Thesis for the Degree of Master of Engineering

Blockchain-Based Access Control and Auditing for User-Centric Medical Record Management

by

Si-wan Noh

Department of Information Security

The Graduate School

Pukyong National University

February 2018

Blockchain-Based Access Control and Auditing for User-Centric Medical Record Management

환자 중심의 개인의료기록 공유를 위한 블록체인 기반의 접근관리 시스템과 감사기술

Advisor: Prof. Kyung-Hyune Rhee

by

Si-wan Noh

A thesis submitted in partial fulfillment of the requirements
for the degree of

Master of Engineering

in the Department of Information Security,
the Graduate School,
Pukyong National University

February 2018

Blockchain-Based Access Control and Auditing for User-Centric Medical Record Management

A dissertation

by

Si-wan Noh

Approved by:

(Chairman) ***Sang Uk Shin***

(Member) ***Won Shin***

(Member) ***Kyung-Hyune Rhee***

February 2018

Contents

List of Figures

환자 중심의 개인의료기록 공유를 위한 블록체인 기반의 접근관리 시스템과 감사 기술

노 시 완

부경대학교 대학원 정보보호학 협동과정

Abstract

의료 환경과 인프라의 발전으로 인해 환자는 치료 혹은 정기적인 검사를 받기 위해 자신의 지역적인 문제를 고려할 필요가 줄어들었고 자신의 의료기록을 타 지역에 있는 의사에게 네트워크를 통해 전달하여 진단을 받는 헬스케어 서비스가 각광을 받고 있다. 하지만 전문적인 진단을 위해 관련기관들 간의 개인의료정보(Personal Health Record)의 공유는 환자의 프라이버시 침해 문제를 발생시킬 수 있는 가능성을 내포하고 있다. 또한 이러한 정보의 노출은 환자에게 금전적인 타격이나 개인의 사회적 지위에 대해 악영향을 끼칠 수 있으며 때로는 환자의 생명에 위협이 될 수도 있다. 많은 연구자들이 클라우드 기반의 의료정보 공유 기법을 제안하였으나 클라우드 서비스 제공자를 신뢰기관으로 설정하여 환자의 의료정보 접근에 대한 관리를 서비스 제공자가 수행 함으로서 서비스 제공자에 대한 신뢰를 가정하는 시스템 구조의 한계를 가지고 있었다. 본 논문에서는 블록체인이 가지는 기록된 정보의 투명성과 수정의 어려움과 같은 특성을 사용하여 안전한 의료정보의 공유를 위한 의료정보관리시스템을 제안한다. 클라우드 서버에 저장된 환자의 의료기록에 대해 시스템에 참여하는 환자와 환자의 의료기록에 접근하는 모든 참여자(의사, 약사, 보험사 등)의 접근기록은 시스템의 모든 참여자가 가지는 블록체인에 기록되므로 환자는 별도의 기관을 통하지 않고 의료기록에 대한 접근제어가 가능하며 언제든지 저장된 기록에 대한 모든 접근기록에 대한 감사를 수행할 수 있다. 또한 본 논문에서는 의료기록이 저장되는 클라우드 서비스 제공자에 대해 환자의 프라이버시를 보장하기 위해 프록시 재암호화 기법을 사용하여 환자가 허가한 사용자만 해당 기록에 접근할 수 있도록 한다.

Chapter 1. Introduction

1.1 Motivation

In the past, medical records were kept in the form of a paper document by hospitals. Hence, many patients received medical care service from their family doctor and a lot of medical records are kept by their family doctor. However, paper documents required space for storing it. Also, to retrieve a record from document storages is difficult. These days, with the development of the technology, patients receive healthcare service from specialists in each field (e.g. dentist, orthopedist, neurologist, etc.). To efficiently share these medical records among specialists, an electronic medical record (EMR) was proposed [1]. Traditional paper documents are converted to electronic format with the associated information as a collection of records. These records are stored in the database and provided it only if an access request of this data is valid.

When the patient receives medical service at hospitals, he/she needs a several examinations to receive medical treatment (e.g. magnetic resonance imaging, X-ray images, CT scans, etc.). Even though the patient had an examination just recently from the other hospital, it might lead to additional costs for the patient and the hospital. Sharing of medical records between specialists belonging to the same hospital is relatively easy to achieve. However, cross-institutional sharing of a medical record is complex. To protect privacy of patients from unauthorized accesses, a lot of researchers studied about cloud storage server based data sharing system [2][4][9][13].

In order to share his/her medical record with other institutions, the patient uploads his/her medical records to the cloud server. Cloud computing service enables patients to provide their medical record for multiple users with lower cost. However, the stored record contains the patient's critical and sensitive information. An exposure of this information will cause damage to the finance, social status of patients, etc. Also, these medical records are stored on a semi-trusted third-party server. That means the service provider can access these records on the server without a record owner's permission.

1.2 Overview and Contributions

In this thesis, we propose a blockchain-based medical records management system for secure medical data sharing. We focus on the access log auditing without the participation of a fully trusted third party. The patient grants access to his/her medical records on the cloud server for the reqeuster by creating token transactions. The patient can revoke these privileges just by consuming the corresponding token trasaction's output. Also, if the patient wants to update the stored medical data and its access, he/she can updates the data and its access without the support of the third party via blockchain.

In order to achieve our goal, we present a system design for user-centric data sharing without the participation of fully trusted third party. The outline of the rest of this paper is as follows. The next chapter briefly introduce the blockchain technology and related work. In chapter 3, we give a system design with our security goals, and then describe our protocol. We analyze the proposed system in Chapter IV, and finally, conclude this paper in Chapter V.

Chapter 2. Preliminary

2.1 Blockchain

The blockchain is consist of consecutive blocks. Each block contains a hash value of the previous block, a timestamp and transactions. A transaction that is created by users is popagated to a Peer-to-Peer(P2P) network. Nodes in the P2P network propagate the received transaction after perform a transaction validation process. If the transaction is invalid, it cannot be propagated to the P2P network. Also, users store a copy of the blockchain onto the local storage. If someone wants to modifiy data on the blockchain, a nonce value in the block hearder must be modified together. However, it is not possible without the alteration of all subsequent block headers for all users in the network. In Figure 1, we give a basic structure of the Bitcoin blockchain and a structure of the block header is given in Figure 2.

The first distributed blockchain was conceptualised by Satoshi Nakamoto[14] and it was implemented the following yesr as a Bitcoin cryptocurrency's basic compoent. In Bitcoin, a user creates a transfer transaction with his/her digital signature that use Ellpitic Curve Digital Signature Algorithm (ECDSA) and then it is broadcasted to the bitcoin network. These transactions are verified by network members, usually based on the digital signatrue verifying the ownership of the previous transaction output. Only if the transaction has a valid digital signature and the previous transaction output is unconsumed, it can be disseminated by network members.

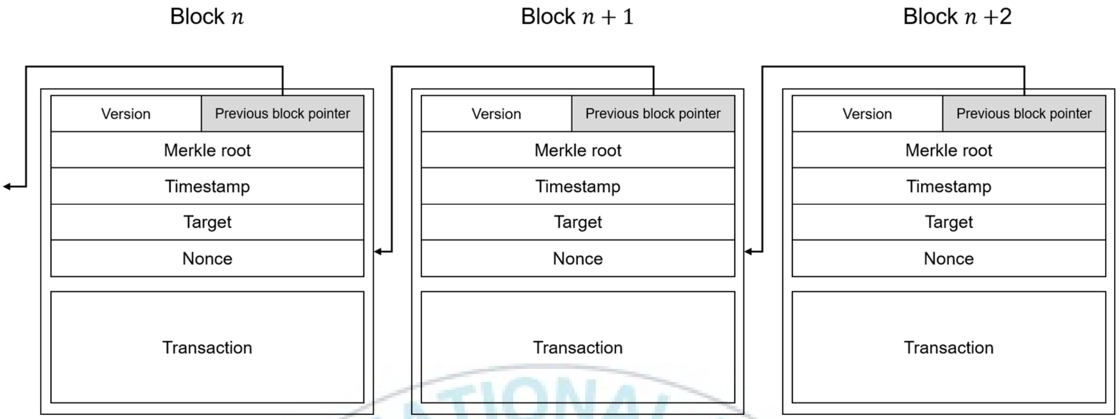
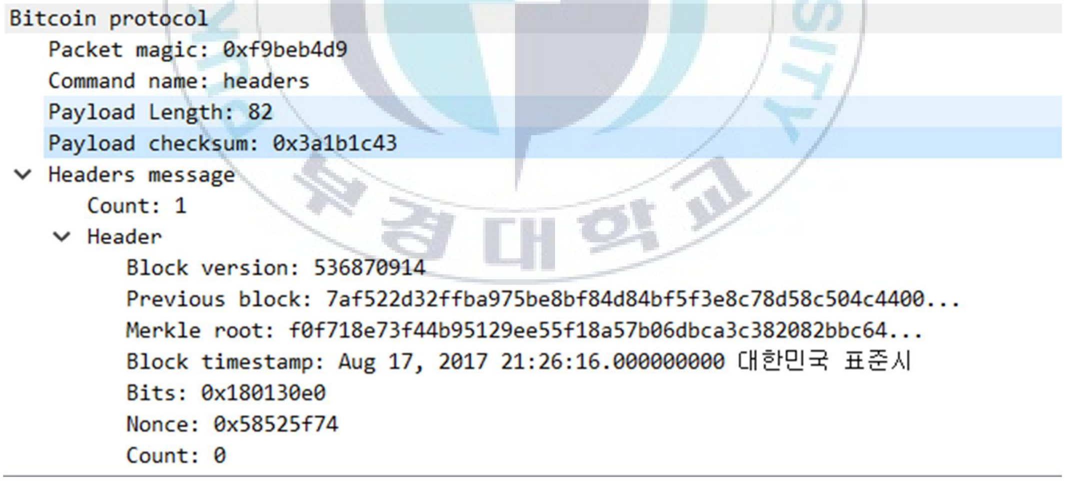


Figure 1. **Bitcoin Block Structure**. Each block contain a hash pointer as a link to a previous block. When the attacker want to modify a transaction in the block n, he/she must modify all the following blocks in the blockchain.



```
Bitcoin protocol
   Packet magic: 0xf9beb4d9
   Command name: headers
   Payload Length: 82
   Payload checksum: 0x3a1b1c43
 ⌄ Headers message
      Count: 1
    ⌄ Header
         Block version: 536870914
         Previous block: 7af522d32ffba975be8bf84d84bf5f3e8c78d58c504c4400...
         Merkle root: f0f718e73f44b95129ee55f18a57b06dbca3c382082bbc64...
         Block timestamp: Aug 17, 2017 21:26:16.000000000 대한민국 표준시
         Bits: 0x180130e0
         Nonce: 0x58525f74
         Count: 0
```

Figure 2. Bitcoin Block Structure, as seen in Wireshark

2.2 Related Work

In this thesis, we consider the following situation: patients want to provide their medical record for a doctor, pharmacist and much more. They store medical records to the cloud server. In this situation, patients grant an access right to the medical record for the doctor and he/she can access the patient's medical record on the cloud server until his/her access right is revoked by the patient. If patients want to audit record access logs, these logs must be kept by someone. In the cloud computing environment, the cloud service provider plays the role of an access control administrator and an auditor. However, if the cloud service provider is compromised, the patient obtains an incorrect information in process of auditing. Also, these logs contain patient's sensitive information like medical records. To prevent an unauthorized access to these logs, only the record owner can obtain the access logs.

2.2.1 Medical Data Sharing

In the previous chapter, we describe a necessity of the medical data sharing to reduce the cost of unnecessary examination. An electronic health information exchage(HIE) allows doctors, nurses, pharmacists, healthcare service providers and patients to access and share a patient's medical data electronically. In the data sharing system to exchange the patient's medical data electronically, all datas are governed by storing the patient's medical data to the service provider's server. In [18], Li et al. proposed a secure health records sharing scheme in the cloud computing environment. To protect the patient's medical data from unauthorized access, medical datas are encrypted under a set of attributes. If the user has proper keys for access, he/she can decrypt stored medical data in the cloud server. In [19], Zhang et al. proposed a consent-based access control to achieve user-centric access control for secure medical data sharing. A consent is an authrization initiated by the patient for the data requester via an agreement between them. This consent is used to access the patient’s medical data stored in a data center.

However, if the patient want to know an access log to audit all access attempt, it can be achieved only via a semi-trusted entity (e.g. service provider). Even if the patient generates the access token, he/she does not know whether it is used or not.

2.2.2 Proxy Re-encryption

To overcome the untrustworthy service provider problem, patients encrypt their medical records under a secret key before storing it to the cloud server. The easiest way to sharing encrypted records between different users is to share the secret key with users. However, it could allow a malicious user to access unauthorized data. In [17], Blaze et al. introduced proxy re-encryption (PRE) scheme. With the PRE scheme, a data is encrypted under patient’s public key before storing it. The patient generates proxy re-encryption key and sends it to the cloud server. Using proxy re-encryption key, the cloud server transforms the encrypted medical record under patient’s public key into an encrypted medical record under requester’s public key on the same record. In [8], Sur et al. proposed a certificateless proxy re-encryption scheme based on bilinear pairing. The scheme prevents the proxy from colluding attack and achieve chosen ciphertext attack security. Even if the requester colludes with the cloud server, they cannot reveal patient’s secret key.

2.2.3 Blockchain-based Access Control

To achieve an auditable access control, blockchain-based access control techniques are proposed [10][11][12]. In [11], Maesa et al. proposed blockchain-based access control system. Each transaction expresses an access right and policies, all transactions are publicly visible on the blockchain. In their proposed system, all users can efficiently exchange their right with other users and their idea motivated our work. However, most studies are focused on access control; they do not consider privacy and secure data sharing.

Unlike the non-cloud computing environment, all file in the cloud comuputing environment is stored on the cloud stroage and is governed by the cloud service provider (i.e., users no longer physically posses the storage of their file). As previously mentioned, if the cloud service provider is compromised, he/she can modify the stored file. To overcome this problem, a notion of the third party auditor (TPA) was proposed. The TPA can efficiently audit the data of users on behalf of data owners without the added costs of auditing the data. In Figure 3, we give a basic archtecture of the TPA service.

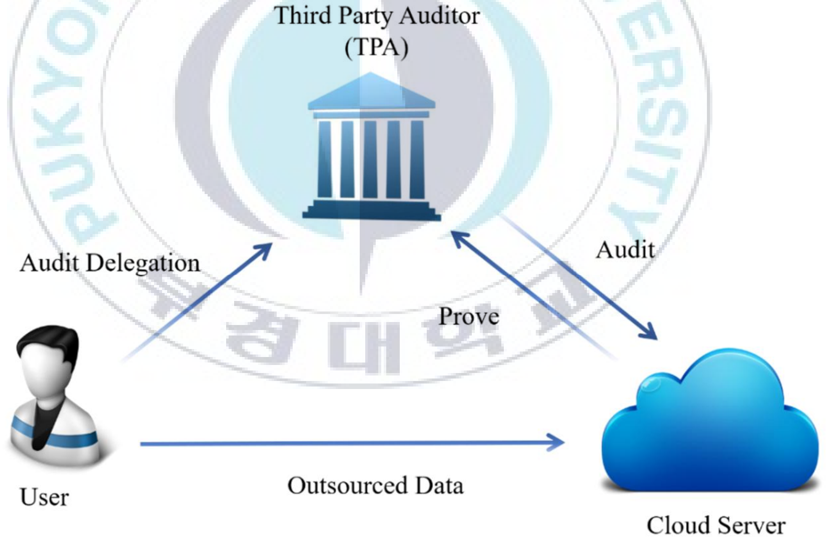


Figure 3. **The architecture of the TPA service**. The user delegates access to his/her data in the cloud server. The TPA audits on the data in the cloud server with low-cost.

The TPA is performed by an audit organization independent of the user-service provider relationship. However, the user privacy problem associated with a still remains.

To avoid this situation, we use the blockchain as a public ledger to which events are posted. This feature of blockchain – as a public ledger in which events are transparent – enables an user-centric auditing without participation of the third party.

However, the transparency is both it's strength and also weakness. In the blockchain every transaction is recorded to the public database, a public ledger that is shared with every participant in the blockchain network. To solve this problem, a CoinJoin technique is proposed by Gregory Maxwell as an anonymous method for Bitcoin transactions. When a user wants to make a transaction without exposing the link between his/her address and the money movement in the blockchain, he/she find someone else who also want to make transaction anonymously and then make a joint transaction with them. Even though a malicious user see this trasaction, he/she could not track a specific output of the transaction. An example of the CoinJoin transaction is given in Figure 4.

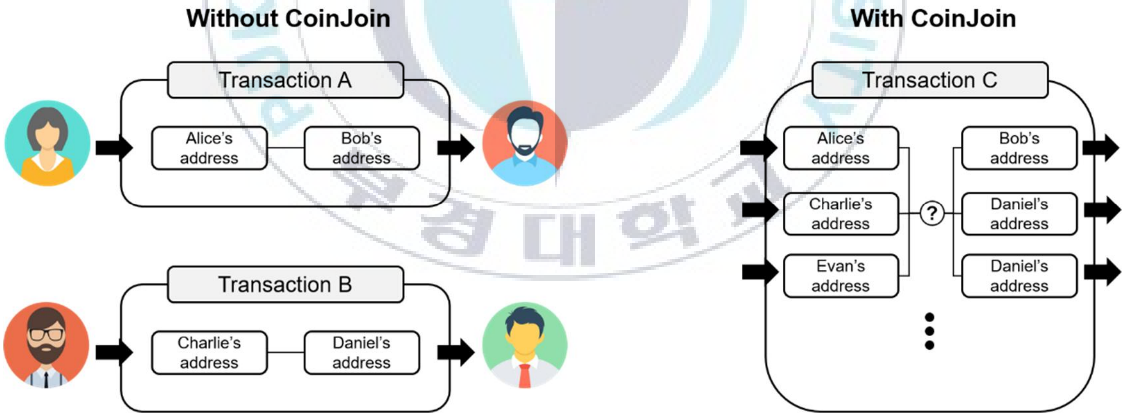


Figure 4. **An Example of the CoinJoin Transaction**. When making a joint payment, there is no way to relate input and ouputs in one transaction and thus the exact direction of money movement remains unknown to third parities.

Chapter 3. Blockchain-Based Record Management

In this chapter, we describe the proposed secure medical record management system. The proposed system consists of setup, enrollment, grant access, get access, update and revocation phases.

3.1 Proposed System Model

We consider a system model shown in Figure 6 which consists of a certification authority, cloud service provider, requesters, and patients.

- **Patient:** A patient stores encrypted Personal Health Records(PHRs) onto the cloud server. When granting an access right to PHRs for requesters, the patient creates a token transaction and broadcasts it to the blockchain. At the same time, the patient generates a proxy re-encryption key to delegate decryption right to the grantor and sends it to the cloud service provider. In our system, all participants must be enrolled by the CA. In the enrollment phase, the patient needs e-mail authentication [7] without exposing his/her real-world identity. The patient's e-mail address is bounded to an address in the system.

- **Requester:** Requesters want to access patients medical record for research or medical treatment purposes. Unlike the patient, the requester must prove that he/she is qualified for the work to access medical records of patients (e.g. national provider identifier). If the requester wants to access medical records that are stored on the cloud storage, he/she creates a request transaction. If the requester's qualifications suit the work, the patient creates a token transaction. It used to access records of the patient on the cloud storage.

- **Cloud Service Provider (CSP):** The CSP is responsible for storing medical records of patients and transforms the encrypted medical record under patient's public key into a new ciphertext under requester's public key on the same record. The CSP stores re-encryption key for re-encryption to his/her local storage.

- **Certification Authority (CA):** In our system, we are using *a permissioned blockchain*. That means, users need CA's permission to join the system. Only users authorized by CA can write on the blockchain by creating transactions. To access medical records of different patients, the access requester must prove that the linkage between the address in the system and his identity in the real world. After that, the requester generates a certified address with the support of the CA. In our system, we consider the CA is *a functionally trusted entity*, i.e. the CA is assumed to be honest and fair, but it does not have access to the private keys of users[6].

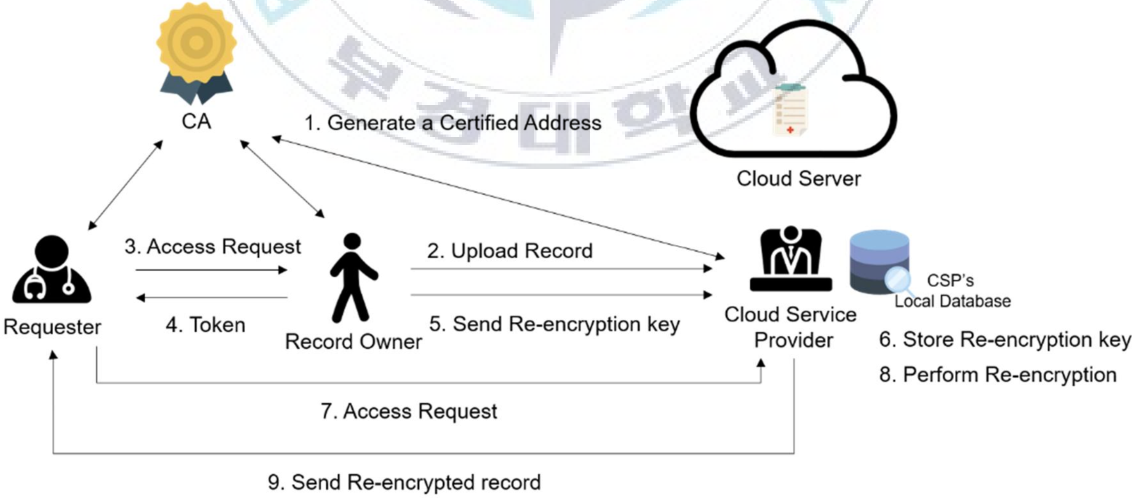


Figure 5. Proposed System Model

We also make the following assumptions to clarify the proposed system.

- Public system parameters generated by the CA are already known to all the users.
- Efficient search methods that can allow users to search transactions on the blockchain are known to users.

3.2 Security Requirements

To design a medical record management for the data sharing in cloud storages, we consider the following security requirements as our design goals.

- **Integrity and confidentiality of the content of records**: Unauthorized users who do not have the access right should not be able to access the record of different users in the cloud storage. Even if the cloud service provider is compromised, medical records of the patient must be hidden from him/her. Additionally, authorized users should be able to verify the integrity of received records.

- **Auditability:** All actions of users in the system must be recorded on the blockchain. If patients want to audit all actions of users toward his/her medical record, he/she must be able to audit event records of the system without falsification.

- **Anonymity of patient identifiers on the blockchain**: Even though the medical record information of the patient is recorded on the public blockchain, no one can be related to the real-world identity of the patient.

- **Unlinkability of requests :** All requests and results are stored in the blockchain, in which the requester's address is publicly linked with his/her access request. It means that a third party can trace the requeste's transaction and watch his/her activity in the system. To avoid this situation, the link between the requester's address and the request transaction should be hidden on the blockchain.

3.3 Proposed System

3.3.1 Preliminaries

Before describing the proposed system, we provide a brief description of the properties of a bilinear pairing and proxy re-encryption. $\mathbb{G}$ is an additive group of prime oreder q. Also, $\mathbb{G}_1$and $\mathbb{G}_2$ be two multiplicative cyclic groups of prime order q. A bilinear map is a map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

- **Bilinear.** $e(g^a, g^b) = e(g,g)^{ab}$, for all $a, b \in Z_q^*$ and $g \in \mathbb{G}_1$.
- **Non-degenerate.** If g is a generator of $\mathbb{G}_1$ then $e(g,g)$ is a generator of $\mathbb{G}_2$.
- **Computable.** $e(g,h)$ is efficiently computable for and $g, h \in \mathbb{G}_1$.

3.3.2 Setup

CA chooses a random generator $g \in \mathbb{G}$ and function $\rho: \mathbb{G} \rightarrow \mathbb{Z}_q$ and then CA picks a random $\alpha_{CA} \in \mathbb{Z}_q$ as his/her master key and compute corresponding public key $PK_{CA} = g^{\alpha_{CA}}$. CA publishes the system parameters $params_{CA} = (\mathbb{G}\ , q, g, PK_{CA}, \rho)$.

CSP chooses bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q and random generator $g_1, h \in \mathbb{G}_1$. CSP picks a random $\alpha_{CSP} \in \mathbb{Z}_q^*$ as his/her master key and compute a public key $PK_{CSP} = g^{\alpha} \in \mathbb{G}_1$. Also, CSP chooses hash functions $H: \{0,1\}^* \rightarrow \mathbb{Z}_q^*, H_1: \{0,1\}^* \rightarrow$

$\mathbb{Z}_q^*, H_2: \mathbb{G}_2^2 \rightarrow \{0,1\}^n, H_3: \{0,1\}^* \rightarrow \mathbb{Z}_q^*, H_4: \{0,1\}^* \rightarrow \mathbb{Z}_q^*, H_5: \{0,1\}^* \rightarrow \mathbb{G}_1$ and computes the group elements $g_2 = e(g, g), g_3 = e(g, h) \in \mathbb{G}_2$ then publishes the system parameters $params_{CSP} = (\mathbb{G}_1, \mathbb{G}_1, e, g_1, h, g_2, g_3, H, H_1, H_2, H_3, H_4, H_5)$.

3.3.3 Enrollment

In this phase, users who want to join the system generate a certified address with the support of CA [6] and he/she can attest to the involvement of the CA using the address. After that, the user generates a private key for a proxy re-encryption using the certified address as an identity.

User (e.g., patient, doctor, insurer, etc.) who want to enroll in the system send a request message $m_{req,x}$ to the CA with the enrollment information as follows:

(1) Select $k \leftarrow \mathbb{Z}_q$ uniformly at random and computes $h = g^k$.

(2) Send a request message $m_{req,PT_i} = \{email_{PT_I}, A_{PT_I}, pk_{PT_I}, h, \sigma_{sk_{PT_I}}(A_{PT_I}, email_{PT_I}, h)\}$ to the CA.

After receiving above message, the CA verifies signature and then sends HTML link to verify ownership of an email address. After verifying it, the CA performs as follows:

(1) Select $k' \leftarrow \mathbb{Z}_q$ uniformly at random and computes a self-certified public key $c = h \cdot g^{k'}$.

(2) Compute $e = \rho(c)$ and $\bar{x} = k' + e \cdot \alpha_{CA}$.

(3) Send a response message $m_{req,CA} = \{A_{PT_I}, pk_{PT_I}, e, \bar{x}, \sigma_{sk_{PT_I}}(A_{PT_I}, e, \bar{x})\}$ to the patient.

The user computes his/her private key $x = \bar{x} + k$ and a certified address $cA_x = H(c)$. To verify this address, the verifier needs a signature signed by private key x. So the user creates *a certification transaction* tx_{cert} by sending transaction to himself. A structure of the certification transaction is given in Figure 7. The input field in the certification transaction contains the user's public key and the corresponding ECDSA signature. Anyone

After generating the address, the patient generates key pairs for a re-encryption process with the support of the CSP as follows:

(1) The patient sends a message to the CSP with his/her certification transaction ID.

(2) After verifying the patient's address, the CSP computes $h_{PT_i} = H_1\left(cA_{PT_i}\right) \in \mathbb{Z}_q^*$ and extract patient's partial private key $d_{PT_i} = g^{1/(\alpha_{CSP}+h_{PT_i})} \in \mathbb{G}_1$.

(3) The CSP sends a hash value of the partial private key $H(d_{PT_i})$ and the patient's partial private key d_{PT_i} to the patient.

(4) The patient selects $a_1, a_2 \in \mathbb{Z}_q^*$ uniformly at random and sets $x_{PT_i} = \left(x_{PT_i,1}, x_{PT_i,2}\right) \in \mathbb{Z}_q^{*\,2}$.

(5) The patient computes a private key for encryption $rSK_{PT_i} = \left(d_{PT_i}, x_{PT_i}\right) \in \mathbb{G}_1 \times Z_q^{*\,2}$ and a public key $rPK_{PT_i} = \left(g_3^{x_{PT_i,1}}, g^{x_{PT_i,2}}\right) \in \mathbb{G}_2 \times \mathbb{G}_{1.}$

Also, requesters perform same processes (compute a certified address and key pair for re-encryption).

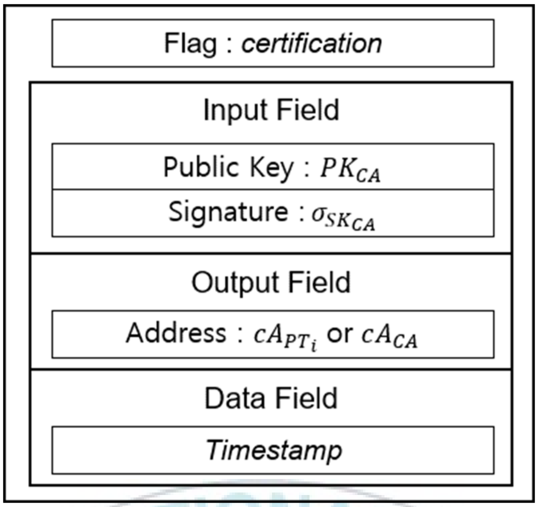


Figure 6. **A Structure of the Certtification Transaction**. Anyone can check the validty of the certification transaction by verifying the signature in the input field using the CA's address and public key. The ouput field contains a multisignature address. It is used to divide up responsibilty for possession of transaction. This transaction is similr to a publc key certificate in the public key infrastructure(PKI). The CA can revoke this transaction any time by using the transaction's output before it is used for access.

3.3.4 Grant Access

After enrollment phase, the patient uploads his/her PHRs onto the cloud server with a record identifiable information and creates a record transaction. To protect the privacy of the patient, PHRs are encrypted by his/her public key. In this phase, the requester sends an access request message with a request transaction. If the patient wants to allow this request, he/she creates a token transaction and generates re-encryption key for the requester. The process is illustrated in Figure 8.

Components of the stored record of the patient PT_i on the cloud server are as follows:

$$Recrod_{PT_i,j} = \{Enc_{rPK_{PT_i}}\left(data_j\right), H\left(data_j\right), cA_{PT_i}, txid_{rec_j}\sigma_{SK_{PT_i}}\left(H\left(data_j\right), cA_{PT_i}\right)\}$$

The ciphertext $C = Enc_{rPK_{PT_i}}\left(data_j\right)$ is computed by the patient as follows:

(1) Compute $h_{PT_i} = H_1\left(cA_{PT_i}\right)$.

(2) Choose a random $\sigma \in \{0,1\}^*$ and compute $r = H_4(data_j \parallel \sigma \parallel cA_{PT_i} \parallel rPK_{PT_i})$.

(3) Compute the ciphertext $C = (C_1, C_2, C_3, C_4)$:

$$C = (\left(g^{h_{PT_i}} \cdot PK_{CSP}\right)^r, h^r, \left(data_j \parallel \sigma\right) \oplus H_2\left(g_2^r \parallel \left(g_3^{x_{PT_i,1}}\right)^r\right), u^r)$$

(where $u = H_5(cA_{PT_i} \parallel rPK_{PT_i} \parallel C_1 \parallel C_2 \parallel C_3)$)

If a doctor wants to access the patient's PHRs in the cloud server, the patient should inform his/her record transaction ID to the doctor. After acquiring it, the doctor creates a request transaction using the CoinJoin technique. After a moment, when the request transaction appears on the patient's blockchain, the patient creates a token transaction only if the requester's certification transaction is valid. In this phase, we consider two types of relationships between transactions.

The first is like an Unspent Transaction Output (UTXO) in Bitcoin [14]. In the Bitcoin payment system, only unspent outputs can be used as inputs to a new transaction. When the user in the Bitcoin payment network received new blocks from his/her neighbor node, he/she verifies the validity of transactions in the received block and stored their outputs to his/her local memory pool (UTXO pool). When a transaction is confirmed by consensus nodes, inputs are deleted from a UTXO pool and outputs are added to the pool as a new UTXO. In our system, similarly, the token transaction is created by the record owner (i.e. patient). It can be consumed as a token for the access record on the cloud server. That is, the token can only be spent once. The used-token cannot be reused to access the data like in the case of Bitcoin.

Second relationship is similar to the first one. The difference between the two relationship is the second relationship allowing the reuse of the used output (i.e. the single input is spent more than once). In our thesis, we call this relationship as a '*reference relationship*'. In our system, the record transaction's output is one of the inputs of the token transaction. If token transactions are created by the same patient, at the same time, their token transaction uses the same transaction output as a transaction input. To do so, if the transaction input '*refers*' the previous transaction's output, the output does not remove from the UTXO pool until it is used to create a record update transaction.

When the record owner creates a token transaction, he/she uses the unspent output of the requester's request transaction and refers to the record transaction of the record owner. If the token transaction is broadcast on the blockchain network, consensus nodes check that the signature in the input field is valid or not using the address in the output filed of the previous transaction (both record transaction and request transaction). However, if the output of the record transaction is consumed by the record owner to update his/her record, this token transaction is invalid (If the requester wants to access the updated record, he/she needs to request the record again).

Before creating a token transaction, the record owner generates re-encryption key for the requester. The token transaction including a hash value is based on the re-encryption key generated by the record owner and the address of the requester. The record owner sends the re-encryption key to the CSP with the transaction ID of the token transaction (This token transaction contains a hash value of the corresponding proxy re-encryption key). The CSP stores it in a local storage with the transaction ID. The record owner computes the re-encryption key $rk_{PT_i \to req}$ the following steps:

(1) Choose a random $s \in \mathbb{Z}_q^*$ and compute $\mu = H_3(g_2^s \parallel cA_{PT_i} \parallel rPK_{PT_i} \parallel cA_{req} \parallel rPK_{req})$.

(2) Compute $h_{PT_i} = H_1(cA_{PT_i})$ and $h_{req} = H_1(cA_{req})$.

(3) Set the proxy re-encryption key as following:

$$\begin{aligned} rk_{PT_i \to req} &= \left(rk_{PT_i \to req}^{(1)}, rk_{PT_i \to req}^{(2)}, rk_{PT_i \to req}^{(3)}\right) \\ &= \left(g^{\left(\alpha_{CSP} + h_{PT_i}\right)}\right), \left(g^{h_{req}} \cdot g_1\right), \left(g^{x_{req,2}}\right)^{x^{PT_i,1}}. \end{aligned}$$

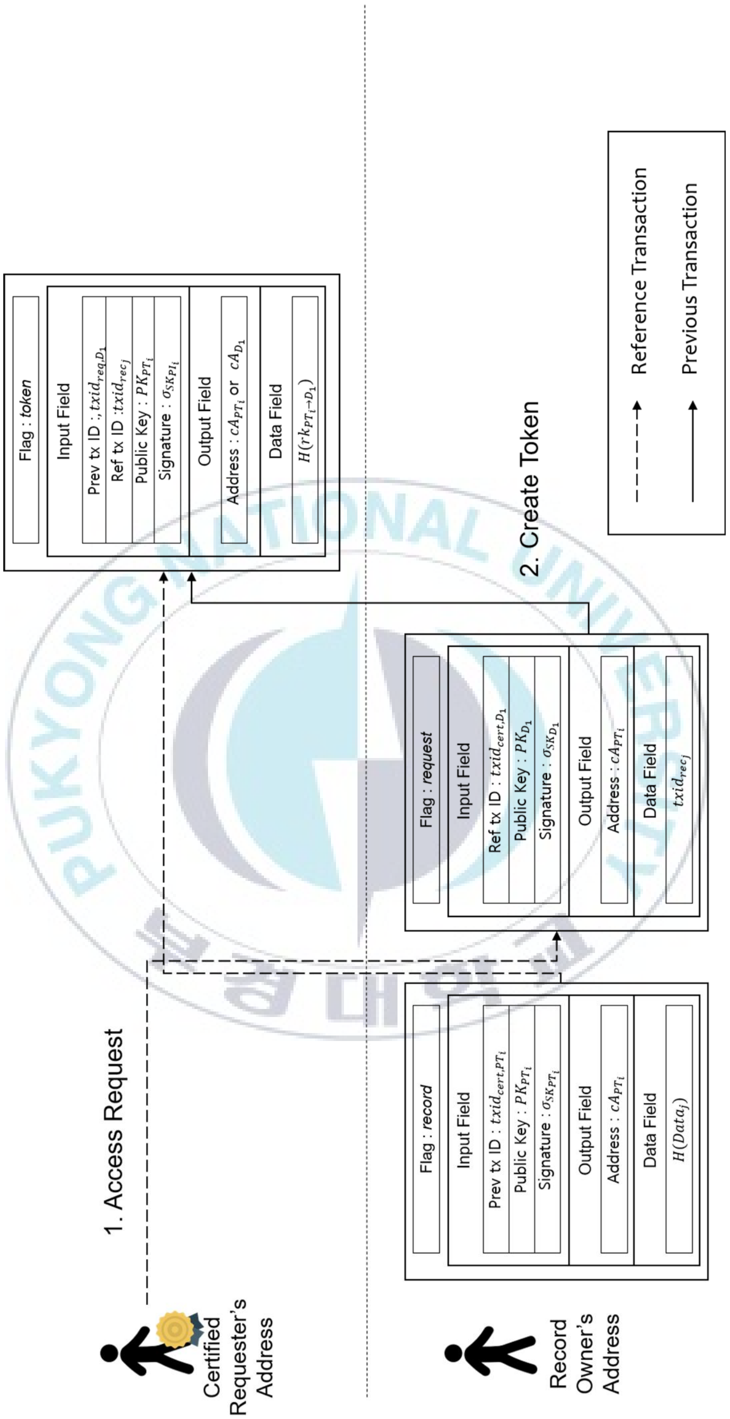


Figure 7. Transaction Relationship in the Grant Access Phase

3.3.5 Get Access

After gaining a token to the medical record on the cloud server, the requester creates a get access transaction tx_{get} to get the re-encrypted medical record. When it is added to the CSP's blockchain, the CSP verifies requester's token transaction. Only if the token transaction is valid and the comparison of a hash value of the re-encryption key stored in local storage with the hash value in the token transaction is the same, the CSP performs re-encryption process as a proxy and sends a re-encrypted medical record to the requester. After sending it, the CSP creates a receipt transaction that contains a hash of the re-encrypted record for the integrity of the data. The process is illustrated in Figure 9.

(1) Check the validation of the request.
(2) Compute $u = H_5(cA_{PT_i} \parallel rPK_{PT_i} \parallel C_1 \parallel C_2 \parallel C_3)$ and $h_{PT_i} = H_1(cA_{PT_i})$
(3) Check that $e(C_1, u) = e\big((g^{h_{PT_i}} \cdot PK_{CSP}\big), C_4)$ and $e(C_2, u) = e(h, C_4)$.
(4) Compute $C_1' = e(C_1, rk^{(1)}_{PT_i \rightarrow req})$ and set $C_1'' = rk^{(2)}_{PT_i \rightarrow req}$.
(5) Compute $C_2' = e(C_2, rk^{(3)}_{PT_i \rightarrow req})$
(6) Set the new ciphertext $C' = \big(C_1', C_1'', C_2', C_3, cA_{PT_i}, rPK_{PT_i}\big)$.
(7) Send the re-encrypted record C' to the requester.
(8) Create a receipt transaction $tx_{receipt}$.

The requester decrypts the re-encrypted record C' as follows.
(1) Check the hash value in the receipt transaction and its validation.
(2) Compute $\kappa = e(C_1'', d_{req})$ and $\mu = H_3(\kappa \parallel cA_{PT_i} \parallel rPK_{PT_i} \parallel cA_{req} \parallel rPK_{req})$.
(3) Compute $\omega = {C_1'}^{1/\mu} \parallel {C_2'}^{1/x_{req,2}}$ and then $\big(data_j \parallel \sigma\big) = C_3 \oplus H_2(\omega)$.
(4) If $C_1' = g_2^{\mu \cdot r}$ and $C_2' = (g_3^{x_{PT_i,1}})^{x_{req,2} \cdot r}$, where $r = H_4(data_j \parallel \sigma \parallel cA_{PT_i} \parallel rPK_{PT_i})$, return $data_j$ as the plaintext of the medical record.

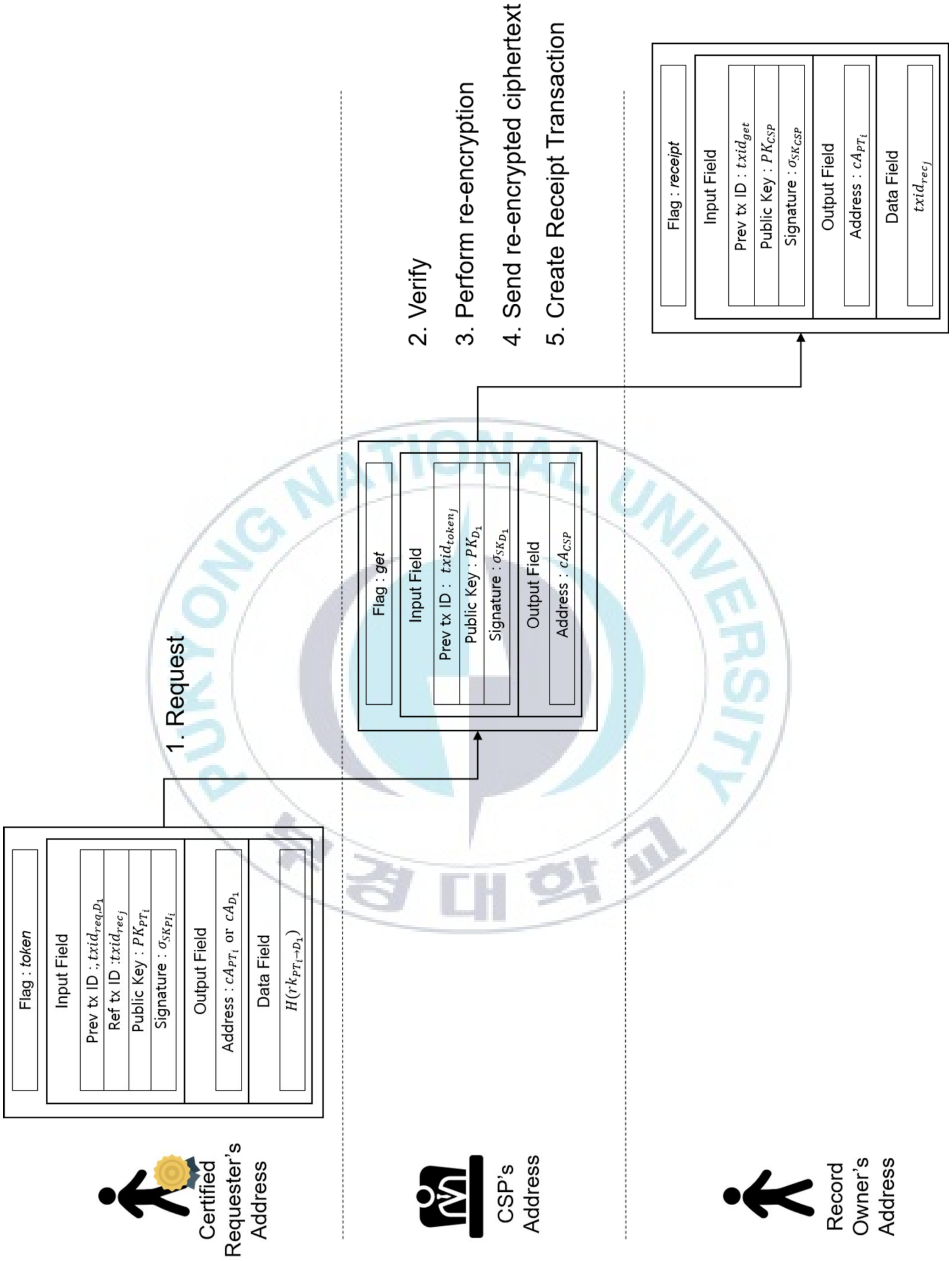

Certified Requester's Address
CSP's Address
Record Owner's Address
Flag : *token*
Input Field
Prev tx ID : $txid_{req,D_1}$
Ref tx ID : $txid_{rec_j}$
Public Key : PK_{PT_i}
Signature : $\sigma_{SK_{PT_i}}$
Output Field
Address : cA_{PT_i} or cA_{D_1}
Data Field
$H(rk_{PT_i \rightarrow D_1})$
1. Request
Flag : *get*
Input Field
Prev tx ID : $txid_{token_j}$
Public Key : PK_{D_1}
Signature : $\sigma_{SK_{D_1}}$
Output Field
Address : cA_{CSP}
2. Verify
3. Perform re-encryption
4. Send re-encrypted ciphertext
5. Create Receipt Transaction
Flag : *receipt*
Input Field
Prev tx ID : $txid_{get}$
Public Key : PK_{CSP}
Signature : $\sigma_{SK_{CSP}}$
Output Field
Address : cA_{PT_i}
Data Field
$txid_{rec_j}$

Figure 8. Transaction Relationship in the Get Access Phase

3.3.6 Revocation

The output filed of the token transaction contains the requester's address and the record owner's address in the system. That is, the record owner can consume the token transaction as well as the requester. The record owner could revoke the token transaction at any time by consuming its output. It can be implemented using Multisignature (multisig) in the Bitcoin payment system. It requires the signature of multiple user before the output of the transaction can be consumed. In our system, we use 1-of-2 multisig to achieve our requirement. After spending the token transaction by the record owner, it cannot be used for record access.

3.3.7 Update

The patient's PHRs are frequently updated. However, the record transaction contains a hash value of PHRs at some point in the past. In our system, the record owner can update his/her record transaction on the blockchain using an update record transaction. The record owner just consumes the output of the previous record transaction and creates a new record transaction with the hash value of the updated record. After creating the updated record transaction, unused token transactions are no longer available. In Figure 10, we give a example of the update record and access.

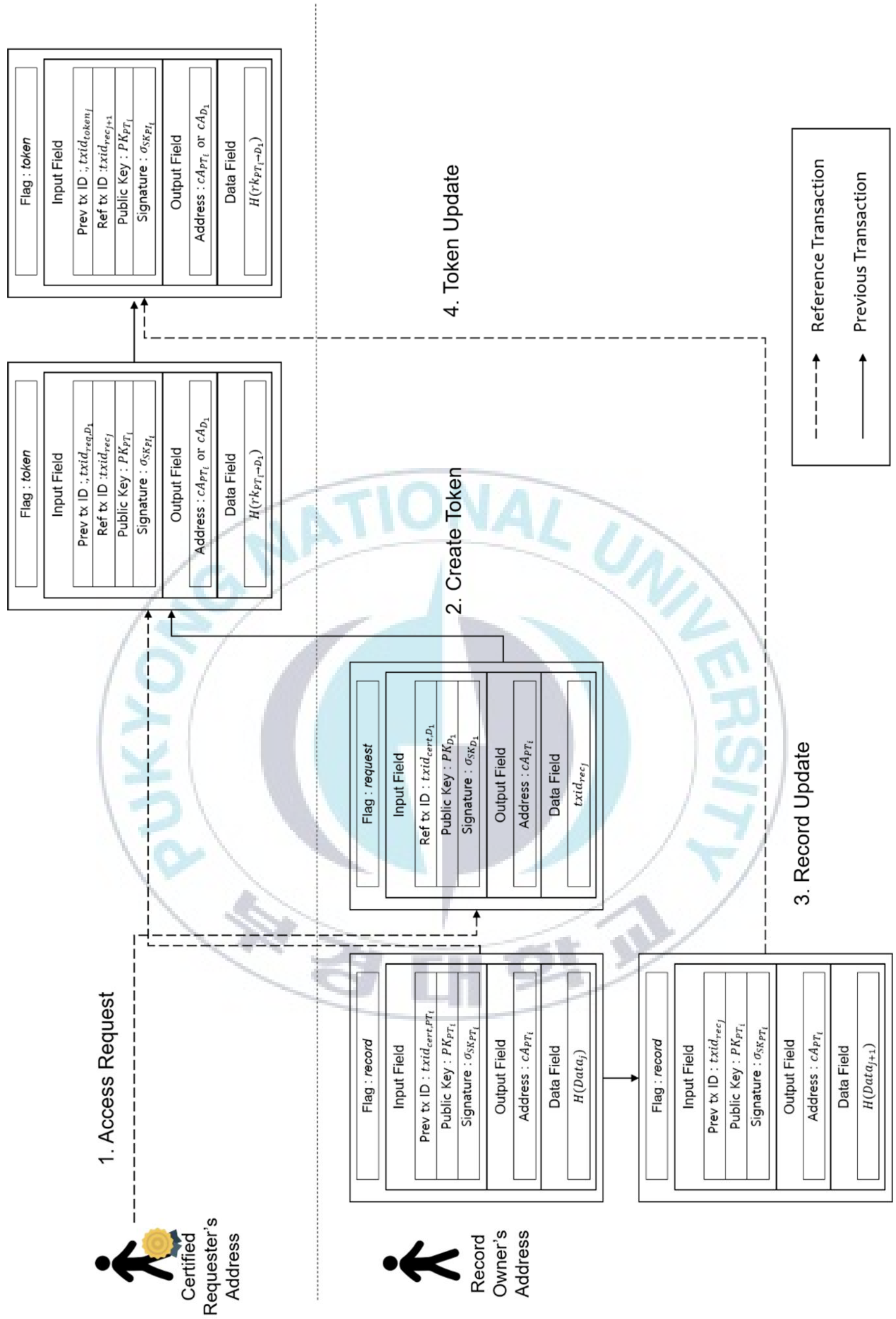


Figure 9. Record Update Phase

Chapter 4. Security Analysis

4.1 Integrity and Confidentiality of the Content of Records

In our system, patients store their medical records to the cloud storage. Records in the cloud server are encrypted by their own secret key to protect their privacy from malicious cloud service provider. When patients upload their medical records to the cloud server, they create a record transaction that contains a hash value of records. It is like a timestamp of the record; the requester can verify the validity of the record when the requester gets it. If malicious cloud service provider wants to change some record in the cloud server for malicious, he/she must change a corresponding record transaction and the block header that containing the record transaction. However, it will be hard, because of an immutability of the blockchain. Also, a confidentiality of the content of records is can be guaranteed by a proof of the scheme in [8].

4.2 Auditability

After the requester using the token transaction (i.e. create get access transaction), the cloud service provider creates a receipt transaction for the record owner. The record owner can trace the access log back through connected transactions. All transactions in the blockchain include a timestamp of the user and an address in the system. It can be possible that the record owner obtains a timeline for auditing. Also, the record owner can obtain these logs without a communication with the cloud service provider.

4.3 Unlinkability of Requests

In the grant access phase, the requester creates the CoinJoin transaction with other requesters. As shown in Figure 11 (left), when the requester does not use the CoinJoin technique to create a request transaction, anyone can link the requester's identity and a request transaction. Even though the patient's anonymity is guaranteed, the linkage between the requester and his/her patients is a potential threat to the requester or patients, even our system. If requesters create the request transaction with them using the CoinJoin technique, nobody can link the requester's identity and his/her request transaction. It is the same for the patient also. However, the patient does not need the requester's real identity to determine the granting of access. The patient just needs the requester's qualifications for the work. If one of them does not satisfied it, that transaction is cannot be broadcast to the network. An example of the CoinJoin transaction is given in Figure 11 (right).

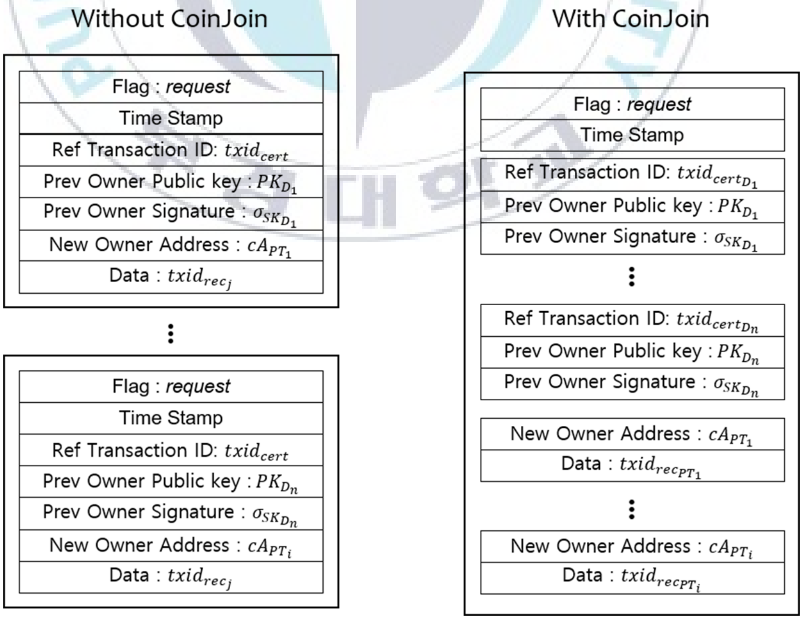

Without CoinJoin
Flag : *request*
Time Stamp
Ref Transaction ID: $txid_{cert}$
Prev Owner Public key : PK_{D_1}
Prev Owner Signature : $\sigma_{SK_{D_1}}$
New Owner Address : cA_{PT_1}
Data : $txid_{rec_j}$
⋮
Flag : *request*
Time Stamp
Ref Transaction ID: $txid_{cert}$
Prev Owner Public key : PK_{D_n}
Prev Owner Signature : $\sigma_{SK_{D_n}}$
New Owner Address : cA_{PT_i}
Data : $txid_{rec_j}$
With CoinJoin
Flag : *request*
Time Stamp
Ref Transaction ID: $txid_{cert_{D_1}}$
Prev Owner Public key : PK_{D_1}
Prev Owner Signature : $\sigma_{SK_{D_1}}$
⋮
Ref Transaction ID: $txid_{cert_{D_n}}$
Prev Owner Public key : PK_{D_n}
Prev Owner Signature : $\sigma_{SK_{D_n}}$
New Owner Address : cA_{PT_1}
Data : $txid_{rec_{PT_1}}$
⋮
New Owner Address : cA_{PT_i}
Data : $txid_{rec_{PT_i}}$

Figure 10. A Structure of the CoinJoin Transaction

4.4 Anonymity of Patient Identifiers on the Blockchain

In the enrollment phase, patients provide their email address to the CA. In our system, the CA does not need real-world identities of patients to register. Even if someone provides a hacked email-address to the CA in the enrollment phase, it does not affect our system. In the case of the Bitcoin, real-world identities are revealed in the buying process. However, in our system, real-world identities of users are not revealed across all phases of system without exposing the relationship between the identity and the address themselves.

Chapter 5. Conclusion

In cloud based data sharing system, cloud service providers can know all accesses to stored records. However, if the record owner wants to know access logs to the stored record, he/she asks them for an auditing. If the cloud service provider is compromised, the record owner obtains incorrect result. In this thesis, we proposed a blockchain-based secure data sharing system in the cloud storage. Due to the features of a blockchain technology, we can make record owners control who can access to their medical record in the cloud server without participant of the fully trusted third party.

References

[1] T. J. Hannan, "Electronic medical records", Health informatics: An overview vol. 133, (1996).

[2] T. Ermakova and B. Fabian, “Secret sharing for health data in multi-provider clouds”, Proceedings of the 15th Conference of Business Informatics (CBI), (2013).

[3] M. Li, S. Yu, Y. Zheng, K. Ren and W. Lou, “Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption”, IEEE transactions on parallel and distributed systems, vol. 24, no. 1, (2013), pp. 131-143.

[4] S. Ruj, M. Stojmenovic and A. Nayak, “Decentralized access control with anonymous authentication of data stored in clouds”, IEEE transactions on parallel and distributed systems, vol. 25, no. 2, (2014), pp. 384-394.

[5] W. S. Ng, B. C. Ooi, K. L. Tan and A. Zhou, “Peerdb: A p2p-based system for distributed data sharing”, Proceedings of the 19th International Conference on Data Engineering (ICDE03), (2003).

[6] G. Ateniese, A. Faonio, B. Magri, and B. De Medeiros, “Certified bitcoins”, Proceedings of the 12th International Conference on Applied Cryptography and Network Security(ACNS), (2014).

[7] S. L. Garfinkel, “Email-based identification and authentication: An alternative to PKI?”, IEEE Security & Privacy, vol. 99, no. 6, (2003), pp. 20-26.

[8] C. Sur, C. Jung, Y. Park and K. Rhee, “Chosen-ciphertext secure certificateless proxy re-encryption”, Proceedings of the 11th International Conference on Communications and Multimedia Security, (2010).

[9] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du and M. Guizani, “MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain”, IEEE Access, vol. 5, (2017), pp. 14757-14767.

[10] Ouaddah, A. Elkalam and A. A. Ouahman, “Towards a novel privacy-preserving access control model based on blockchain technology in IoT”, In Europe and MENA Cooperation Advances in Information and Communication Technologies, Springer International Publishers, (2017) pp. 523-533.

[11] D. D. F. Maesa, P. Mori and L. Ricci, “Blockchain Based Access Control”, Proceedings of the 17th International Conference on Distributed Applications and Interoperable Systems, (2017).

[12] Q. Xia, E. B. Sifah, A. Smahi, S. Amofa and X. Zhang, “BBDS: Blockchain-Based Data Sharing for Electronic Medical Records in Cloud Environments”, Information 2017, vol. 8, no. 2, (2017), pp. 44-59.

[13] S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing”, Proceedings of the 29th Conference on Compute Communication, (2010).

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

[15] G. Zyskind, and O. Nathan, “Decentralizing privacy: Using blockchain to protect personal data”, Proceedings of the 36th IEEE Symposium on Security and Privacy Workshops (SPW), (2015).

[16] Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, “How Blockchain Could Empower eHealth: An Application for Radiation Oncology”, Proceedings of the 3rd International Workshop on Data Management and Analytics for Medicine and Healthcare, (2017).

[17] M. Blaze, G. Bleumer and M. Strauss, “Divertible protocols and atomic proxy cryptography”, Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, (1998).

[18] M. Li, S. Yu, Y. Zheng, K. Ren and W. Lou, “Scalable and secure sharing of personal health records in cloud computing using attribute-based

encryption", IEEE transactions on parallel and distributed systems, vol. 24, no. 1, (2013), pp. 131-143.

[19] A. Zhang, A. Bacchus and X. Lin, "Consent□based access control for secure and privacy□preserving health information exchange", Security and Communication Networks, vol. 9, no. 16, (2016). pp. 3496-3508.

Acknowledgements

연구실에 들어온 지 어느덧 2년이 지났습니다. 2년동안 연구실에 오지않았더라면 해보지 못했을 많은 경험과 추억을 만들었고 어느덧 석사 졸업을 앞두고있습니다. 많이 부족했던 저를 도와 주셨던 모든 사람들에게 감사의 마음을 전하고자 글을 씁니다.

먼저 항상 많은 가르침과 조언을 주시고 많은 것을 경험할 수 있도록 해주신 이경현 교수님께 가장 먼저 감사를 드립니다. 매일 아침부터 저녁까지 연구에 매진하시는 모습을 보면서 스스로 마음을 다잡고 나아가게 해 주셨으며 많은 자극이 되었습니다. 공부 외에 인생을 살아가는데 있어 많은 조언을 주셨고 앞으로의 진로에도 많은 도움을 주셨으며 박사과정을 시작하는데 있어 결정에 가장 큰 지지를 주신 교수님께 감사드립니다.

또한 바쁘신 가운데에도 제 학위논문의 심사를 맡아 주시고 조언을 해주신 신상욱 교수님과 신원 교수님께 감사 드립니다. 특히 2년전 대학원 진학에 계기를 만들어 주신 신상욱 교수님께 다시 한번 감사를 드립니다. 4학년 마지막 학기, 막연한 미래에 헤매고 있을 때 교수님의 대학원 진학 권유는 인생에 있어 가장 큰 전환점 이였다고 저는 생각합니다.

연구실 선배로써 많은 도움을 주신 박영호 선배님과 서철 선배님께도 감사드립니다. 선배님들의 도움이 있었기에 이렇게 졸업을 할 수 있었다고 생각합니다. 선배님들의 기대에 부응하는 후배가 아니었던 점을 항상 죄송스럽게 생각합니다.

대학원 입학 동기로서 여기까지 함께 온 동이선배님과 Akash. 길다면 길고 짧다면 짧은 2년 이였지만 함께 해줘서 감사합니다. 2016년 함께 생활했던 ITB 인도네시아 친구들

Vincent, Candra, Annisa, 지금은 졸업한 Lewis, Brian, Sam. 2015년 11월 졸업전까지 함께 했던 Yani, Raffino, Hari, 지금은 연구실에 박사과정으로 함께 하고있는 Sandi. 연구실에서 가장 오래 함께한 Bayu, 항상 밤에 뭔가 열심히 하는 Bruno, 2년동안 만났던 모든 외국인 학생들에게 감사하고 모두와 함께 했던 시간들이 뜻 깊고 소중한 시간이었습니다.

첫 한국인 석사 후배인 현우, 경모, 연구실 거쳐간 많은 학부생들 용순,민재,재효,진웅. 지금 함께 하고있는 민호,지형 모두 고맙고 특히 믿음 선배님 감사합니다. 혼자 연구실에서 있었으면 버티지 못했을 것을 선배님 덕에 여기까지 왔습니다. 신상욱 교수님 연구실의 시현, 지선이 연구실 행사 있을 때마다 도와줘서 고맙다.

그리고 주말에 자주 연구실 찾아오시고 많은 도움 주신 김용호, 최경화 선생님 감사합니다.

그리고 대학원 생활하며 저에게 지원을 아끼지 않으신 부모님 감사합니다.

그 외 따로 언급하지는 못했지만 2년간 만났던 모든 사람들에게 감사합니다. 이제 석사과정을 졸업하고 박사과정에 진학하며 새로운 마음가짐으로 모두의 기대에 부응할 수 있도록 노력하겠습니다. 감사합니다.

2018년 1월 연구실에서……