K.Sriram *et al*, International Journal of Computer Science and Mobile Applications,

Vol.2 Issue. 3, March- 2014, pg. 35-41 **ISSN: 2321-8363**

A SCALABLE AND SECURE SHARING OF PHR IN CLOUD COMPUTING

Mr.K.Sriram[1], Ms.N.Radhika[2]

[1]PG Student, [2]Assistant Professor

Department of Computer Science and Engineering, PRIST University, Trichy District, India

([1] pudusriram@yahoo.com)

Abstract

Personal health record (PHR) is an emerging patient-centric model of health information exchange, which is often outsourced to be stored at a third party, such as cloud providers. However, there have been wide privacy concerns as personal health information could be exposed to those third party servers and to unauthorized parties. To assure the patients' control over access to their own PHRs, it is a promising method to encrypt the PHRs before outsourcing. Yet, issues such as risks of privacy exposure, scalability in key management, flexible access and efficient user revocation, have remained the most important challenges toward achieving fine-grained, cryptographically enforced data access control. In this project, we propose a novel patient-centric framework and a suite of mechanisms for data access control to PHRs stored in semi-trusted servers. To achieve fine-grained and scalable data access control for PHRs, we leverage Transposition Ciphers (TPC) technique to encrypt each patient's PHR file. Hence it provides more secured and flexible cloud environment for the maintenance of personal health records.

Keywords: Personal health records, cloud computing, data privacy, fine-grained access control, attribute-based encryption

1.INTRODUCTION

In recent years, personal health record (PHR) has emerged as a patient-centric model of health information exchange. A PHR service allows a patient to create, manage, and control her personal health data in one place through the web, which has made the storage, retrieval, and sharing of the medical information more efficient. Especially, each .1 recently, architectures of storing PHRs in cloud computing have been proposed in [2], [3]. While it is exciting to have convenient PHR services for everyone, there are many security and privacy risks which could impede its wide option. The main concern is about whether the patients could actually control the sharing of their sensitive personal health information (PHI), especially when they are stored on a third-party server which people may not fully trust. On the one patient is promised the full control of her medical records and can share her health data with a wide range of users, including healthcare providers, family members or friends. Due to the high cost of building and maintaining specialize data centers, many PHR services are outsourced to or provided by third-party service providers, for example, Microsoft Health Vaulth and, although there exist healthcare regulations such as HIPAA which is recently amended to incorporate business associates [4], cloud providers are usually not covered entities [5]. On the other hand, due to the high value of the sensitive PHI, the third-party storage servers are often the

Vol.2 Issue. 3, March- 2014, pg. 35-41 **ISSN: 2321-8363**

targets of various malicious behaviors which may lead to exposure of the PHI. As a famous incident, a Department of Veterans Affairs database containing sensitive PHI of 26.5 million military veterans, including their social security numbers and health problems was stolen by an employee who took the data home without authorization [6].

1.1 ABE for Fine-Grained Data Access Control

A number of works used ABE to realize fine-grained access control for outsourced data [13], [14], [9], [15]. Especially, there has been an increasing interest in applying ABE to secure electronic healthcare records (EHRs). Recently, Narayanet al. proposed an attribute-based infrastructure for EHR systems, where each patient's EHR files are encrypted using a broadcast variant of CP-ABE [16] that allows direct revocation. However, the cipher text length grows linearly with the number of unrevoked users.

1.2 Revocable ABE

It is a well-known challenging problem to revoke users/attributes efficiently and on-demand in ABE. Traditionally, this is often done by the authority broadcasting periodic key Updates to unrevoked users frequently [13], [22], which does not achieve complete backward/forward security and is less efficient. Recently, [23] and [24] proposed two CP-ABE schemes with immediate attribute revocation capability, instead of periodical revocation. However, they were not designed for MA-ABE. In addition, Raj et al. [25] proposed an alternative solution for the same problem in our paper using Lawks and Waters'(LW)decentralized ABE scheme [26]. The main advantage of their solution is, each user can obtain secret keys from any subset of the TAs in the system, in contrast to the CC MAABE. The LW ABE scheme enjoys better policy expressiveness, and it is extended by [25] to support user revocation. On the downside, the communication overhead of key revocation is still high, as it requires a data owner to transmit an updated cipher text component to every no revoked user. They also do not differentiate personal and public domains. In this paper, we bridge the above gaps by proposing a unified security framework for patient-centric sharing of PHRs in a multi domain, multi authority PHR system with many users.

1.3 Security Model

In this paper, we consider the server to be semi trusted, i.e., honest but curious as those in [28] and [15]. That means the server will try to find out as much secret information in the stored PHR files as possible, but they will honestly follow the protocol in general. On the other hand, some users will also try to access the files beyond their privileges. For example, a pharmacy may want to obtain the prescriptions of patients for marketing and boosting its profits. To do so, they may collude with other users, or even with the server. In addition, we assume each party in our system is preloaded with a public/private key pair, and entity authentication can be done by traditional Challenge-response protocols.

1.4 Enhanced Key-Policy Generation Rule

In addition to the basic key-policy generation rule, the attribute tuples assigned by the same AA for different users do no intersect with each other, as long as their primary attribute types are distinct.

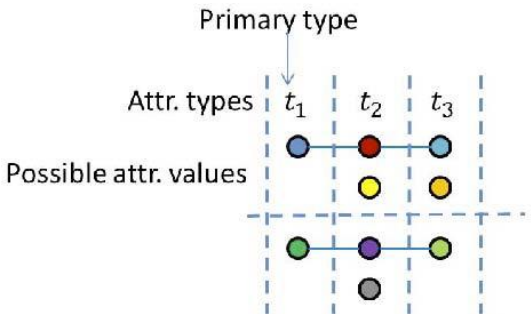


Fig. 1.4. Illustration of the enhanced key-policy generation rule, Solid horizontal lines represents possible attribute associations for two users

1.5 (Enhanced Encryption Rule)

In addition to the basic encryption rule, as long as there are multiple attributes of the same primary type, corresponding non intersected attribute tuples are included in the cipher text's attribute set.

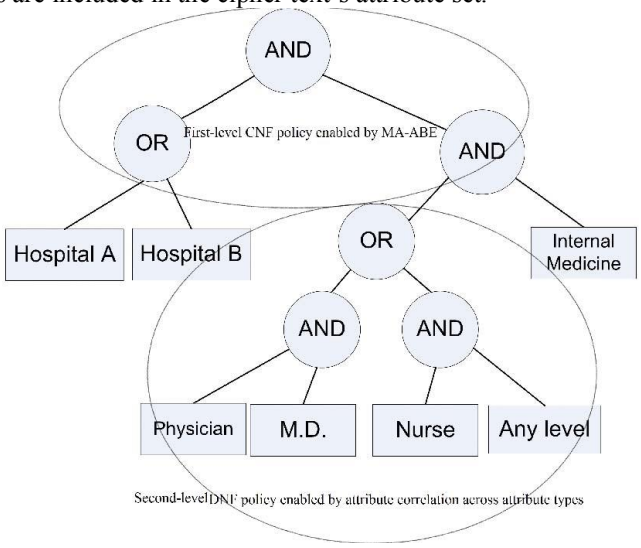


Fig 1.5 An example policy realizable under our framework using MAABE, following the enhanced key generation and encryption rules

1.6 Enhancing MA-ABE for User Revocation

The original CC MA-ABE scheme does not enable efficient and on-demand user revocation. To achieve this for MAABE we combine ideas from YWRL's revocable KP-ABE [9],[15] (its details are shown in supplementary material, available online), and propose an enhanced MA-ABE scheme. In particular, an authority can revoke a user or user's attributes immediately by encrypting the cipher texts and updating users' secret keys, while a major part of these operations can be delegated to the server which enhances efficiency. The idea to revoke one attribute of a user in MA-ABE is as follows: The AA who governs this attribute actively updates that attribute for all the affected unrevoked users.

Vol.2 Issue. 3, March- 2014, pg. 35-41 **ISSN: 2321-8363**

1) The public/master key components for the affected attribute
2) The secret key component corresponding to that attribute of each unrevoked user
3) Also, the server shall update all the cipher texts containing that attribute.

In order to reduce the potential computational burden for theAAs, we adopt proxy encryption to delegate operations 2 and 3 to the server, and use lazy-revocation to reduce the Overhead. In particular, each data attribute i is associated With a version number very. Upon each revocation event, if is an affected attribute, the AA submits a rekey rki$i0 ¼ t0 Is=it to the server, who then encrypts the affected cipher texts And increases their version numbers. The unrevoked users' Secret key components are updated via a similar operation Using the rekey. To delegate secret key updates to the Server, a dummy attribute needs to be additionally defined By each of N _ 1 AAs, which are always ANDed with each User's key-policy to prevent the server from grasping the Secret keys. This also maintains the resistance against up to N _ 2 AA collusion of MA-ABE (as will be shown by our Security proof). Using lazy-revocation, the affected cipher texts And user secret keys are only updated when an Affected unrevoked user logs into the system next time. By the form of the rekey, all the updates can be aggregated from the last login to the most current one.To revoke a user in MA-ABE, one needs to find out a minimal subset of attributes (_) such that without it the user's secret key's access structure (AAu) .

1.7 Enforce Write Access Control

For certain parts of the PHR data, medical staffs need to have temporary access when an emergency happens to a patient, who may become unconscious and is unable to change her access policies beforehand. The medical staffs will need some temporary authorization (e.g., emergency key) to decrypt those data. Under our framework, this can be naturally achieved by letting each patient delegate her emergency key to an emergency department. Specifically, In the beginning, each owner defines an "emergency" Attribute and builds it into the PSD part of the cipher text of Each PHR document that she allows break-glass access. She Then generates an emergency key skEM using the single node Key-policy "emergency," and delegates it to the ED Who keeps it in a database of patient directory? Upon Emergency, a medical staff authenticates her to the ED, Requests and obtains the corresponding patient's skEM, and then decrypts the PHR documents using skEM. After the Patient recovers from the emergency, she can revoke the Break-glass access by computing a rekey: rkEM, submit it to The ED and the server to update her skEM and CT to their Newest versions, respectively.

1.8 Handle Dynamic Policy Changes

Our scheme should support the dynamic add/modify/delete of part of the document access policies or data Attributes by the owner. For example, if a patient does not want doctors to view her PHR after she finishes a visit to a hospital, she can simply delete the cipher text components corresponding to attribute "doctor" in her PHR files. Adding and modification of attributes/access policies can be done by proxy re-encryption techniques [22]; however, they are expensive. To make the computation more efficient, each owner could store the random number s used in encrypting the FEK3 of each document on her own computer, and construct new cipher text components corresponding to added/changed attributes based on s.PHR encryption and access. The owners upload ABE encrypted PHR files to the server (3). Each owner's PHR file is encrypted both under a certain fine-grained and role based access policy for users from the PUD to access and under a selected set of data attributes that allows access from users in the PSD. Only authorized users can decrypt the PHR files, excluding the server. For improving efficiency, the data attributes will include all the intermediate file types from a leaf node to the root. For example, in Fig. 2, an "allergy" file's attributes are f PHR; medical history; allergy. The data readers download PHR files from the server, and they can decrypt the files only if they have suitable attribute-based keys (5). The data contributors will be granted write access to someone's PHR, if they present proper write keys (4).User revocation. Here, we consider revocation of a data User revocation. Here, we consider revocation of a data reader or her attributes/access privileges.

Vol.2 Issue. 3, March- 2014, pg. 35-41 **ISSN: 2321-8363**

There are several possible cases:
1. Revocation of one or more role attributes of a public Domain user;
2. Revocation of a public domain user which is equivalent to revoking all of that user's attributes. These operations are done by the AA that the user belongs to, where the actual computations can be delegated to the server to improve efficiency (8).
3. Revocation of a personal domain user's access Privileges;
4. Revocation of a personal domain user. These can be initiated through the PHR owner's client application in a similar way.
Policy updates. A PHR owner can update her sharing.

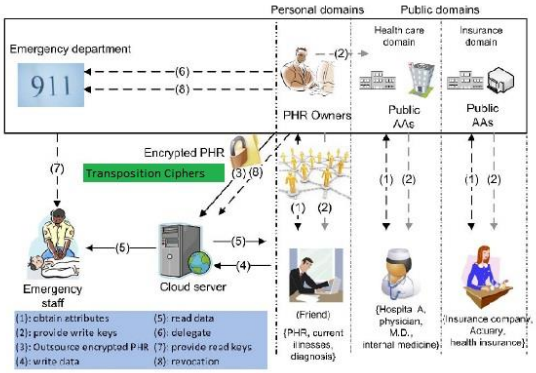


Fig. PHR Architecture

2. RELATED WORK

This paper is mostly related to works in cryptographically enforced access control for outsourced data and attribute based encryption. To realize fine-grained access control, the traditional public key encryption (PKE)-based schemes [8],[10] either incur high key management overhead, or require encrypting multiple copies of a file using different users' keys. To improve upon the scalability of the above solutions, one-to-many encryption methods such as ABE can be used .In Goya let al.'s seminal paper on ABE [11], data are encrypted under a set of attributes so that multiple users who possess proper keys can decrypt. This potentially makes encryption and key management more efficient [12]. A fundamental property of ABE is preventing against user collusion. In addition, the encrypt or is not required to know the ACL.

3. PROPOSED WORK

- Secure sharing of PHRs
- Tried on semi-trusted servers.
- In order to protect the personal health data stored on a semi-trusted server.
- We adopt Transposition Cipher (TPC) as the main encryption primitive.
- Using TPC, access policies are expressed based on the positions based users or data.
- Which enables a patient to selectively share her PHR among a set of users by encrypting the file under a set of positions based?
- The complexities per encryption, key generation and decryption are only linear with the number of positions based involved.
- To integrate TPC into a large-scale PHR system,

- key management scalability
- dynamic policy updates
- Efficient on-demand revocations are non-trivial to solve.

3.1 Security Model

In this paper, we consider the server to be semi trusted, i.e., honest but curious as those in [28] and [15]. That means the server will try to find out as much secret information in the stored PHR files as possible, but they will honestly follow the protocol in general. On the other hand, some users will also try to access the files beyond their privileges. For example, a pharmacy may want to obtain the prescriptions of patients for marketing and boosting its profits. To do so, they may collude with other users, or even with the server. In addition, we assume each party in our system is preloaded with a public/private key pair, and entity authentication can be done by traditional challenge-response protocols.

3.2 Requirements

To achieve "patient-centric" PHR sharing, a core requirement is that each patient can control who are authorized to access to her own PHR documents. Especially, user-controlled Read/write access and revocation are the two core security objectives for any electronic health record system, pointed out by Mandela al. [7] in as early as 2001. The security and performance requirements are summarized as follows Data confidentiality. Unauthorized users (including the server) who do not possess enough attributes, satisfying the access policy or do not have proper key access privileges. Whenever a user's attribute is no longer valid, the user should not be able to access future PHR files using that attribute. This is usually called attribute revocation, and the corresponding security property is forward secrecy [23]. The data access policies should be flexible, i.e., dynamic changes to the predefined policies shall bellowed, especially the PHRs should be accessible under emergency scenarios. Scalability, efficiency, and usability. The PHR system should support users from both the personal domain and public domains. Since the set of users from the public domain may be large in size and unpredictable, the system should be highly scalable, in terms of complexity in key management, communication, computation and storage. Additionally, the owners' efforts in managing users and keys should be minimized to enjoy usability.

3.3 Overview of Our Framework

The main goal of our framework is to provide secure patient-centric PHR access and efficient key management at the same time. The key idea is to divide the system into multiple security domains (namely, public domains and personal domains) according to the different users' data access requirements. The PUDs consist of users who make access based on their professional roles, such as doctors, nurses, and medical researchers. In practice, a PUD can be mapped to an independent sector in the society, such as the health care, government, or insurance sector. Especially, in a PUD multi authority ABE is used, in which there are multiple "attribute authorities" (AAs), each governing a disjoint subset of attributes. Role attributes are defined for PUDs, representing the professional role or obligations of a PUD user. Users in PUDs obtain their attribute-based secret keys from the AAs, without directly interacting with the owners. To control access from PUD users, owners are free to specify role-based fine-grained access policies for her PHR files, while do not need to know The list of authorized users when doing encryption. Since the PUDs contain the majority of users, it greatly reduces the key management overhead for both the owners and users. Each data owner (e.g., patient) is a trusted authority of her own PSD, who uses a KP-ABE system to manage the secret keys and access rights of users in her PSD. Since the users are personally known by the PHR owner, to realize patient-centric-access, the owner is at the best position to grant user access privileges on a case-by-case basis.

Vol.2 Issue. 3, March- 2014, pg. 35-41 **ISSN: 2321-8363**

Encryption

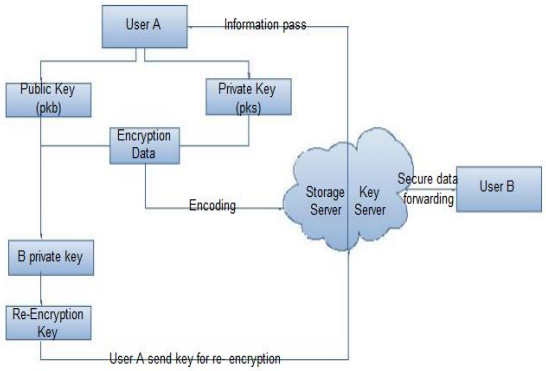
C = E(D(E(P,K1),K2),K3)

Decryption

P = D(E(D(C,K3),K2),K1)

TRANSPOSITION CIPHERS

Transposition Cipher permutes symbols in a block of symbols



.

Fig:3,3 Encryption diagram

4. CONCLUSION

In this paper, we have proposed a novel framework of secure sharing of personal health records in cloud computing. Considering partially trustworthy cloud servers, we argue that to fully realize the patient-centric concept, patients shall have complete control of their own privacy through encrypting their PHR files to allow fine-grained access. The framework addresses the unique challenges brought by multiple PHR owners and users, in that we greatly reduce the complexity of key management while enhance the privacy guarantees compared with previous patients can allow access not only by personal users, but also various users from public domains with different professional roles, qualifications, and affiliations. Furthermore, works. We utilize ABE to encrypt the PHR data, so that we enhance an existing MA-ABE scheme to handle efficient and on-demand user revocation, and prove its security. Through implementation and simulation, we show that our solution is both scalable and efficient.

REFERENCES

[1]. M. Li, S. Yu, K. Ran, and W. Lou, "Securing Personal Health Records in Cloud Computing: Patient-Centric and Fine-Grained Data Access Control in Multi-Owner Settings," Proc. Sixth Int'l ICST Conf. Security and Privacy in Comm. Networks (SecureComm '10), pp. 89-106, Sept. 2010.

[2] H. Lo¨ hr, A.-R. Sadeghi, and M. Wigand, "Securing the E-Health Cloud," Proc. First ACM Int'l Health Informatics Sump. (IHI '10), pp. 220-229, 2010.

[3] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized Private Keyword Search over Encrypted Personal Health Records in Cloud Computing," Proc. 31st Int'l Conf. Distributed Computing Systems (ICDCS '11), June 2011.

[4] "The Health Insurance Portability and Accountability Act,"
http://www.cms.hhs.gov/HIPAAGenInfo/01_Overview.asp, 2012.

[5] "Google, Microsoft Say Hipaa Stimulus Rule Doesn't Apply to Them,"
http://www.ihealthbeat.org/Articles/2009/4/8/, 2012.

