Journal Of Harmonized Research (JOHR)

Journal Of Harmonized Research in Engineering
2(1), 2014, 239-245

ISSN 2347 – 7393

Original Research Article

SCALABLE AND SECURE SHARING OF PERSONAL HEALTH RECORDS IN CLOUD COMPUTING USING ATTRIBUTE-BASED ENCRYPTION

Batta Srikanth Varma, B.Venkateswara Reddy

K.I.T.S., Divili

Abstract- Cloud computing servers provides promising platform for storage of data. Sharing of personal medical records is an emerging patient centric model of health information exchange, which is often outsourced to store at third party, such as cloud providers. The confidentiality of the medical records is major problem when patients use commercial cloud servers to store their medical records because it can be view by everyone, to assure the patients' control over access to their own medical records; it is a promising method to encrypt the files before outsourcing and access control should be enforced though cryptography instead of role based access control. There are various other 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. To achieve fine grained and scalable data access control for medical records stored in semi trusted servers, we leverage attribute based encryption (ABE) techniques to encrypt each patient's medical record file. In this paper, we describe a new approach which enables secure storage and controlled sharing of patient's health data. We explore key- policy attribute based encryption and multi-authority attribute based encryption to enforce patient access control policy such that everyone can download the data ,but only authorize user can view the medical records. This project also supports multiple owner scenarios and divides the users in the system into multiple security domains that greatly reduce the key management complexity for owners and users. A high degree of patient privacy is guaranteed by exploiting multi-authority ABE. In this paper we presents the detail design of modules and implementation Packages of the proposed framework.

Index Terms- Multi-Authority Attribute Based Encryption, Key-Policy Attribute Based Encryption, Secure Sharing.

For Correspondence:
srivarma505ATgmail.com
Received on: March 2014
Accepted after revision: March 2014
Downloaded from: www.johronline.com

1. Introduction

Patient centric medical records information exchange is model for the sharing of medical records, which allows patient to create, manage and control his/her medical information in

centralized place through the web or cloud. Patient can now share his/her medical records effectively with a wide range of users such as family members, friends and doctors. Cloud Computing made lots of attraction, because of there is provision of storage as service and software as service, by which software service providers can enjoy the virtually infinite and elastic storage and computing resources. As such, the providers are more and more willing to shift their storage and application services into the cloud like Microsoft and Amazon, instead of building specialized data centres, in order to lower their operational cost .While it is exciting to have these services in the cloud for everyone, there are many security and privacy risks which could impede its wide adoption. The main concern is about the privacy of patients' personal health data and who could gain access to the medical records when they are stored in a cloud server. Since patients lose physical control to their own personal health data, directly placing those sensitive data under the control of the servers cannot provide strong privacy assurance at all. While going for cloud computing storage, the data owner and cloud servers are in two different domains. On one hand, cloud servers are not entitled to access the outsourced data content for data confidentiality; on the other hand, the data resources are not physically under the full control of data owner. Storing personal medical records on the cloud server leads to need of Encryption mechanism to protect the medical health record, before outsourcing to the cloud. To deal with the potential risks of privacy exposure, instead of letting the service providers encrypt patients' data, medical records sharing services should give patients (patient / medical record owners) full control over the selective sharing of their own medical data. To this end, the medical records should be encrypted in addition to traditional access control mechanisms provided by the server. We use Java Paring Based Cryptography library (JPBC) for the implementation of KP-ABE and MA-ABE. In this paper, we discussed the design and Implementation detail for the of the proposed framework.

1. Attribute Based Encryption

The concept of ABE was introduced along with another cryptography called fuzzy identity-based encryption (FIBE) by Sahai and Waters. Both schemes are based on bilinear maps (pairing). In ABE system, users' private keys and cipher text are labelled with sets of descriptive attributes and access policies respectively, and a particular key can decrypt a particular cipher text only if associated attributes and policy are matched.

A. Key-Policy Attribute-Based Encryption

The key-policy attribute-based encryption (KP-ABE) was first introduced in 2006 by Goyal et al. [2] In this cryptography system, cipher text are labelled with sets of attributes. Private keys, on the other hand, are associated with access structures A. A private key can only decrypt a cipher text whose attributes set is authorized set of the private key's access structure. KP-ABE is a cryptography system built upon bilinear map and Linear Secret Sharing Schemes.

B. Multi-Authority attribute-Based encryption

In a multi-authority ABE system, we have many attribute authorities, and many users. There are also a set of system wide public parameters available to everyone (either created by a distributed protocol between the authorities). A user can choose to go to an attribute authority, prove that it is entitled to some of the attributes handled by that authority, and request the corresponding decryption keys. The authority will run the attribute key generation algorithm, and return the result to the user. Any party can also choose to encrypt a message, in which case he uses the public parameters together with an attribute set of his choice to form the cipher text. Any user who has decryption keys corresponding to an appropriate attribute set can use them for decryption.

2. Related Work

For access control of outsourced data, partially trusted servers are often assumed. With cryptographic techniques, the goal is trying to enforce who has (read) access to which parts of a patient's PHR documents in a fine-grained way.

A. Symmetric key cryptography (SKC) based solutions

Symmetric-key algorithms are a class of algorithms for cryptography that use the same cryptographic keys for both encryption of plaintext and decryption of cipher text. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link Vimercati et.al Proposed a solution for securing outsourced data on semi-trusted servers based on symmetric key derivation methods, which can achieve fine-grained access control. Unfortunately, the complexities of file creation and user grant/revocation operations are linear to the number of authorized users, which is less scalable.

B. Public key cryptography (PKC) based solutions

PKC based solutions were proposed due to its ability to separate write and read privileges. To realize fine-grained access control, the traditional public key encryption (PKE) based schemes proposed by J. Benaloh, M. Chase, E. Horvitz, and K. Lauter in their work "Patient controlled encryption: ensuring privacy of electronic medical records" ,they purpose the solution scenario and shows how public and symmetric based encryption used , disadvantage of their solution is either incur high key management overhead, or require encrypting multiple copies of a file using different users' keys.

C. Attribute Based Encryption based solutions

A number of works used ABE to realize fine-grained access control for outsourced data, especially, there has been an increasing interest in applying ABE to secure electronic healthcare records (EHRs). Narayan et al. proposed an attribute-based infrastructure for EHR systems, where each patient's EHR files are encrypted using a broadcast variant of Cipher Text-ABE (CP-ABE)[4] .However, the cipher text length grows linearly with the number of unrevoked users. In a variant of ABE that allows delegation of access rights is proposed for encrypted EHRs. Ibraimi et.al. Applied cipher text policy ABE (CP-ABE) to manage the sharing of PHRs, and introduced the concept of social/professional domains but they do not use multi-authority ABE. In Akinyele et al. investigated using ABE to generate self-protecting EMRs, which can either be stored on cloud servers or cell phones so that EMR could be accessed when the health provider is offline. Drawback is device dependency and revocation is not supported. Other Common drawback of all above solutions is problem of key-escrow as they consider single trusted authority.

3 Framework for patient-centric, secure and scalable PHR sharing

In this section, we describe our novel patient-centric secure data sharing framework for cloud-based PHR systems.

We consider a PHR system where there are multiple PHR owners and PHR users. The owners refer to patients who have full control over their own PHR data, i.e., they can create, manage and delete it. There is a central server belonging to the PHR service provider that stores all the owners' PHRs. The users may come from various aspects; for example, a friend, a caregiver or a researcher. Users access the PHR documents through the server in order to read or write to someone's PHR, and a user can simultaneously have access to multiple owners' data.

A typical PHR system uses standard data formats. For example, continuity-of-care (CCR) (based on XML data structure), which is widely used in representative PHR systems including Indivo, an open-source PHR system adopted by Boston Children's Hospital. Due to the nature of XML, the PHR files are logically organized by their categories in a hierarchical way.

3.1.1 Security Model

In this paper, we consider the server to be semi-trusted. 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.

3.1.2 Requirements

To achieve "patient-centric" PHR sharing, a core require- ment 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 Mandl 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 should be prevented from decrypting a PHR document, even under user collusion. Fine-grained access control should be enforced meaning different users are authorized to read different sets of documents.
- On-demand revocation. 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]. There is also user revocation, where all of a user's access privileges are revoked.
- Write access control: We shall prevent the unauthorized contributors to gain write-access to owners' PHRs, while the legitimate contributors should access the server with accountability.
- Scalability, efficiency and usability: The PHR system should support users from both the personal do- main 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.

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 (PUDs) and personal domains (PSDs)) 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. For each PSD, its users are personally associated with a data owner (such as family members or close friends), and they make accesses to PHRs based on access rights assigned by the owner.

A. System Setup and Key Distribution: The system first defines a common universe of data attributes shared by every PSD, such as "basic profile", "medical his- tory", "allergies", and "prescriptions". An emergency attribute is also defined for break-glass access. Each PHR owner 's client application generates its corresponding public/master keys. The public keys can be published via user 's profile in an online healthcare social-network (HSN) (which could be part of the PHR service; e.g., the Indivo system [27]). There are two ways for distributing secret keys. First, when first using the PHR service, a PHR owner can specify the access privilege of a data reader in her PSD, and let her application generate and distribute corresponding key to the latter, in a way resembling invitations

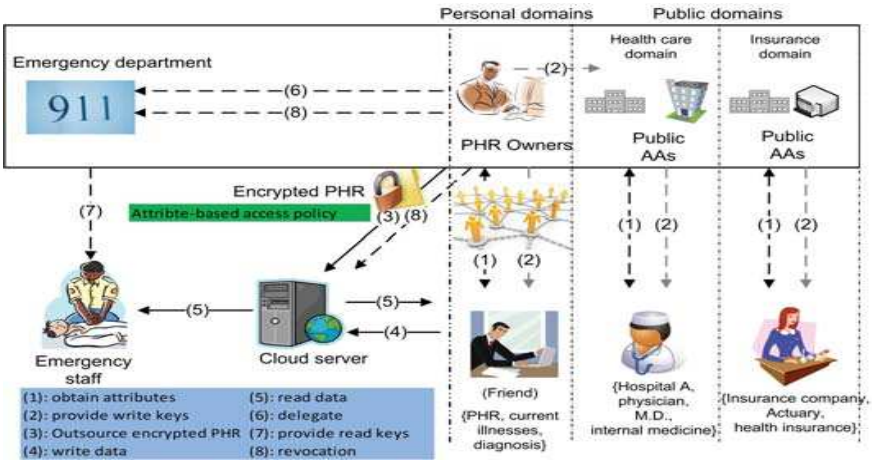
in GoogleDoc. Second, a reader in PSD could obtain the secret key by sending a request (indicating which types of files she wants to access) to the PHR owner via HSN, and the owner will grant her a subset of requested data types. Based on that, the policy engine of the application automatically derives an access structure, and runs keygen of KP-ABE to generate the user secret key that embeds her access structure. In addition, the data attributes can be organized in a hierarchical manner for efficient policy generation, see Fig. 2. When the user is granted all the file types under a category, her access privilege will be represented by that category instead.

B. PHR Encryption and Access: The owners upload ABE-encrypted PHR files to the server. 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.

C. User Revocation: Here we consider revocation of a data reader or her attributes/access privileges. 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.

D. Policy Updates: A PHR owner can update her sharing policy for an existing PHR document by updating the attributes (or access policy) in the cipher text. The supported operations include add/delete/modify, which can be done by the server on behalf of the user.

E. Break glass: When an emergency happens, the regular access policies may no longer be applicable. To handle this situation, break-glass access is needed to access the victim's PHR. In our framework, each owner's PHR's access right is also delegated to an emergency department (ED, (6)). To prevent from abuse of break-glass option, the emergency staff needs to contact the ED to verify her identity and the emergency situation, and obtain temporary read keys ((7)). After the emergency is over, the patient can revoke the emergent access via the ED.



5. System Implementation and algorithms

This stage focuses on specific tools such as programming languages, libraries and components which allow to Quickly producing software of high quality. Implementation is the stage of the project when the theoretical Design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a Successful new system and in giving the user, confidence that the new system will work and be effective.

Algorithms of for KP-ABE with enhancement are discussed as below:

1) KP-ABE Setup (A): Outputs public key PK and Master key MK for A as set of attributes

- Associate for each attribute in A with attributes universe as U = {1, 2, ..., n}.
- It defines a bilinear group G1 of prime order p with a generator g, a bilinear map e : G1 × G1 -> G2 which has the properties of bi linearity, computability, and non-degeneracy.

-Associate each attribute i ∈ U with a number ti and also chose y uniformly at random in Zp* and y.

-The public key is:

PK = (T1 = gt1, T1 = gt|U | , Y = e(g, g)y)

-The master key is:

MK = (t1,…, t|U |, y)

2) KP-ABE Encryption (M, γ, PK) :M message in GT with a set of attributes γ , PK is public Key , outputs Cipher Text E.

-Choose a random value s in Zp. Encrypt a secret message M in GT with a set of attributes γ.

-The cipher text is:

E = (γ, E' = MY s, {Ei = Ti s} where iε γ)

3) KP-ABE Key Generation (A, MK) :

This algorithm output a secret key D embedded with an access structure T. The access structure A is realized by the following three steps:

1. For root node r, set value secret = y. mark all node un-assigned and mark root node assigned.
2. Recursively, for each assigned non-leaf node,

a. If the operator is ^ (and) and its child nodes are marked un-assigned, let n be the number of child nodes, set

the value of each child node, except the last one, to be si ε Zp, and the value of the last node to be sn = s -Σ si

.Mark this node assigned.

b. If the operator is V (or), set the values of its child nodes to be s. Mark this node assigned.

3. For each leaf attribute aj,i ε T , compute Dj;i = Tj∧si

Output: Secret Key Sk= { Dj,i}

4) KP-ABE Decryption (E, D) This algorithm takes as input the cipher text E encrypted under the attribute set U,

the user's secret key SK for access tree T, and the public key PK. Finally it output the message M if and only if U satisfies T.

Basic Algorithm of the MA-ABE with enhancement is:

1) Key Issue (Attributes, MK, PK). This algorithm, the AAs collectively actively generates a secret key for a user. For

a user with (secret) ID *u*, the secret key is in the form :

SKu = ⟨Du = g^Ru ,, {Dk;i = g^(qk(i)/tk,i) , Verk,i} k∈{1…N} ⟩ where *Ru* is a global ID for user *u*, and

qk(0) = Σ*k vk* − *u*.

2) Encryption (M, PK, attributes []): This algorithm takes a message *M*, *PK* and a set of attributes and outputs the

Cipher text *E* as follows:

The encryptor first chooses an *s* ∈ Z*p*, and then returns:

CT = [E0 = M·Y^ s, E1 = g2^s, {Ck;i = Tk;i^s, Verk;i};k∈{1..N}]

Where i= no of attributes form authority k

3) Decryption (*CT, SKu*): This algorithm takes as input a cipher text CT and a user secret key *SKu*. If for each AA *k*,

If the version of attribute in SK and CT matches, algorithm pairs up *Dk;i* and *Ck;i* and reconstructs *e*(*g*1, *g*2)^*sqk*(0).

After multiplying all these values together with *e*(*Du*,*E*1), *u* recovers the blind factor *Y* ^ *s* and thus gets *M*.

4) Update Parameter: This algorithm updates an attribute to a new version by redefining its system master key and public key component. It also outputs a proxy re-encryption key and re-secret-key between the old version and the new version of the attribute.

5) Update Secret Key: This algorithm translates the secret key component of attribute i in the user secret key SK from an old version into the

latest version using re-secret-key generated in step 4.

6) Re-Encrypt File: This algorithm translates the ciphertext component of an attribute i of a file from an old version into the latest version using proxy- encryption key generated in step 4.

3) *Main Packages and Systems:*

a. *Package kpabe :* Implements Key-Policy Attribute-Based Encryption Algorithms such as Setup, Encryption,

Key Generation, and Decryption, Policy Generation algorithms, policy update. They are the most important parts during KP-ABE construction.

b. *Package Multi Authority: Implements* Multi-Authority Attribute-Based Encryption Algorithms such as Setup:

Implements Setup, Encryption, Key Generation, and Decryption, Policy Generation algorithms, policy update. Update Parameter, Regenerate secret key and re-encrypt the files.

c. *Package Abe Encrypt:* Implements the Encrypt – which encrypt the file under both data attribute under key-policy ABE and Multi Authority-Based Encryption. Main functions are Decrypt Medical Record Private,

Decrypt Medical Record Public and Encrypt.

d. *Package Abe Web Service:* Provides the interface with above packages, which can be used in any platform to generate GUI using Web Service. Using this package , it makes easier to use API for any platform.

e. *PMRS System:* Allows user to register with system and encrypt medical records, upload medical records, view medical records using above packages.

6. Conclusion

In this Paper, we have presented the detail design and implementation detail of proposed a novel framework of secure sharing of personal medical 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 medical record files to allow fine-grained access. The framework addresses the unique challenges brought by multiple owners and users, in that we greatly reduce the complexity of key management while ensured the privacy. We utilize various forms of ABE to encrypt the medical record files, so that patients can allow access not only by personal users, but also various users from public domains with different professional roles, qualifications and affiliations.

7. Reference

[1].M. Li, S. Yu, K. Ren, and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in SecureComm‟10, Sept.2010, pp. 89–106.

[2]H. L¨ohr, A.-R. Sadeghi, and M. Winandy, "Securing the e-health cloud," in Proceedings of the 1st ACM International Health Informatics Symposium, ser. IHI ‟10, 2010, pp. 220–229.

[3]M. Chase and S. S. Chow, "Improving privacy and security in ulti-authority attribute-based encryption," in CCS '09, 2009, pp.121–130.

[4]S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in ASSIACCS‟10, 2010.

[5]S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in IEEE INFOCOM‟10, 2010.