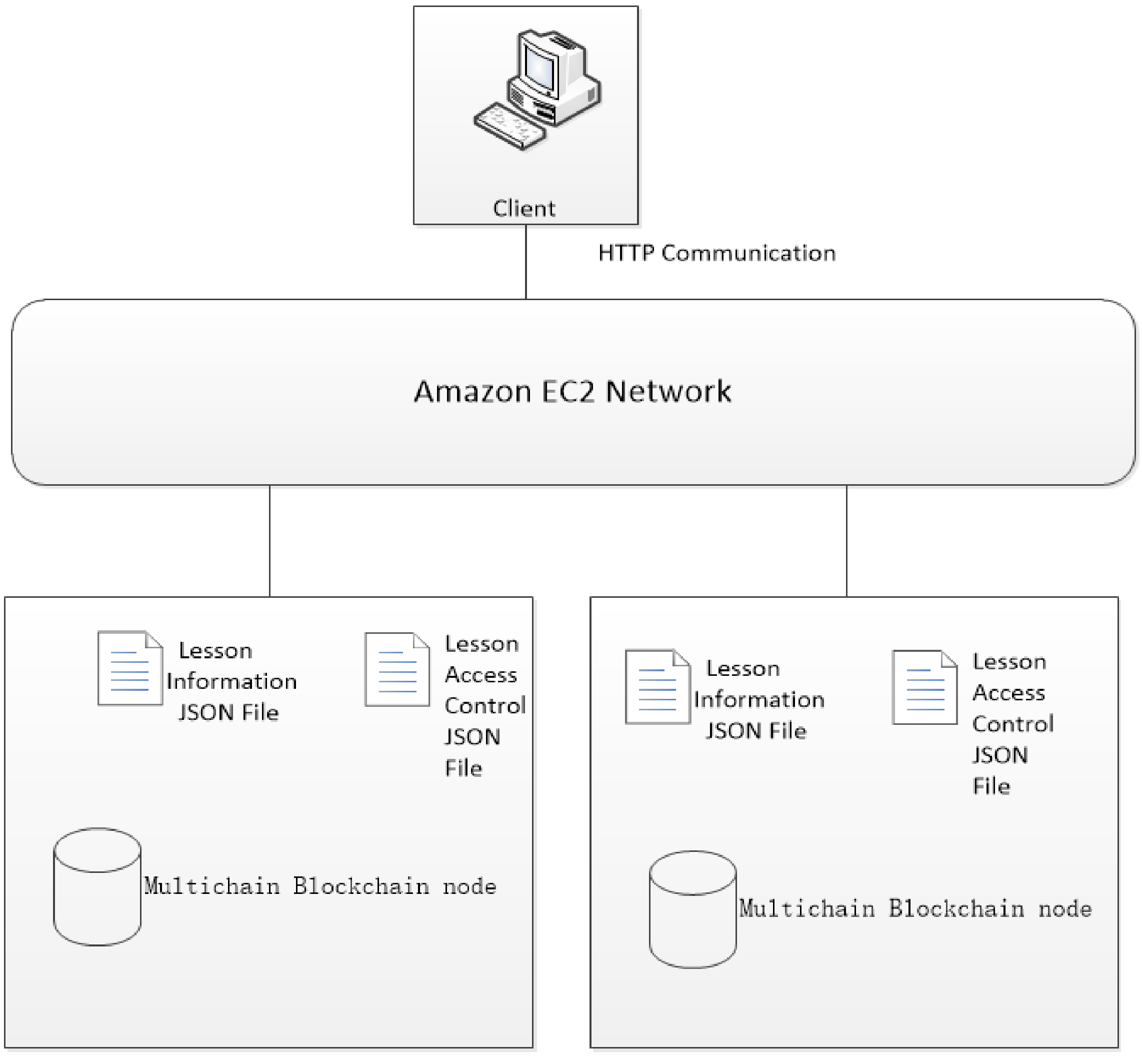

Client
HTTP Communication
Amazon EC2 Network
Lesson Information JSON File
Lesson Access Control JSON File
Multichain Blockchain node
Lesson Information JSON File
Lesson Access Control JSON File
Multichain Blockchain node

Figure 49: Experimental setup for experiment 2 on AWS

Table 3 shows the hardware specifications of experiment 2.

EC2 Instances	Specifications
Instance 1	**Instance type**: t2.medium **Availability zone**: us-east-1b **OS**: Ubuntu 16.04 LTS **Processor**: Intel(R) Xeon(R) CPU E5-2676 v3 **Memory**: 8GB RAM

Table 3: Instance specifications for experiment 2

1. Average response time in creating the E-learning resource on AWS without delay

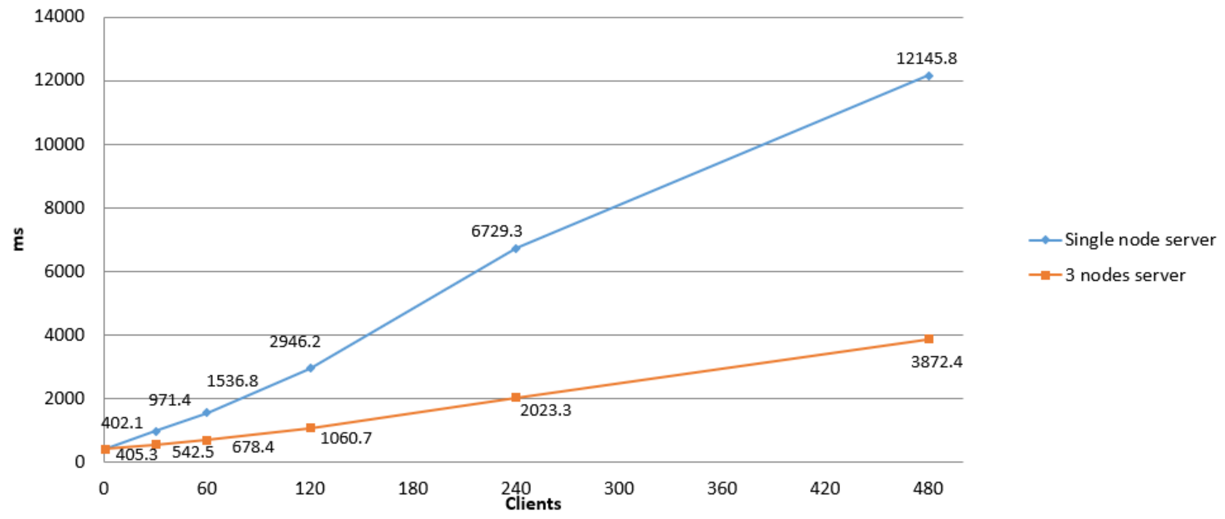


Figure 50: Average response time in creating the E-learning resource on AWS without delay

Compared with the LAN setup, the average response time in creating the E-learning resource on AWS is longer. In terms of the single server, the average response time with no delay ranges from 402.1ms to 12145.8ms. By contrast, in regard to the three nodes model, the average response time ranges from 405.3ms to 3872.4ms. It is clear that three nodes configuration performs much better than the single server configuration (3872.4ms versus to 12145.8ms). Meanwhile, on AWS, when the number of clients is large enough (480 clients), the average response time with no delay is around 70ms slower than that in the LAN (3808.52ms). It seems that building E-learning platforms in a peer-to-peer network by using Blockchain is indeed a reliable way to extend the horizontal scalability and further improve the performance in a cloud environment.

2. Average response time in consuming the E-learning resource on AWS without delay

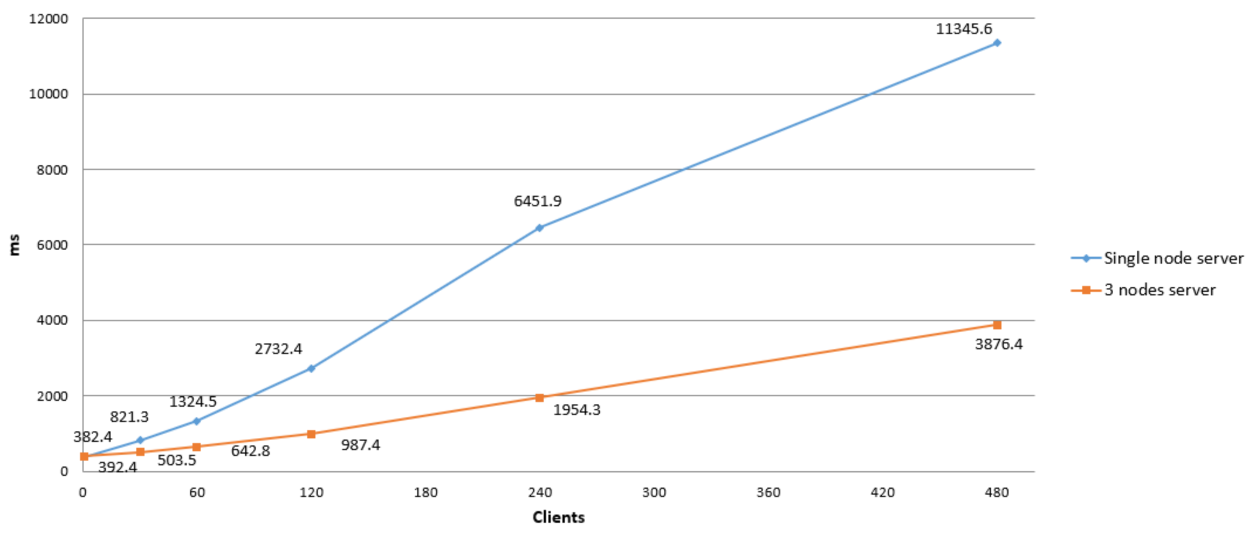


Figure 51: Average response time in consuming the E-learning resource on AWS without delay

When it comes to the process of consuming the E-learning resource, the overall average response time for both one server and three nodes is a little less than that in creating the E-learning resource. When there is only one client connecting with E-learning platforms through AWS, the average response time for one server (382.4ms) is 10ms quicker than that for the three nodes. Nevertheless, when the number of clients reaches 480, the average response time for one server (11345.6ms) is much slower than that (3876.4ms) for three nodes model. It could demonstrate that similar to the process of creating the E-learning resource on AWS, the system horizontal scalability and performance could improve a lot through adopting the peer-to-peer network.

3. Average response time in creating the E-learning resource on AWS with 250ms delay

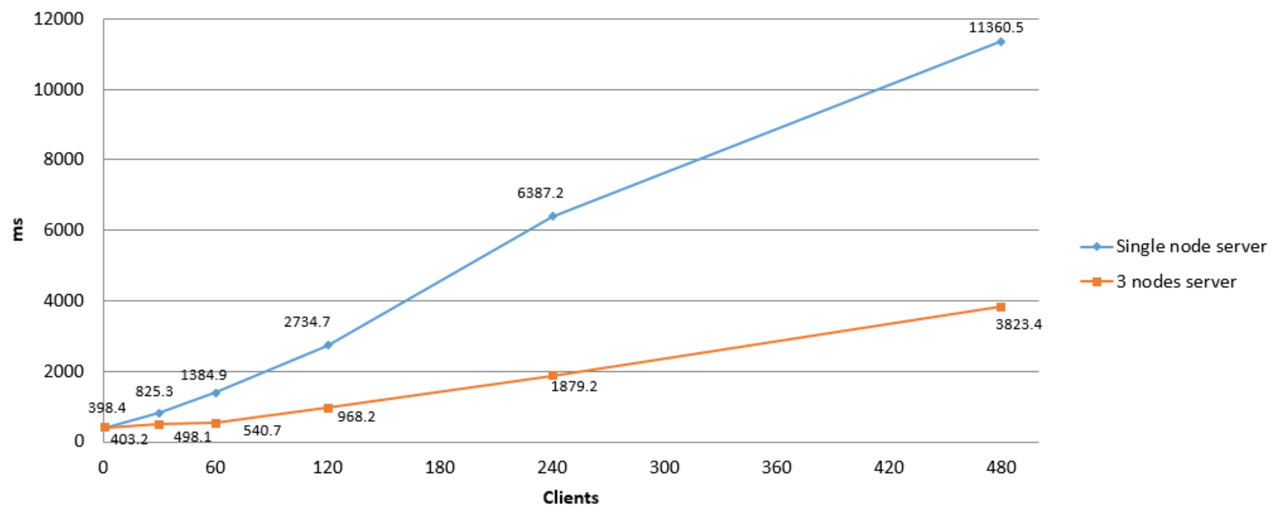


Figure 52: Average response time in creating the E-learning resource on AWS with 250ms delay

The trend of average response time in creating the E-learning resource on AWS with 250ms delay is closely similar to the same condition with no delay, ranging from 398.4ms for the single server, 403.2ms for the three nodes model to 11360.5ms and 3823.4ms, respectively. Even though the difference is slight, the delay of 250ms indeed improves the performance of E-learning platforms when compared with no delay.

4. Average response time in consuming the E-learning resource on AWS with 250ms delay

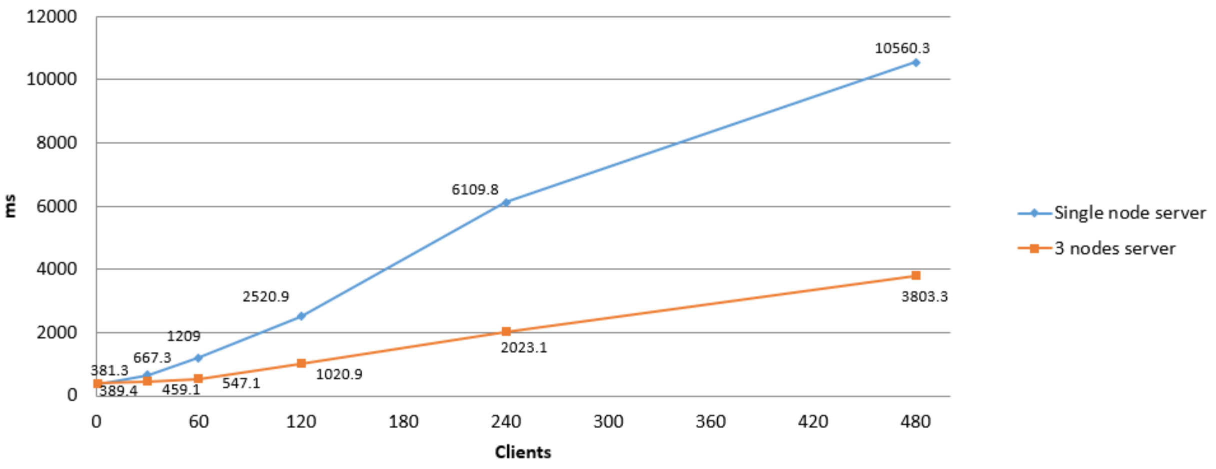


Figure 53: Average response time in consuming the E-learning resource on AWS with 250ms delay

In terms of consuming the E-learning resource, the overall trend is very similar to creating the E-learning resource. When there is only one client, the average response time is 381.3ms and 389.4ms for a single server and three nodes, respectively. However, when the client number becomes 60, the average response time for one server (1209ms) is twice than that (547.1ms) for three nodes. With the number of clients increasing, the gap enlarges dramatically. When the number of clients reaches 480, the average response time is 10560.3ms and 3803.3ms, respectively. This demonstrates that in consuming the E-learning resource, the system performance is improved by adopting the peer-to-peer network.

5. Average response time in creating the E-learning resource on AWS with 500ms delay

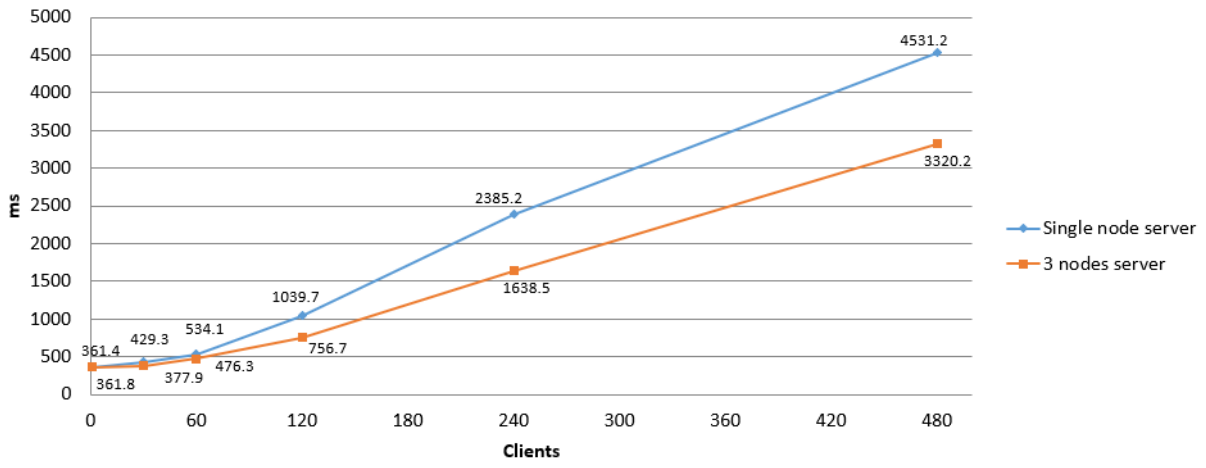


Figure 54: Average response time in creating the E-learning resource on AWS with 500ms delay

It is interesting that with 500ms delay in creating the E-learning resource on AWS, the system performance for one server boosts a lot ranging from 361.4ms with 1 client to 4531.2ms with 480 clients. However, the three nodes model also outperformed single server model (3320.2ms for 480 clients connecting). Meanwhile, by increasing the delay from 250ms to 500ms, the system performance also improves a bit.

6. Average response time in consuming the E-learning resource on AWS with 500ms delay

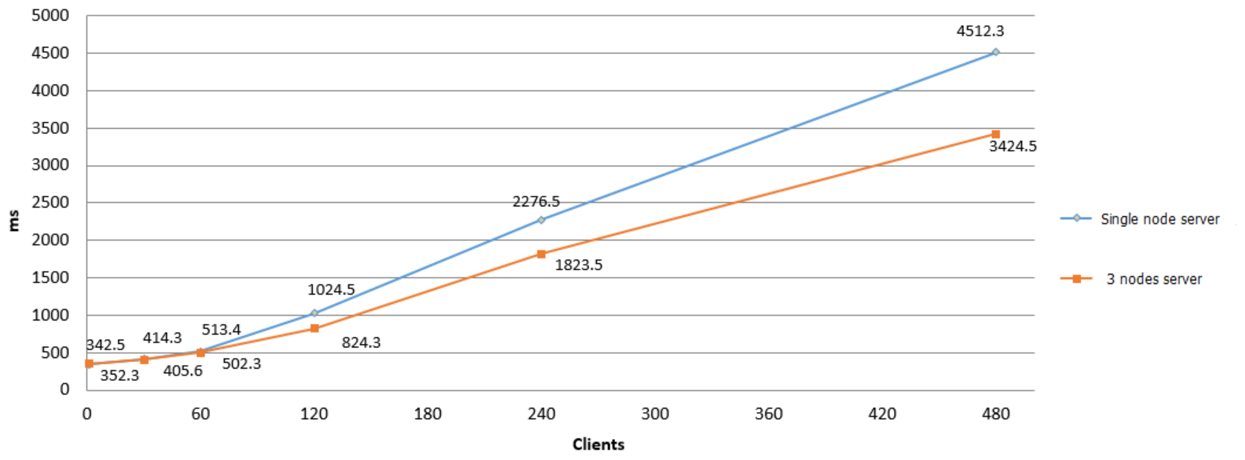


Figure 55: Average response time in consuming the E-learning resource on AWS with 500ms delay

Like the trend of creating the E-learning resource on AWS with 500ms delay, in the process of consuming the E-learning resource, the average response time ranges from 342.5ms for a single server and 352.3ms for three nodes model when there is only one client. When the number of clients reaches 480, the average response time for one server is 4531.2ms, around 1100ms slower than that for three nodes (3424.5ms).

4.3 Summary

The last two sections have compared the single server node model and three server nodes in 0ms, 250ms and 500ms delay in the processes of both creating and consuming the E-learning resources. In order to compare the different environment (LAN and AWS), the following four bar charts graphically show the result. It is worth noting that there is no need to compare the average response time of different delays since extending the delay would just have a subtle influence on the system performance.

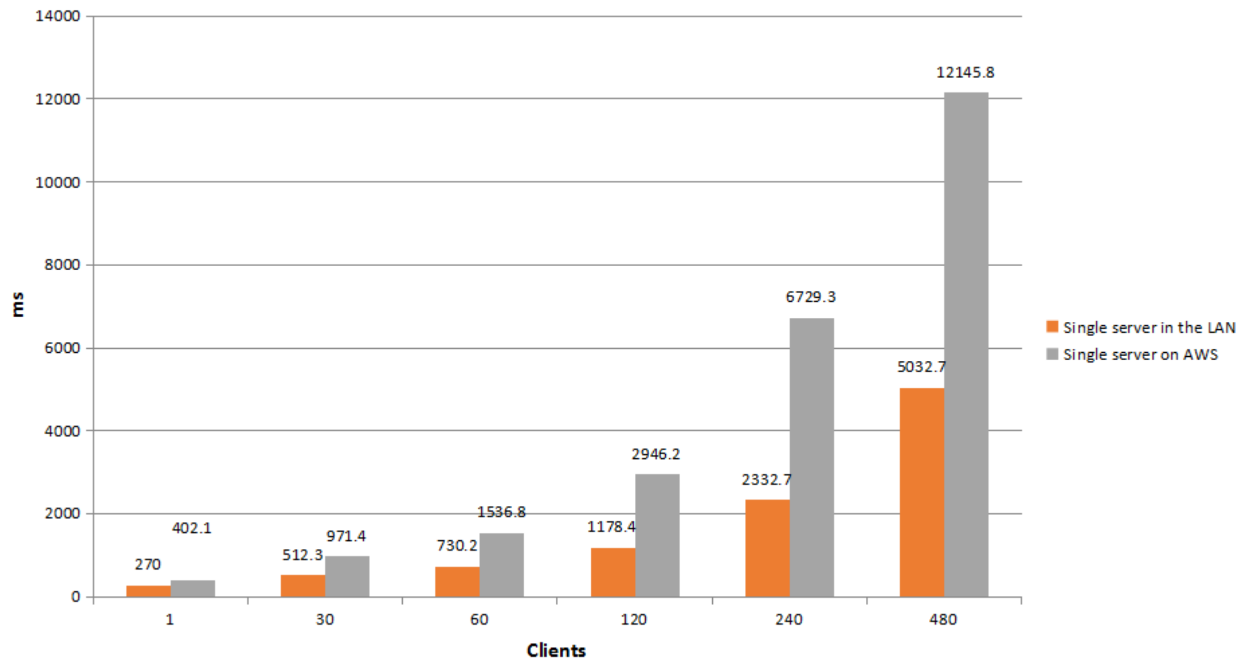


Figure 56: Average response time comparisons between LAN and AWS in creating the E-learning resource without delay for single server

In the process of creating the E-learning resource, the gap of average response time for the single server between LAN and AWS enlarges with the number of clients increases. When there is only one client, the average response time in the LAN is 270ms versus to 402.1ms on AWS. However, when the number of clients reaches 240, the average time of AWS (6729.3ms) is almost three times larger than that in the LAN (2332.7ms).

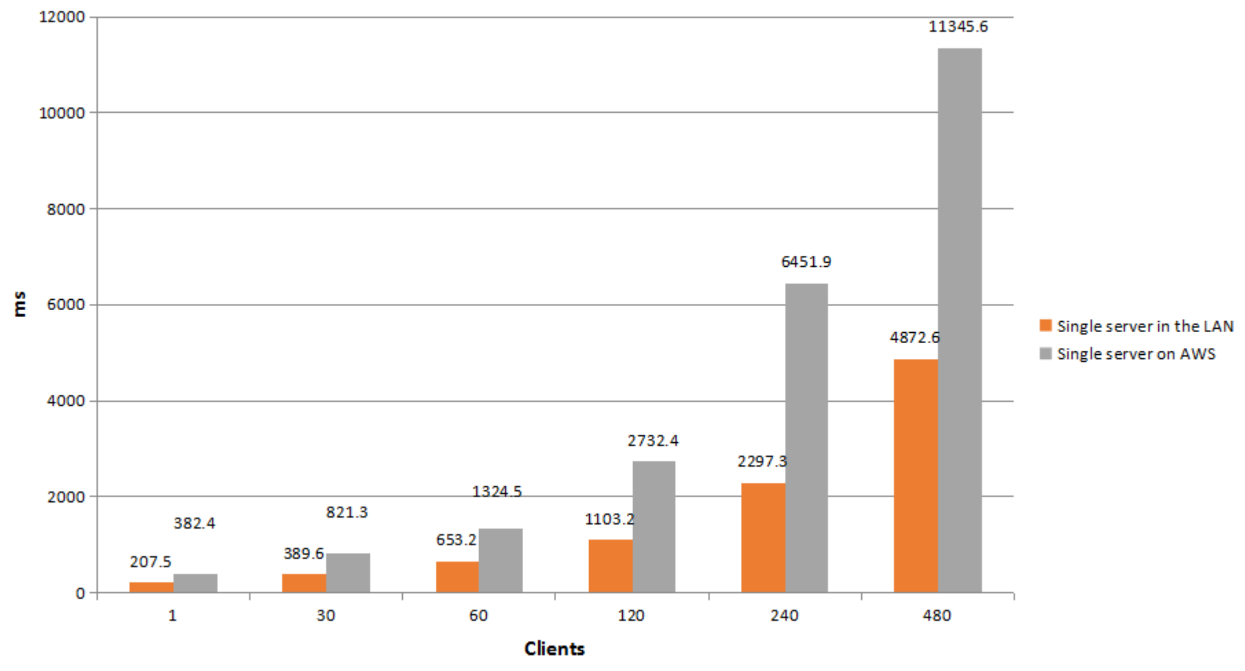


Figure 57: Average response time comparisons between LAN and AWS in consuming the E-learning resource without delay for single server

In the process of consuming the E-learning resource, the trend is almost the same with that in creating the E-learning resource. The average response time for one client in terms of LAN and AWS is 207.5ms and 382.4ms, respectively. When the number of clients gets to 480, the respective response time is 4872.6ms versus to 11345.6ms.

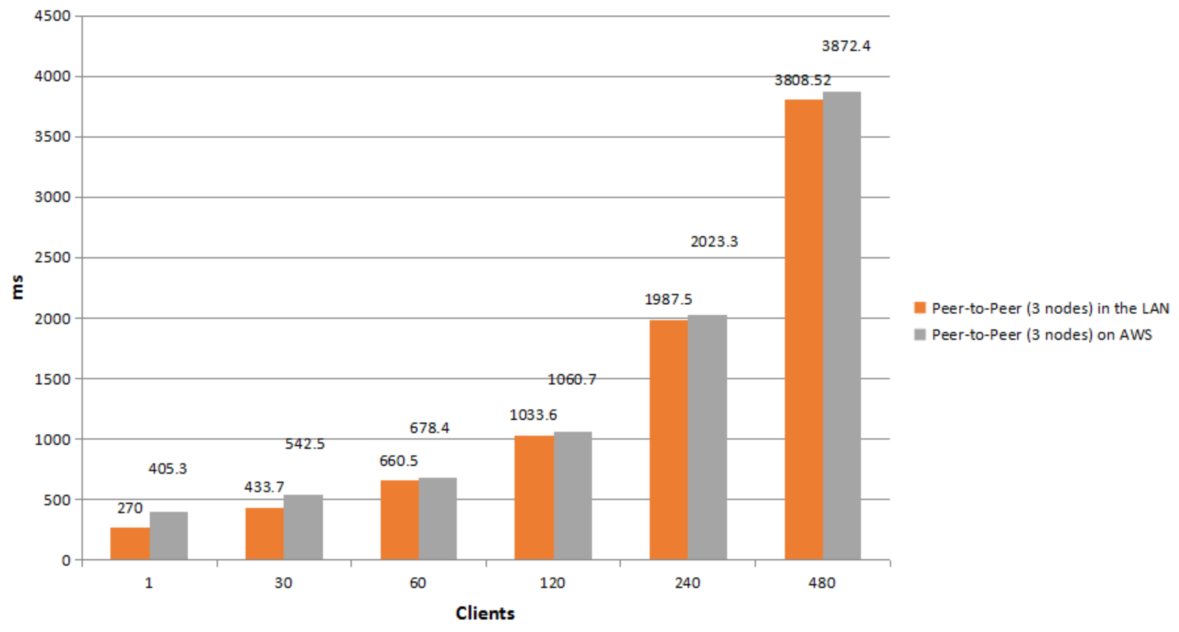


Figure 58: Average response time comparisons between LAN and AWS in creating the E-learning resource without delay for peer-to-peer (3 nodes)

In the situation of peer-to-peer (3 nodes) model, the gap of average response time between LAN and AWS in creating the E-learning resource without delay is almost insignificant especially when the client number is large enough (over 240). When there is only one client, the gap between these two values is 135.3ms (270ms versus to 405.3ms). Nevertheless, when the number of clients reaches 480, the gap becomes nearly 60ms (3808.52ms versus to 3872.4ms). This demonstrates that the horizontal scalability extends sharply by adding nodes in Cloud environment.

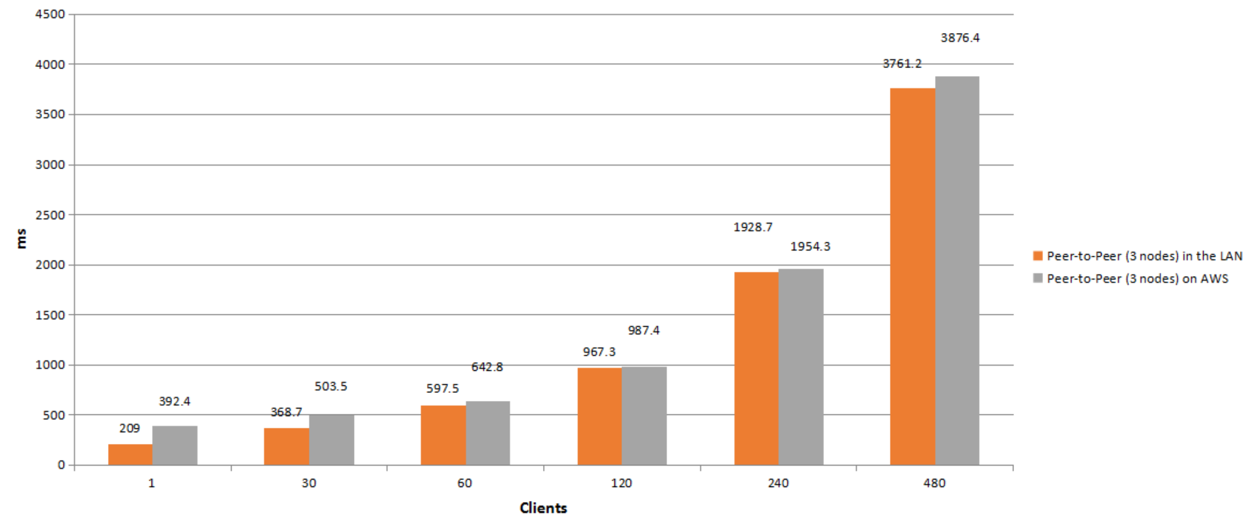


Figure 59: Average response time comparisons between LAN and AWS in consuming the E-learning resource without delay for peer-to-peer (3 nodes)

When comes to the process of consuming the E-learning resource, the trend is almost the same. The average response time ranges from 209ms to 3761.2ms in the LAN while this value ranges from 392.4ms to 3876.4ms on AWS. It is very interesting that the closest average response time appears in 120 clients number: only 20.1ms gap (967.3ms versus to 987.4ms).

In summary, it is very clear that adding servers could increase the horizontal scalability to a large extent. Meanwhile, Adding servers in the context of Cloud Services could boost the system performance dramatically. Additionally, extending the delay could have an impact on the system performance but negligible. In these experiments, the single server model represents that there is only one server in the peer-to-peer network which can approximately reflect the central server architecture as shown in Figure 3 while three nodes model stands for the peer-to-peer network as shown in Figure 4.

Since the number of E-learners is always on a large scale, horizontal scalability and performance should always be considered as priorities in designing an E-learning platform. These experiments verify the effectiveness of using Blockchain to solve the research questions in the following way:

1. How to improve the performance of access control system on a peer-to-peer E-learning platform by the decentralized solution?

Compared with the single server node in the LAN, adding servers (3 server nodes) in Multichain Blockchain could reduce the average response time a bit when the number of clients is large (more than 30 clients). However, when it comes to the cloud environment, adding servers (3 server nodes) in Multichain Blockchain could reduce the average response time dramatically when the number of clients is large. When the number of clients reaches 480, the average response time for the single server is nearly 3 times more than that for three servers on AWS (12145.8ms versus 3872.4ms for creating process and 11345.6ms versus 3876.4ms for consuming process). This demonstrates the performance is boosted with the number of clients increasing in practical term.

2. How to increase the horizontal scalability of decentralized access control systems on a peer-to-peer E-learning platform?

From the curved lines, the average response time for 3 servers is always lower than that for the single server in the same condition when the number of clients exceeds 30. This further demonstrates that horizontal scalability in a decentralized access control system could be increased by adding extra servers in a peer-to-peer E-learning platform.

CHAPTER 5
Conclusion and Future Work

5.1 Conclusion

In brief, Blockchain is a groundbreaking and ingenious method to secure, audit and track important files. Blockchain3.0 has been applied to various kinds of fields including E-learning industry. E-learning platform becomes increasingly social and collaborative in which different E-learners manage their own E-learning resources independently, simultaneously, some E-learning resources should be shared within the platform. In this situation, peer-to-peer network model plays its full role. Not until the appearance of bitcoin e-cash system has the problem of Byzantine General problem been solved elegantly in a peer-to-peer network. POW algorithm, mining mechanism, smart contracts as well as block transactions make up bitcoin system. Even though its legitimacy is controversial, it indeed brings an ingenious distributed consensus mechanism in a peer-to-peer network. Based on the bitcoin system, Blockchain technology was born. There are numerous benefits of such technology:

1) Decentralization: there is no need to introduce a trusted third-party intermediary, instead, a distributed consensus mechanism will be applied.
2) Transparency: everyone can see what happens in the Blockchain which allows the system to be more transparent and trustworthy.
3) Immutability: it becomes almost impossible(except for 51% attacks) to modify the transaction in the Blockchain.
4) Highly-secure: all transactions recorded on the Blockchain are encoded and provide integrity validation.
5) Efficiency: in the area of finance, Blockchain makes the process of transactions simple and fast.

Besides, decentralized access control in a peer-to-peer E-learning platform provides strict protection for sensitive E-learning resources. Building on top of Multichain Blockchain platform, decentralized access control becomes possible since the access control JSON file is stored separately in each node and maintained by the distributed consensus mechanism in a secure way. Furthermore, the simplification of JSON format makes the access control file straightforward and easy to understand. Meanwhile, horizontal scalability in a decentralized access control system could be increased by adding extra servers in a peer-to-peer E-learning platform.

In my proposed E-learning platform, a bunch of technologies have been used. Firstly, Beego framework, a Golang web API framework serves as the middleware between local database and Multichain Blockchain platform. It is worth noting that Golang which is a newer programming language is becoming more and more powerful especially in the field of distributed and concurrent computing. Beego framework is an easy-to-use framework and provides lots of convenient interfaces for developers. Secondly, Java programming language is used to create and extract streams API from Multichain Blockchain platform. Meanwhile, it is also used as a tool to develop Android Application (Lesson Basket). Last but not least, various front-end programming languages are employed to generate a nice-looking GUI for E-learners.

5.2 Contributions

Previous research focuses more on the centralized implementation of E-learning platform and there are indeed numerous successful and widely-used E-learning platforms such as ANGEL, Blackboard, Moodle, Canvas and etc. However, as explained in Chapter 1, the lacks of the centralized server could become even more serious in the context of E-learning platform. My research concerns building an E-learning platform on a peer-to-peer server-less model. Besides, the domain of decentralized access control model in a peer-to-peer E-learning platform is still in its initial stage where the related research is scarce. Moreover, it is the age of Blockchain3.0, researchers show countless enthusiasm to apply it to different area. My research combines Blockchain into the E-learning industry and further proposes an innovative method of building E-learning platform on a peer-to-peer network. Last but not least, Golang language, the new age of programming language equipped with the high performance of distributed computing, is selected as a tool to build the platform.

5.3 Limitation and Future work

When doing my research work, the Blockchain is still in its development period. Thus, there are inevitable insufficient existing in my proposed E-learning platform. There are reasons to believe such deficiencies could be overcome with the improvement of Blockchain technology. In terms of future work, the following functions could be considered to add:

1) **More server nodes:** More server nodes could be added in the future to test the performance and horizontal scalability.
2) **Mining mechanisms:** More mining mechanisms can be explored in the future to avoid the energy wasting issue.
3) **Role-based access control**: RBAC can be binded here to get more fine-grained access control.
4) **Social computing:** E-learning platform is becoming more and more social and collaborative. Therefore, the theory of social computing could be applied into this field.
5) **Smart contracts:** Smart contracts are an effective and powerful mechanism to enforce an agreement between different parties. In an E-learning platform, such kind of enforcement could be added among peers.
6) **Applications of other Blockchain platforms:** Multichain is not the only Blockchain platform. Other Blockchain platforms such as Etherium, BigChainDB and Eris should also be involved to evaluate the efficiency.

REFERENCES

[1] Dagger, D., O'Connor, A., Lawless, S., Walsh, E., & Wade, V. P. (2007). Service-oriented e-learning platforms: From monolithic systems to flexible services. IEEE Internet Computing, 11(3).

[2] Trends, E. L. M. (2017). Forecast 2017-2021 Report. A report by Docebo. Docebo.

[3] M. Derawi, (2014). "Securing e-learning platforms," International Conference on Web and Open Access to Learning (ICWOAL), Dubai, 2014, pp. 1-4.

[4] Sandhu, R. S., & Samarati, P. (1994). Access control: principle and practice. IEEE communications magazine, 32(9), 40-48.

[5] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

[6] Balasubramanian, M., Bhatnagar, A., Chaturvedi, N., Chowdhury, A. D., & Ganesh, A. (2007, March). A framework for decentralized access control. In Proceedings of the 2nd ACM symposium on Information, computer and communications security (pp. 93-104).

[7] Miltchev, S., Smith, J. M., Prevelakis, V., Keromytis, A., & Ioannidis, S. (2008). Decentralized access control in distributed file systems. ACM Computing Surveys (CSUR), 40(3), 10.

[8] Ruj, S., Stojmenovic, M., & Nayak, A. (2014). Decentralized access control with anonymous authentication of data stored in clouds. IEEE transactions on parallel and distributed systems, 25(2), 384-394.

[9] Ouaddah, A., Mousannif, H., Elkalam, A. A., & Ouahman, A. A. (2017). Access control in the Internet of Things: Big challenges and new opportunities. Computer Networks, 112, 237-262.

[10] Han, W., Xu, M., Zhao, W., & Li, G. (2010). A trusted decentralized access control framework for the client/server architecture. Journal of Network and Computer Applications, 33(2), 76-83

[11] Asmaa, K., & Najib, E. (2015, December). Towards a secure access control model for E-learning platform based on multi agent systems. In Intelligent Systems Design and Applications (ISDA), 2015 15th International Conference on (pp. 307-312). IEEE.

[12] Ouadoud, M., Chkouri, M. Y., Nejjari, A., & El Kadiri, K. E. (2016, October). Studying and comparing the free e-learning platforms. In Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on (pp. 581-586). IEEE.

[13] Costinela-Luminia, C. D., & Nicoleta-Magdalena, C. I. (2012). E-learning security vulnerabilities. Procedia-Social and Behavioral Sciences, 46, 2297-2301.

[14] Kang, B. H., & Kim, H. (2015). Proposal: a design of e-learning user authentication system. International Journal of Security and Its Applications, 9(1), 45-50.

[15] Asha, S., & Chellappan, C. (2008, April). Authentication of e-learners using multimodal biometric technology. In Biometrics and Security Technologies, 2008. ISBAST 2008. International Symposium on (pp. 1-6). IEEE.

[16] Luminita, D. C. (2011). Information security in E-learning Platforms. Procedia-Social and Behavioral Sciences, 15, 2689-2693.

[17] Baltzan, P. (2012). Business driven technology. McGraw-Hill/Irwin.

[18] Aïmeur, E., Hage, H., & Onana, F. S. M. (2008, January). Anonymous credentials for privacy-preserving e-learning. In E-Technologies, 2008 International MCETECH Conference on (pp. 70-80). IEEE.

[19] Edtech, (2017). "LMS Data-Spring 2017 Updates," 2017. [Online]. Available: http://edutechnica.com/tag/market-share/.

[20] Dobre, I., (2010) Critical Study of the present e-learning systems, Academia Romana, Romania, (Chapter 2).

[21] Lucian C. (2017). "Flaws in Moodle CMS put thousands of E-learning websites at risk,". [Online]. Available: https://www.pcworld.idg.com.au/article/616374/flaws-moodle-cms-put-thousands-e-learning-websites-risk/.

[22] Ahmad, S., & Bokhari, M. U. (2012). A new approach to multi agent based architecture for secure and effective e-learning. International Journal of Computer Applications, 46(22), 26-29.

[23] International Organization for Standardization. (1989). Information processing system Open Systems Interconnection- Basic Reference Model- Part 2:Security Architecture Retrived from https://www.iso.org/standard/14256.html

[24] Evered, M., & Bögeholz, S. (2004, January). A case study in access control requirements for a health information system. In Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation-Volume 32 (pp. 53-61). Australian Computer Society, Inc.

[25] Downs, D. D., Rub, J. R., Kung, K. C., & Jordan, C. S. (1985, April). Issues in discretionary access control. In Security and Privacy, 1985 IEEE Symposium on (pp. 208-208). IEEE.

[26] Li, N. (2011). Discretionary access control. In Encyclopedia of Cryptography and Security (pp. 353-356). Springer US.

[27] McCollum, C. J., Messing, J. R., & Notargiacomo, L. (1990, May). Beyond the pale of MAC and DAC-defining new forms of access control. In Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on (pp. 190-200). IEEE.

[28] Schroeder, M. D. (1972). Cooperation of mutually suspicious subsystems in a computer utility (No. MAC-TR-104). MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC.

[29] Bell, D. E., & La Padula, L. J. (1976). Secure computer system: Unified exposition and multics interpretation (No. MTR-2997-REV-1). MITRE CORP BEDFORD MA.

[30] Osborn, S. (1997, November). Mandatory access control and role-based access control revisited. In Proceedings of the second ACM workshop on Role-based access control (pp. 31-40). ACM.

[31] Sandhu, R. S. (1993). Lattice-based access control models. Computer, 26(11), 9-19.

[32] Qian, X., & Lunt, T. F. (1996). A MAC policy framework for multilevel relational databases. IEEE Transactions on Knowledge and Data Engineering, 8(1), 3-15.

[33] Brewer, D. F., & Nash, M. J. (1989, May). The chinese wall security policy. In Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on (pp. 206-214). IEEE.

[34] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-based access control models. Computer, 29(2), 38-47.

[35] Osborn, S., Sandhu, R., & Munawer, Q. (2000). Configuring role-based access control to enforce mandatory and discretionary access control policies. ACM Transactions on Information and System Security (TISSEC), 3(2), 85-106.

[36] Sandhu, R., Ferraiolo, D., & Kuhn, R. (2000, July). The NIST model for role-based access control: towards a unified standard. In ACM workshop on Role-based access control (Vol. 2000, pp. 1-11).

[37] Zhiyong Z. & Kefeng F. (2013, Mar). Digital Rights Management and Security Technology(pp.73-81)

[38] Sandhu, R., & Zhang, X. (2005, June). Peer-to-peer access control architecture using trusted computing technology. In Proceedings of the tenth ACM symposium on Access control models and technologies (pp. 147-158). ACM.

[39] Ziqing S., Jinlan L., & Xiaohua T. (2012). Authorization technology based on OAuth2.0. Applications of Computer System, (3), 260-264.

[40] Hammer-Lahav, E. (2010). The oauth 1.0 protocol.

[41] Noureddine, M., & Bashroush, R. (2011, September). A provisioning model towards Oauth 2.0 performance optimization. In Cybernetic Intelligent Systems (CIS), 2011 IEEE 10th International Conference on (pp. 76-80). IEEE.

[42] Yao L. (2017). Third-party authorization framework based on Spring and OAuth2.0. Computer Technology and Evolution, 27(3), 167-170.

[43] Tingting W., & Songze Z.. (2017). Research on Safety Authorization Grant Model based on OAuth2.0 Software Engineer, 20(1), 55-59.

[44] Hammer-Lahav, E., Recordon, D., & Hardt, D. (2011). The OAuth 2.0 authorization protocol. Network Working Group Internet-Draft.

[45] Nakamoto, S. (2009). Bitcoin open source implementation of P2P currency. P2P Foundation, 18.

[46] Linden, A. (2016). Gartner’s 2016 Hype Cycle for Emerging Technologies. Cited from http://www.gartner.com/newsroom/id/3412017 on August 16,2016.

[47] Imran, B. (2017 March). Mastering Blockchain(pp.16-39). " O'Reilly Media, Inc.".

[48] Tanenbaum, A. S., & Van Steen, M. (2007). Distributed systems: principles and paradigms. Prentice-Hall.

[49] Fox, G. (2001). Peer-to-peer networks. Computing in Science & Engineering, 3(3), 75-77.

[50] Czirkos, Z., & Hosszú, G. (2008). Peer-to-Peer Methods for Operating System Security. In Encyclopedia of Networked and Virtual Organizations (pp. 1185-1191). IGI Global.

[51] Brewer, E. A. (2000, July). Towards robust distributed systems. In PODC (Vol. 7).

[52] Liu, J., Wang, S., Zhou, A., Kumar, S., Yang, F., & Buyya, R. (2017). Using proactive fault-tolerance approach to enhance cloud service reliability. IEEE Transactions on Cloud Computing.

[53] Brewer, E. (2012). CAP twelve years later: How the" rules" have changed. Computer, 45(2), 23-29.

[54] Gilbert, S., & Lynch, N. (2002). Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. Acm Sigact News, 33(2), 51-59.

[55] Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS), 4(3), 382-401.

[56] Castro, M., & Liskov, B. (1999, February). Practical Byzantine fault tolerance. In OSDI (Vol. 99, pp. 173-186).

[57] Tschorsch, F., & Scheuermann, B. (2015). Bitcoin and beyond: A technical survey on decentralized digital currencies. IEEE Communications Surveys & Tutorials, 18(3), 2084-2123.

[58] Chaum, D. (1983). Blind signatures for untraceable payments. In Advances in cryptology (pp. 199-203). Springer US.

[59] W. Dai. (1998). B-Money [Online]. Available: http://www.weidai.com/bmoney.txt

[60] Vishnumurthy, V., Chandrakumar, S., & Sirer, E. G. (2003, June). Karma: A secure economic framework for peer-to-peer resource sharing. In Workshop on Economics of Peer-to-Peer Systems (Vol. 35).

[61] H. Finney. (2004). Rpow [Online]. Available: http://cryptome.org/rpow.html

[62] N. Szabo. (2005). Bit Gold [Online]. Available: http://unenumerated.blogspot.de/2005/12/bit-gold.html

[63] Malkhi, D., & Reiter, M. (1997, May). Byzantine quorum systems. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing (pp. 569-578). ACM.

[64] Douceur, J. R. (2002, March). The sybil attack. In International Workshop on Peer-to-Peer Systems (pp. 251-260). Springer Berlin Heidelberg.

[65] Bos, J. W., Halderman, J. A., Heninger, N., Moore, J., Naehrig, M., & Wustrow, E. (2014, March). Elliptic curve cryptography in practice. In International Conference on Financial Cryptography and Data Security (pp. 157-175). Springer Berlin Heidelberg.

[66] Ron, D., & Shamir, A. (2013, April). Quantitative analysis of the full bitcoin transaction graph. In International Conference on Financial Cryptography and Data Security (pp. 6-24). Springer Berlin Heidelberg.

[67] O'Dair, M., Beaven, Z., Neilson, D., Osborne, R., & Pacifico, P. (2016). Music on the Blockchain.

[68] D. Eastlake III and T. Hansen. (2011, May). US Secure Hash Algorithms (SHA and SHA-Based HMAC and HKDF), RFC 6234 (Informational), Internet Engineering Task Force [Online]. Available: http://www.ietf.org/rfc/rfc6234.txt

[69] Turek, J., & Shasha, D. (1992). The many faces of consensus in distributed systems. Computer, 25(6), 8-17.

[70] King, S., & Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. self-published paper, August, 19.

[71] Bastiaan, M. (2015). Preventing the 51%-attack: a stochastic analysis of two phase proof of work in bitcoin. In Availab le at http://referaat. cs.utwente.nl/conference/22/paper/7473/preventingthe-51-attack-a-stochasticanalysis-of-two-phase-proof-of-work-in-bitcoin. pdf.

[72] Swan, M. (2015). Blockchain: Blueprint for a new economy. " O'Reilly Media, Inc.".

[73] Szabo, N. (1994). Smart contracts. Unpublished manuscript.

[74] Peters, G. W., & Panayi, E. (2016). Understanding modern banking ledgers through Blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In Banking Beyond Banks and Money (pp. 239-278). Springer International Publishing.

[75] Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016, May). Hawk: The Blockchain model of cryptography and privacy-preserving smart contracts. In Security and Privacy (SP), 2016 IEEE Symposium on (pp. 839-858). IEEE.

[76] Buterin, V. (2014). A next-generation smart contract and decentralized application platform. white paper.

[77] Amit. (2015, Aug). Public and Private Blockchain Concepts and Examples, Let's Talk Payments [Online]. Available: https://letstalkpayments.com/public-and-private-

Blockchain-concepts-and-examples

[78] Pree, W. (2016). Blockchain: Technology and Applications.

[79] Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, 151.

[80] Vasin, P. (2014). Blackcoin’s proof-of-stake protocol v2.

[81] Greenspan, G. (2015). Multichain Private Blockchain—White Paper.

[82] Zheng, Z., Xie, S., Dai, H. N., & Wang, H. (2016). Blockchain Challenges and Opportunities: A Survey. Work Pap.

[83] NRI (2015). Survey on Blockchain Technologies and Related Services. Tech. Rep.

[84] Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016, May). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In Security and Privacy (SP), 2016 IEEE Symposium on (pp. 839-858). IEEE.

[85] Bruce, JD (2014). The mini-Blockchain scheme. Tech. Rep.

[86] Chepurnoy, A., Larangeira, M., & Ojiganov, A. (2016). A Prunable Blockchain Consensus Protocol Based on Non-Interactive Proofs of Past States Retrievability. arXiv preprint arXiv, 1603.

[87] Kraft, Daniel (2016). “Difficulty control for Blockchain-based consensus systems”. In: Peerto-Peer Networking and Applications 9.2, pp. 397–413.

[88] Fielding, R. T., & Taylor, R. N. (2000). Architectural styles and the design of network-based software architectures (p. 151). Doctoral dissertation: University of California, Irvine.

[89] Zhang, L., Yu, S., Ding, X., & Wang, X. (2014, August). Research on IOT RESTful web service asynchronous composition based on BPEL. In Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on (Vol. 1, pp. 62-65). IEEE.

[90] Egham, U.K. (2017). Gartner Says Worldwide Sales of Smartphones Grew 9 Percent in First Quarter of 2017. Cited from http://www.gartner.com/newsroom/id/3725117