VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi, Karnataka – 590018

A

Technical Seminar Report

on

"Fabric-iot: A Blockchain-Based Access Control System in IoT"

Submitted by

SHUBHASHINI PAL (1KT17IS019)

Under the guidance of

Mrs. SHRUTI B.P

Assistant Professor

Department of ISE

SRI KRISHNA INSTITUTE OF TECHNOLOGY

Department of Information Science and Engineering

No.29, Hesaraghatta Main Road, Chimney hills, Chikkabanavara P.O., Bengaluru – 560090

2020-2021

SRI KRISHNA INSTITUTE OF TECHNOLOGY

No.29, Hesaraghatta Main Road, Chimney hills, Chikkabanavara P.O., Bengaluru – 560090

Department of Information Science and Engineering

CERTIFICATE

Certified that the Technical Seminar work entitled **"Fabric-iot: A Blockchain-Based Access Control System in IoT"** carried out by **SHUBHASHINI PAL (1KT17IS019)** a bonafide student of **Sri Krishna Institute of Technology**, Bengaluru in partial fulfillment for the award of **Bachelor of Engineering in Information Science and Engineering** of the **Visvesvaraya Technological University**, Belagavi during the year 2020-21. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements in respect of Technical Seminar work prescribed for the said Degree.

Signature of the Guide	**Signature of the HOD**
Mrs. Shruti B.P	**Dr. Hemalatha K. L.**
Assistant Professor	**Professor and HOD**
Dept. of ISE, SKIT	**Dept. of ISE, SKIT**

DECLARATION

I, **SHUBHASHINI PAL,** student of VIII semester in Information Science and Engineering, Sri Krishna Institute Technology, Bengaluru, hereby declare that the Technical Seminar work entitled **"Fabric-iot: A Blockchain-Based Access Control System in IoT"** has been carried out by me under the supervision of my guide **Mrs. Shruthi B.P., Assistant Professor, Dept. of Information Science and Engineering, Sri Krishna Institute of Technology, Bengaluru** and submitted in partial fulfillment for the award of degree in **Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University**, Belagavi during the academic year 2020-2021. I further declare that the Technical Seminar report has not been submitted to any other University for the award of any other degree.

Place: Bengaluru **SHUBHASHINI PAL (1KT17IS019)**

Date:

ABSTRACT

IoT devices have some special characteristics, such as mobility, limited performance, and distributed deployment, which makes it difficult for traditional centralized access control methods to support access control in current large-scale IoT environment

To address these challenges, this paper proposes an access control system in IoT named fabric-iot, which is based on Hyperledger Fabric blockchain frame work and attributed based access control (ABAC). The system contains three kinds of smart contracts, which are Device Contract (DC), Policy Contract (PC), and Access Contract (AC). DC provides a method to store the URL of resource data produced by devices, and a method to query it. PC provides functions to manage ABAC policies for admin users.AC is the core program to implement an access control method for normal user Combined with ABAC and blockchain technology, fabric-iot can provide decentralized, fine-grained and dynamic access control management in IoT. The results show that fabric-iot can maintain high throughput in largescale request environment and reach consensus efficiently in a distributed system to ensure data consistency.

ACKNOWLEDGEMENT

The completion of Technical Seminar work brings with a sense of satisfaction, but it is never complete without thanking the persons responsible for its successful completion.

At the outset, I express my most sincere grateful acknowledgment to the holy sanctum **"Sri Krishna Institute of Technology",** the temple of learning, for giving me an opportunity to pursue the degree course in Information Science and Engineering and thus helping me in shaping the career.

I extend my deep sense of sincere gratitude to **Dr. Manjunatha A., Principal**, Sri Krishna Institute of Technology**,** Bengaluru, for providing me an opportunity to continue my higher studies.

I extend my special in-depth, sincere gratitude to my guide **Mrs. Shruti B.P., Assistant Professor, Department of Information Science and Engineering,** Sri Krishna Institute of Technology, Bengaluru for her constant support and valuable guidance for completion of the Technical Seminar work.

I extend my sincere gratitude to **Dr. Hemalatha K.L.**, **Professor and HOD, Department of Information Science and Engineering,** Sri Krishna Institute of Technology, Bengaluru for her constant support.

I would like to thank my Technical Seminar Coordinator **Mrs. Sandhya B.R., Assistant Professor, Department of Information Science and Engineering**, Sri Krishna Institute of Technology, Bengaluru, for her support.

I would like to thank all the teaching and non-teaching staff members in my **Department of Information Science and Engineering,** Sri Krishna Institute of Technology**,** Bengaluru, for their support.

Finally, I would like to thank all my friends and family members for their constant support, guidance and encouragement.

SHUBHASHINI PAL (1KT17IS019)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction to Fabric-iot

Fabric-iot: A Blockchain-Based Access Control System in IoT, In the era of Industry, in modern era with the developments of Internet and computer hardware, more and more devices are connected with each other through wireless network, making the scale of Internet of Things (IoT) larger and larger.

IoT is a distributed network composed of a large number of sensors and gateways. IoT devices interact with the environment all the time, producing different types of data resources, such as image, audio, video, digital signal, etc.

The resources produced by IoT devices often contain privacy and sensitive data, so there will be serious consequences when they are obtained illegally. The access control technology is an important means to protect resources, which has been widely used in various systems and environments.

Traditional access control methods include discretionary access control (DAC), identity-based access control (IBAC), and mandatory access control (MAC), etc. But these methods are all centralized designs, which have the disadvantages of single-point failure, difficult to expand, low reliability and low throughput. IoT devices may belong to different organizations or users, and probably have mobility and limited performance, which make centralized access control difficult to meet the requirements of access control in IoT environment.

Attributed based access control (ABAC) is a logical access control model, which controls the access between sub- jects and objects, according to the attributes of entries, operations and related environments. ABAC firstly extracts the attributes of user (subject), resource (object), permission and environment respectively, then combines the relationship of these attributes flexibly, and finally transforms the management of permission into the management of attribute, providing a fine-grained and dynamic access management method.

Blockchain is another kind of emerging data management technology, it ensures the reliability of data through distributed storage, and the blocks are linked by hash algorithm as a chain to ensure the integrity of the data It synchronizes data between nodes through P2P network and consensus algorithm, ensuring data consistency. Hyperledger Fabric is an open- source blockchain development platform, which not only has the characteristics of blockchain such as decentralized ledger, immutable, and group consensus, but also provides more efficient consensus mechanisms, higher throughputs, smart contracts, and support for multiple organizations and ledgers. In this project, applying blockchain technology to IoT access control, design and implement an access control system named fabric-iot, which is based on Hyperledger Fabric and ABAC. By using distributed architecture, fabric-iot can trace records, provide dynamic access control management and solve the access control problem in IoT.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Systems

[1] Fabric-iot: A Blockchain-Based Access Control System in IoT

IoT devices have some special characteristics, such as mobility, limited performance, and distributed deployment, which makes it difficult for traditional centralized access control methods to support access control in current large-scale IoT environment. To address these challenges, this paper proposes an access control system in IoT named fabric-iot, which is based on Hyperledger Fabric blockchain framework and attributed based access control (ABAC). The system contains three kinds of smart contracts, which are Device Contract (DC), Policy Contract (PC), and Access Contract (AC). DC provides a method to store the URL of resource data produced by devices, and a method to query it. PC provides functions to manage ABAC policies for admin users. AC is the core program to implement an access control method for normal users. Combined with ABAC and blockchain technology, fabric-iot can provide decentralized, fine-grained and dynamic access control management in IoT. The results show that fabric-iot can maintain high throughput in large-scale request environment and reach consensus efficiently in a distributed system to ensure data consistency.

Disadvantage:

1. The concern here is to extended iot control.

[2] An Empirical Study on System Level Aspects of Internet of Things (IoT)

Internet of Things (IoT) is an integration of the Sensor, Embedded, Computing, and Communication technologies. The purpose of the IoT is to provide seamless services to anything, any time at any place**.** IoT technologies play a crucial role everywhere, which brings the fourth revolution of disruptive technologies after the internet and Information and Communication Technology (ICT). Addressing the predominant system-level design aspects like energy efficiency, robustness, scalability, interoperability, and security issues result in the use of a potential IoT system.

this paper presents the current state of art of the functional pillars of IoT and its emerging applications to motivate academicians and researches to develop real-time, energy efficient, scalable, reliable, and secure IoT applications. This paper summarizes the architecture of IoT, with the contemporary status of IoT architectures. Highlights of the IoT system-level issues to develop more advanced real-time IoT applications have been discussed. Millions of devices exchange information using different communication standards, and interoperability between them is a significant issue. This paper provides the current status of the communication standards and application layer protocols used in IoT with the detailed analysis. The computing paradigms like Cloud, Cloudlet, Fog, and Edge computing facilitate IoT with various services like data offloading, resource and device management, etc.

Disadvantage:

1. There aspect levels were not satisfactory.

[3] IoT-RTP and IoT-RTCP: Adaptive Protocols for Multimedia Transmission Over Internet of Things Environments.

Recently, the Internet of Things (IoT) has attracted the interest of network researchers all over the world. Multimedia transmission through IoT presents an important challenge owing to nodes diversity. In this paper, adaptive versions of the real-time transport protocol (RTP) and real-time control protocol (RTCP), i.e., IoT-RTP and IoT-RTCP, are proposed. In these versions, the nature of IoT environments, such as transmission channels heterogeneity, sudden change in session size, and different multimedia sources, is considered. The basic idea of the proposed adaptive versions is to divide the large multimedia sessions into simple sessions with awareness of network status. To achieve this target, additional fields are added to the RTP and RTCP headers. These fields work under certain conditions to decrease the network overload. Finally, to test the performance of the proposed IoT-RTP and IoT-RTCP, as environment is constructed using the network simulation package (NS2). The results of intensive simulations proved that the proposed adaptive versions of the multimedia protocols outperform the basic ones in terms of end-to-end delay, delay jitter, number of receiver reports, packet loss, throughput, and energy consumption.

Disadvantage:

1. The RTP was not able to carry out autonomous control.

[4] IoT-Assisted ECG Monitoring Framework With Secure Data Transmission for Health Care Applications

In The emerging Internet of Things (IoT) framework allows us to design small devices that are capable of sensing, processing and communicating, allowing sensors, embedding devices and other things' to be created which will help to understand the surroundings. In this paper, the IoT assisted lector diagram (ECG)monitoring frame work with secure data transmission has been proposed for continuous cardiovascular health monitoring. The development and implementation of a lightweight ECG Signal Strength Analysis has been proposed for automatic classification and real time implementation, using ECG sensors, Arduino, Android phones, Bluetooth and cloud servers with the proposed IoT-assisted ECG monitoring system. For secure data transmission, the Lightweight Secure IoT (LS-IoT) and Lightweight Access Control (LAC) has been proposed. The ECG signals taken from the MIT-BIH and Physio Net Challenges databases and ECG signals for various physical activities are analyzed and checked in real-time. The proposed IoT assisted ECG monitoring framework has great potential to determine the clinical acceptance of ECG signals to improve the efficiency, accuracy and reliability of an unsupervised diagnostic system. Medical care services have been one of the most significant problems for both people and governments with a rapid increase in human populations and preserve usage. Whereas, the problems of the maturing population are more real, according to a study from the World Health Organization (WHO). The health status of elderly people must be checked more often, a more prominent test of current medicinal frameworks. Careful consideration must be given to identifying human diseases in a comfortable and precise way at alecost. Because of the growth, experience and expertise gained over the years in cardiac analysis.

Disadvantages:

1.The System needs to be improved.

2. Operational speed is less.

[5] Smart Urban Living: Enabling Emotion-Guided Interaction with Next Generation Sensing Fabric

With the rapid development of technologies, such as 5G, cloud computing, and artificial intelligence, smart cities are gradually being implemented from a concept to actual development and deployment.

A smart city integrates the management of multiple internets of things (IoT) terminals to improve urban development and to provide people with a better way of life. As a mobile and flexible terminal form of IoT, sensing fabric has been widely used in smart cities. In this article, a new generation of sensing fabric is combined with the smart city, and an emotion-guided interaction architecture of smart urban living is proposed. However, there is a lot of information redundancy in the spatiotemporal environment that is collected by the various sensing fabrics in a smart city. Therefore, an energy consumption optimization model for sensing fabrics based on the information redundancy in the spatiotemporal environment is also put forward. By recognizing the current data collection ability and residual energy of sensing fabric, an optimal data allocation strategy is obtained to further improve the user experience of the smart urban system. Simulation experiments were carried out to verify the effectiveness of the proposed model.

Disadvantage:

1. The accuracy is less due to data attack and needs to be improved.

2.2 Proposed System

The proposed system's iot process is designed to overcome the limitations of the above-mentioned systems. The research provides new insights into autonomous blockchain based iot system. design and manufacture and into possible ways to increase the efficiency which shortens the time, and define a device resource sharing model according to the data production of the IoT devices in real life. The model makes the data resources generated by the device correspond to the URL one by one, greatly simplifying the sharing mode and storage structure of the device resources. And propose a blockchain-based access control system for IoT named fabric-iot and describe its workflow and architecture in detail. The system uses distributed architecture to separate users and devices, and implemented the dynamic management of permissions to support efficient access. And we design three kinds of smart contracts based on the Hyperledger Fabric platform. The first one implements the ABAC model. The second one implements the ABAC policy management. The last one implements the device resource management. and we introduce the network initialization, chain code installation, and smart contract invoking of fabric-iot in detail. And we design two groups of comparative experiments to verify the system performance and consensus speed. The remaining of this paper is organized as follows. Section II introduces the related work. Section III details the structure and operating mechanism of Hyperledger Fabric, as well as ABAC model. The design of resource model, policy model, system's structure, workflow, and the implements of smart contract are presented in Section IV. In Section V, we demonstrate how to setup fabric-iot system and how to work with it to manage the access control of the IoT resources. We set two groups of comparative experiments, and then analysis the results. In last section we make a conclusion for this paper and give a preview for further work.

CHAPTER 3
OBJECTIVES

3.1 The Objectives of Fabric-iot System:

The main objective of the proposed research is to overcome the disadvantages of the existing systems and to also enhance the previous existing systems and to provide an overall brief strategy considering from design of ABAC to its working. It also aims at providing an evaluation method to measure the efficiency of the work of these IOT.

1) To define a device resource sharing model according to the data production of the IoT devices in real life. The model makes the data resources generated by the device correspond to the URL one by one, greatly simplifying the sharing mode and storage structure of the device resources.

2) To propose a blockchain-based access control system for IoT named fabric-iot and describe its workflow and architecture in detail. The system uses distributed architecture to separate users and devices, and implemented the dynamic management of permissions to support efficient access.

3) To design three kinds of smart contracts based on the Hyperledger Fabric platform. The first one implements the ABAC model. The second one implements the ABAC policy management. The last one implements the device resource management.

4) To introduce the network initialization, chain code installation, and smart contract invoking of fabric-iot in detail.

5) To design two groups of comparative experiments to verify the system performance and consensus speed.

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE

4.1 Architecture Of Fabric-iot

The overall methodology and the architecture of the Fabric-iot: A Blockchain-Based Access Control System in IoT, in brief has been consolidated and is shown in the Figure 4.1. Fabric-iot, a blockchain-based access control system for IoT, consists of four parts: users, blockchain, smart gateway, and devices

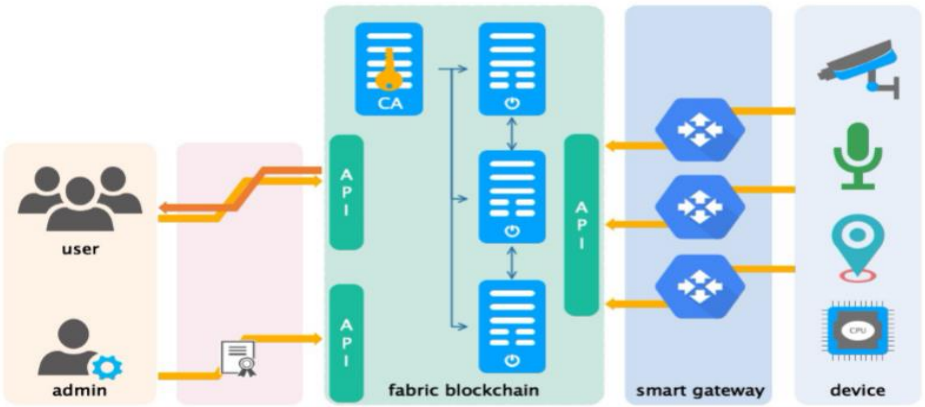

Figure 4.1: Architecture Of Fabric-iot.

The conceptual framework of Fabric-iot: A Blockchain-Based Access Control System in IoT can be summarized in four steps:

Step 1: USERS This system divides users into two types: admin, and common user. The admin is responsible for managing the blockchain system and maintaining the program of smart gateway.

The admin needs to provide a certificate to access the blockchain system. The specific operation allowed are as follows.

i. Add new smart contracts. Admin can deploy new smart contracts on the blockchain through API.

ii. Upgrade contracts. The admin can upload a new smart contract to nodes and install it, while the old one will be upgraded to a new version. The common user, which means the owner of the device, gets the resource URL by sending attributed based authorization request to the blockchain system.

2) BLOCKCHAIN It is the core of system. All nodes need to obtain CA authentication before joining the blockchain system. The blockchain is developed based on Hyper ledger Fabric

which implements access control by smart contracts.

Block chain system exposes API for users and smart gateway to access. It mainly implements three functions as follows.

i. Device resource URL data storage.

ii. Attribute-based user rights management.

iii. Authentication of user access to resources.

3) SMART GATEWAY As the bridge between devices and blockchain system, it can receive the message from the device and put the URL it contained to block chain, avoiding the pressure on blockchain system caused by direct access of devices.

4) IoT DEVICES As the largest group, the IoT devices generally do not have strong computing ability, enough storage, and durable battery. Therefore, it is impossible to deploy IoT devices as peer nodes of blockchain directly. IoT devices has a unique MAC address or product ID, which can be distinguished from other devices. Generally, the devices may belong to both some users or groups. Whenever a new resource is generated by the device, a message contains the URL of resource is sent to the smart gateway. In this system, MQTT protocol is used as the message transmission protocol.

4.2 Connections between Users and Resources

Associated with Iot and Block chain:

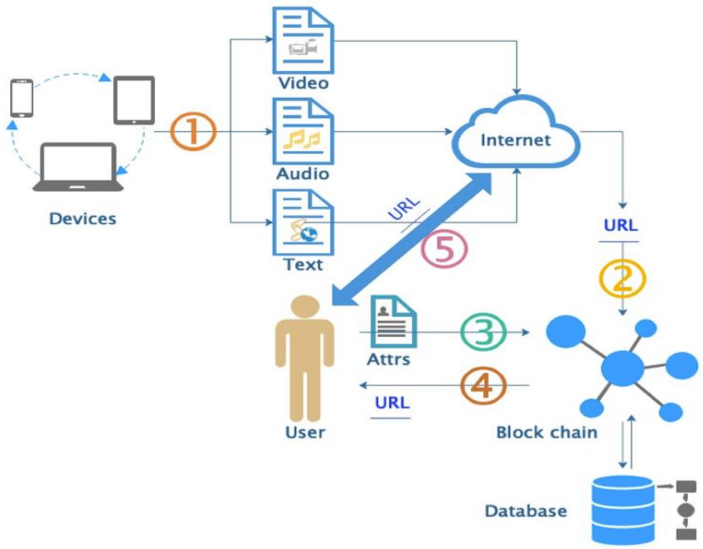

Figure 4.2: Connections between Users and Resources.

The connection between device resources and users is shown in Fig.2.The brief work flow is shown in the1-5 serial number in the figure.

1) The device distributes the resource to the Internet and generates the resource URL

2) The device saves the resource URL to the blockchain system.

3) Users request the blockchain system by attributes to get authorization.

4) Blockchain sends URL to the authorized users.

5) Users download or pull resource data on the Internet according to the URL.

Combined with ABAC model and the characteristics of data generated by the IoT devices, the device access control policy model is defined as follows:

P ={AS,AO,AP,AE} **(Equation 4.1)**

AS ={userId,role,group} **(Equation 4.2)**

AO ={deviceId,MAC} **(Equation 4.3)**

AP =(1, allow 0, deney) **(Equation 4.4)**

AE ={createTime,endTime,allowedIP} **(Equation 4.5)**

P (Policy): It represents a policy of attributed access control. This set contains four elements: AS, AO, AP, and AE.

AS (Attribute of Subject): It represents the attributes of a subject (user) and includes three types: user ID (unique identification user), role (user role), and group (user group).

AO (Attribute of Object): It represents the attributes of an object (resource), which consist of a device ID or a MAC address of device. In this model, we do not regard the resource URL as an attribute directly. Instead, we use the unique identifications as attributes of a device. Because in reality, the network of the device is variable and the data produced by the device is dynamic. We assume that the function of a device in the system is single, and each ID or MAC can only correspond to one resource URL at one time.

AP (Attribute of Permission): It indicates whether user have access to resources. The value1stands for ‘‘allow’’ and 2 stands for ‘‘deny’’. When AP is initialized, the default value is 1. Admin can revoke access authorization according to the situation by setting the value of AP to 0.

AE (Attribute of Environment): It indicates the attributes of environment which required for access control. AE has three kinds of attributes: time, end time, and allowed IP. Time stands for the creation time of policy. End time stands for the expiration time of policy. The policy will be invalid when current time is later than end time. Allowed IP is aimed to prevent the IP address outside the network segment from accessing the system.

4.3 Work Flow of Fabric-iot

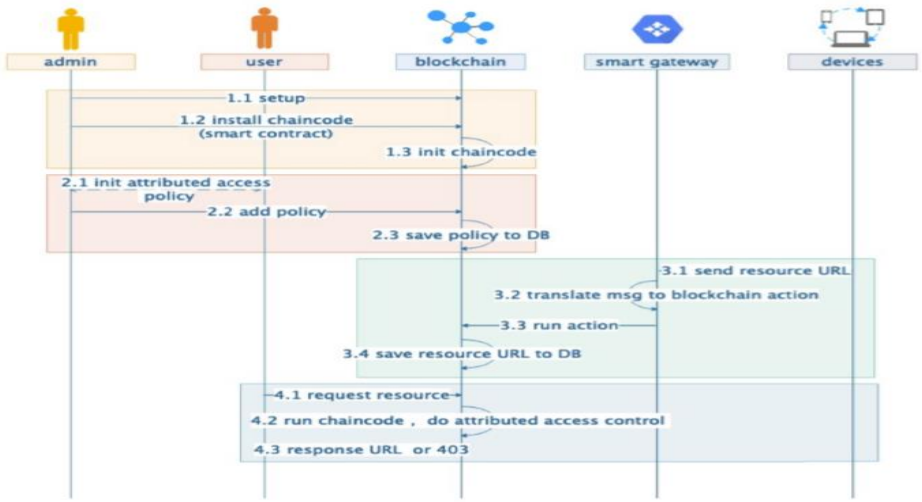

Figure 4.3: Work Flow of Fabric-iot.

Part1 the block chain network initialization and the chain code installation are the basis process of the system. These operations require the admin to work in the intranet. Flow 1 mainly including three steps.

Step 1: Before setting up a Hyperledger Fabric network, we need to create certificates for all members such as peer nodes, ordered nodes, channels, users, etc. All certificates are generated by CA.

CA→{Certpeer,Certorder,Certchannel,Certuser} **(Equation 4.6)**

The peer node and the order node run in the docker container. The certificate needs to be packaged into their docker image before running.

Build (conf ,Cert) build →Image run −→Container (Equation 4.7)

After all the peer nodes and ordered nodes are set up successfully, we start to create the channel. Every channel

joined an independent blockchain and ledger.

{blockchain,ledger} join →Channel (Equation 4.8)

Step 2: So far, a basic Hyperledger Fabric network is setup. In order to build applications, we need to design chain code. Our source code of chain code is written in Golang.

Code(F(x)...) →CC (Equation 4.9)

Admin uses Hyperledger Fabric SDK or client to install CC (chain code). All CC will be installed to peer nodes.

Install (CC) SDK/Client →Peer (Equation 4.10)

Step 3: Once the chain code is installed, it needs to be initialized. Invoke function is used to initialize chain code. Every instantiated chain code will be saved in the container as an endorsement.

Invoke (Init) SDK/Client →Peer (Equation 4.11)

Part2 Develop access control policy and save them to blockchain system. This process requires the user and admin work together to decide and customize the access policy in advance and uploads them to the blockchain system by admin.

Step1: Admins and users jointly make access policies, which are defined based on the attributes of subject (user), object (device resource), operation, and environment.

Decide (AS,AO,AE,AP)→ABACP (Equation 4.12)

Step2 : After the access policy is defined, the admin uploads it to the blockchain network.

Upload (ABACP)→contract (Equation 4.13)

Step3: Admin connects to blockchain to add, modify and delete the policy by running the Policy Contract. The value of the policy is saved in the SDB and the record of action is written into the ledger.

Policy Contract (ABACP)→{Ledger,SDB} (Equation 4.14)

Part3 Device reports resource URL to smart gateway, and after that, smart gateway uploads it to blockchain system.

Step1: The device generates a message included device ID and URL and sends it to smart gateway by MQTT.

{deviceId,URL}→Msg MQTT →SG (Equation 4.15)

Step 2: Smart gateway parses messages and generates an action for blockchain.

Translate (Msg)→BA (Equation 4.16)

Step 3: Smart gateway connects to blockchain client to run the action.

Run(BA) Cli −→DeviceContract (Equation 4.17)

Step 4 Blockchain saves the URL of device resource by invoking functions of DC.

Device Contract (deviceId,URL)→{Ledger,SDB} (Equation 4.18)

Part 4 The process of acquiring resources based on attributes is the core of system. It has three steps as follows:

Step1: User initiates attributed based requests.

userId →Request{AS∧AO∧AP} (Equation 4.19)

Step 2: Invoking the functions of AC after receiving the requests.

Access Contract(Request)→(1, OK 0, Forbidden (Equation 4.20)

Step 3: If the validation passes, blockchain system query URL by invoking the function in DC and return it to users. If it failed, 403 error (403 stands for forbidden in HTTP status codes) will be returned to users.

result =(1, DC −−→URL 0, → 403 (Equation 4.21)

4.4 Smart Contract Design

Smart contract is the core of access control implementation. There are three kinds of smart contracts in this system: Policy Contract (PC), Device Contract (DC), and Access Contract (AC).

POLICY_CONTRACT It provides the following methods to operate ABACP. Auth(): Admin defines the ABACP for users, and sends the request for adding ABACP to the blockchain system. An example of ABACPR (attributed based access control policy request) Admin encrypts the data with the public key of the PC node, and the n signs their quest with the private key. PC invokes Auth() to verify the identity of admin. with its public key and decrypts the data with self's private key. Check_Policy(): As shown in Algorithm 1. PC needs to check the validity of ABACP. A legal ABACP needs to contain the above four attributes, and the type of each attribute also needs to meet the requirements.

Algorithm 1 PolicyContract.CheckPolicy(): Check ABAC Policy Before Putting it Into DB

Input: ABACP
Output: True or False

```
1: < AS, AO, AE, AP > ← ABACP
2: IsOK = True
3: for item in AS do:
4:     if item ∉ < userId, role, group > then
5:         IsOK = False
6:     end if
7: end for
8: for item in AO do:
9:     if item ∉ < deviceId, MAC > then
10:        IsOK = False
11:    end if
12: end for
13: if Val(AE)! = 1 or0
14:    IsOK = False
15: end if
16: for item in AP do:
17:    if item ∉ < createTime, endTime, allowedIP > then
18:        IsOK = False
19:    end if
20: end for
21: return IsOK
```

Figure 4.4: Connections of PolicyContract CheckPolicy.

AddPolicy(): As shown in Algorithm 2. After ABACP is verified to be legal by CheckPolicy(), the PC invokes AddPolicy () to add ABACP to the SDB, at the same time, all the action records will be written to the ledger.

UpdatePolicy(): In some cases, admin needs to modify ABACP. The function UpdatePolicy() implements the interface of updating SDB, and the operation record of updating will also be written to the blockchain.

UpdatePolicy() is similar to AddPolicy(), which also invokes the put method of the application interface to overwrite the old value.

DeletePolicy(): ABACP has an expiration time and it can be canceled by admin. There are two cases where deletion occurs. One occurs when admin actively delete a policy by

Algorithm 2 PolicyContract.AddPolicy(): Add ABAC Policy to Blockchain

Input: ABACP
Output: Error or null

```
1: @implement SmartContract Interface
2: APIstub ChaincodeStub ← Invoke()
3: if CheckPolicy(ABACP) == False
4:     return Error('BadPolicy')
5: end if
6: Id ← Sha256(ABACP.AS + ABACP.AO)
7: err ← APIstub.PutState(Id, ABACP)
8: if err! = null then
9:     return Error(err.Text)
10: end if
11: renturn null
```

Figure 4.5: Connections of PolicyContract AddPolicy.

Invoking this function. And the other occurs when the CheckAccess() method is executing, if the attribute ‘‘endTime’’ is expired, then it will invoke this function in PC to delete the related policy. As shown in Algorithm 3.

Algorithm 3 PolicyContract.DeletePolicy(): Delete ABAC Policy From Blockchain

Input: *AS*, *AO*
Output: *Error or null*

```
1: @implement SmartContract Interface
2: APIstubChaincodeStub ← Invoke()
3: Id ← Sha256(AS + AO)
4: err ← APIstub.GetState(Id)
5: if err! = null then
6:     return Error(err.Text)
7: end if
8: APIstub.DelState(Id)
9: if err! = null then
10:    return Error(err.Text)
11: end if
12: renturn null
```

Figure 4.6: Connections of PolicyContract DeletePolicy.

QueryPolicy(): It implements the interface of database querying which provides a function to get ABACP for other chain code. We choose CouchDB as SDB. Although it is a key-value document database, CouchDB supports complex queries similar to mongo DB. In that case, QueryPolicy() supports querying ABACP by AS or AO.

2) DEVICE CONTRACT DC is mainly responsible for storing the resource URL of the device into SDB. DC has 2 input parameters {DeviceId,URL}. The functions provided are as follows. AddURL(): It uses DeviceId as a key and URL as a value to store in SDB.

GetURL(): It queries the corresponding URL value from SDB according to DeviceId.

3) ACCESS CONTRACT It verifies whether the user’s ABACR matches the ABAC policy. Like PC, the request data is signed by the user’s private key, after that, AC verifies the signature to check the user’s identity by public key of user. The methods provided are as follows.

Auth(): Similar to the PC method of the same name, it verifies the request with the user’s public key and check the authenticity of its identity.

GetAttrs(): It parses the property data field after the signature is verified. Only part of ABAC attributes are included in ABACR:{AS,AO}, while AE needs to be determined by AC.

Finally, these attributes are combined as{AS,AO,AE}. CheckAccess(): It is the core function to achieve access control management, as shown in Algorithm4. Firstly, It gets the attribute set by GetAttrs(). Secondly, it invokes the QueryPolicy() method of PC to query the corresponding ABACP according to AS and AO. If the returned result is empty, which means that there is no policy to support the request, it will return a 403 error directly (indicating that there is no permission). Thirdly, it starts to judge one by one whether the AE of request matches the AE of ABACP and whether the value of AP is 1 (stands for allow). If all attributes match the policy, the verification passes. Finally, it invokes the GetURL() function of DC to get the URL of the resource and return it to the user, otherwise 403 error will be returned.

Algorithm 4 AccessContract.CheckAccess(): Check User's Access

Input: *ABAC_Request*
Output: *URL or Error*

1: $< A_uS, A_uO, A_uE > \leftarrow GetAttrs(ABAC\_Request)$
2: $P =< P_1, P_2, \ldots P_n > \leftarrow PC.QueryPolicy(A_uS, A_uO)$
3: if $P == Null$ then
4: return $Error(403)$
5: endif
6: for P *in* $< P_1, P_2, \ldots P_n >$ do
7: $< \ldots, A_pP, A_pE > \leftarrow P$
8: if $Value(A_pP) ==$ '*deney*' then
9: *continue*
10: if $A_uE \cap A_pE ==$ then
11: *continue*
12: $URL \leftarrow DC.GetURL(A_uO)$
13: end for
14: if $URL! = Null$ then
15: return URL
16: else
17: return $Error(403)$
18: end if

Figure 4.7: Connections of PolicyContract CheckAccess.

4.4.1 Environments

Hardware	
CPU	i7 7500u 2.9GHz, i7 8700k 3.7GHz
Memory	8G,8G
Hard Disk	256G, 1T
Software	
OS	Mac OS 10.14.6, Deepin Linux 15.11
docker	v19.03.2
docker-compose	v1.24.1
node	v12.12.0
golang	v1.12.9
hyperledger fabric	v1.4.3

Figure 4.8: Hardware requirement.

The experiment was carried out on two PCs. Hardware and software are listed above.

4.4.2 System Building Process and Realization

The experiment is divided into three parts. The first part mainly introduces the structure of fabric-iot, as well as initialize configuration and start-up steps. The second part introduces the process of chain code installation. The third part describes how to implement the IoT resource access control method based on ABAC by invoking three kinds of smart contracts (PC, AC, and DC).

A. STRUCTURE AND INITIALIZATION OF FABRIC-IoT: Fabric-iot consists of eight kinds of docker nodes which are shown below. The steps of system initialization are as follows.

Node Name	Description	Number
fabric-iot/couchdb	database node	4
fabric-iot/ca	CA node	2
fabric-iot/peer	peer node	4
fabric-iot/orderer	orderer node	1
hyperledger/fabric-tools	tools of hyperledger	1
fabric-iot/chaincode/PC	PolicyContract node	4
fabric-iot/chaincode/DC	DeviceContract node	4
fabric-iot/chaincode/AC	AccessContract node	4

Figure 4.9: Connections of structure of initialization.

Step1: Use the Hyperledger cryptogenic tool to generate root certificates and secret key pairs for nodes (peer, orderer, etc.).

Step2: Move those certificates and secret key pairs to the directory specified, which will be mounted by a docker image of CA and take effect when the container running. Other nodes can authenticate their identity to the CA with their signatures.

Step3: Use the configtxgen tool to generate a genesis block, which is used to package transactions contains the configuration of nodes and channels.

B. IMPLEMENT OF ABAC: There are some ways to invoke chain code in Hyperledger Fabric platform. The outsiders can use client or SDK (support Java, go lang, node) to invoke chain code in Hyperledger Fabric platform. Fabric-iot uses the client written by node SDK to invoke chain code. The steps are as follows.

Step1:CA node generates secret key pairs for client, which are saved in a wallet of user.

Step2: Admin runs a client to connect to the peer node to submit or evaluate (corresponding write and read operations) a transaction. Step3: Peer node queries or updates the SDB by making a consensus with other peer nodes under the service of ordered node.

CHAPTER 5

RESULTS

A conceptual integrated framework for fabric-iot block chain-based operations of systems is achieved. A management technique and a performance evaluation method for fabric-iot are defined. Data Fabric provides a consistent interface that makes it easy to provide IoT data to specific workloads or applications. The need for this type of solution is magnified when the huge amount of data from IoT starts to become part of an organization's overall data asset.

CHAPTER 6

CONCLUSION

In the proposed approach, it combines the blockchain technology with the ABAC model, takes advantages of the blockchain technology such as decentralization, tamper-proof and trace-ability, solves the problem that the traditional access control method based on the centralized designs is difficult to meet the access control requirements in IoT. Firstly, according to the actual production data of the IoT devices, we propose a device authority model. Secondly, according to the ABAC model, we implement the ABAC policy management and ensures the access security of the device resources by implementing the smart contract application.

CHAPTER 7

FUTURE ENHANCEMENTS

The implementation of experimental verification and validation for the Fabric-Iot with block chain combination. Furthermore, an open-source access control system named fabric-iot based on Hyperledger Fabric is designed and implemented. This system adopts distributed architecture, which can provide fine-grained and dynamic access control management for the physical network. Finally, the steps of blockchain network building, chain code installation, smart contract invoking are described in detail, and the experiments show a convincible result. Above all, this paper provides a practical reference for other researchers to carry out relevant research. Future works can be improved in the following aspects:

1) The experiments in this paper is carried out on two PCs. In the future, we consider using the cluster or the edge computing service to deploy, and further verify the distributed performance of this system.

2) In the future, more physical devices can be used to test the reliability and throughput of the system.

3) Future research can try to improve the scalability of fabric-iot and to support more IoT application integration.

BIBLIOGRAPHY

[1] [K. Kostal, P. Helebrandt, M. Bellus, M. Ries, and I. Kotuliak, ‘‘Management and monitoring of iot devices using blockchain,’’ Sensors, vol. 19, no. 4, p. 856, Feb. 2019.

[2] J.Yang, S.He, Y.Xu, L.Chen, and J.Ren, ‘‘A trusted routing scheme using blockchain and reinforcement learning for wireless sensor networks,’’ Sensors, vol. 19, no. 4, p. 970, Feb. 2019

[3] S. Ding, J. Cao, C. Li, K. Fan, and H. Li, ‘‘A novel attribute-based access control scheme using blockchain for IoT,’’ IEEE Access, vol. 7, pp. 38431–38441, 2019.

[4] M. Cui, D. Han, and J. Wang, ‘‘An efficient and safe road condition monitoring authentication scheme based on fog computing,’’ IEEE Internet Things J., vol. 6, no. 5, pp. 9076–9084, Oct. 2019.

[5] M. Ma, G. Shi, and F. Li, ‘‘Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario,’’ IEEE Access, vol. 7, pp. 34045–34059, 2019.

[6] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, ‘‘Block chain based decentralized trust management in vehicular networks,’’ IEEE Internet Things J., vol. 6, no. 2, pp. 1495–1505, Apr. 2019

[7] O.Novo, ‘‘Scalable access management in IoT using blockchain: A performance evaluation,’’ IEEE Internet Things J., vol. 6, no. 3, pp. 4694–4701, Jun. 2019

[8] H. Li and D. Han, ‘‘Edu RSS: A blockchain-based educational records secure storage and sharing scheme,’’ IEEE Access, vol. 7, pp. 179273–179289, 2019.