Secure Sharing For Personal Medical Record in Cloud Computing

C. Saranya[1], Dr. M. Chidambaram[2]
[1, 2] Department of Computer Science
[1, 2] Raja Serfoji Government College (Autonomous), Thanjavur

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.*

Keywords- Personal health record, Secure Sharing PHR, PHR Services, Features of secure sharing.

I. INTRODUCTION

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 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 specialized data centers, many PHR services are outsourced to or provided by third-party service providers, for example, Microsoft HealthVault1.While it is exciting to have convenient PHR services for everyone, there are many security and privacy risks which could impede its wide adoption.

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 hand, 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 personal health information (PHI), the third-party storage servers are often the targets of various malicious behaviors which may lead to expo- sure 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]. To ensure patient-centric privacy control over their own PHRs, it is essential to have fine-grained data access control mechanisms that work with semi-trusted servers.

A feasible and promising approach would be to en-crypt the data before outsourcing. Basically, the PHR owner herself should decide how to encrypt her files and to allow which set of users to obtain access to each file. A PHR file should only be available to the users who are given the corresponding decryption key, while remain confidential to the rest of users. Furthermore, the patient shall always retain the right to not only grant, but also revoke access privileges when they feel it is necessary [7].

(1) We propose a novel ABE-based framework for patient-centric secure sharing of PHRs in cloud comput- ing environments, under the multi-owner settings. To ad- dress the key management challenges, we conceptually divide the users in the system into two types of domains, namely *public* and *personal domains.* In particular, the ma- jority professional users are managed distributively by attribute authorities in the former, while each owner only needs to manage the keys of a small number of users in her personal domain. In this way, our framework can simultaneously handle different types of PHR sharing .control, handles dynamic policy updates, and provides break-glass access to PHRs under emergence scenarios.

(2) In the public domain, we use multi-authority ABE (MA-ABE) to improve the security and avoid key escrow problem. Each attribute authority (AA) in it governs a disjoint subset of user role attributes, while none of them alone is able to control the security of the whole system. We propose mechanisms for key distribution and encryption so that PHR owners can specify per- sonalized fine-grained role-based access policies during file encryption. In the personal domain, owners directly assign access privileges for personal

users and encrypt a PHR file under its data attributes. Furthermore, we enhance MA-ABE by putting forward an efficient and on-demand user/attribute revocation scheme, and prove its security under standard security assumptions. In this way, patients have full privacy control over their PHRs.

(3) We provide a thorough analysis of the complexity and scalability of our proposed secure PHR sharing solution, in terms of multiple metrics in computation, communication, storage and key management. We also compare our scheme to several previous ones in complexity, scalability and security. Furthermore, we demonstrate the efficiency of our scheme by imple- menting it on a modern workstation and performing experiments/simulations.

II. THE SECURITY AND PERFORMANCE REQUIREMENTS

- *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.

- The data access policies should be flexible, i.e.,dynamic changes to the predefined policies shall be allowed, specially the PHRs should be accessible under emergency scenarios.

- *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.

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 con- sist 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.

In both types of security domains, we utilize ABE to realize cryptographically enforced, patient-centric PHR access. 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. For PSD, *data attributes* are defined which refer to the intrinsic properties of the PHR data, such as the category of a PHR file. For the purpose of PSD access, each PHR file is labeled with its data attributes, while the key size is only linear with the number of file categories a user can access. Since the number of users in a PSD is often small, it reduces the burden for the owner. When encrypting the data for PSD, all that the owner needs to know is the intrinsic data properties.

The multi-domain approach best models different user types and access requirements in a PHR system. The use of ABE makes the encrypted PHRs self-protective, i.e., they

can be accessed by only authorized users even when storing on a semi-trusted server, and when the owner is not online. In addition, efficient and on-demand user revocation is made possible via our ABE enhancements.

PHR Encryption and Access. The owners upload ABE-encrypted PHR files to the server ((3)). Each own- er'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□□*P HR, 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 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.

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

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 ac- cess 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.

MAIN DESIGN ISSUES

In this section, we address several key design issues in secure and scalable sharing of PHRs in cloud computing, under the proposed framework.

Using MA-ABE in the Public Domain

For the PUDs, our framework delegates the key manage- ment functions to multiple attribute authorities. In order to achieve stronger privacy guarantee for data owners, the Chase-Chow (CC) MA-ABE scheme [21] is used, where each authority governs a disjoint set of attributes distributively. It is natural to associate the ciphertext of a PHR document with an owner-specified access policy for users from PUD. However, one technical challenge is that CC MA-ABE is essentially a KP-ABE scheme, where the access policies are enforced in users' secret keys, and those key-policies do not directly translate to document access policies from the owners' points of view. By our design, we show that by agreeing upon the formats of the key-policies and the rules of specifying which attributes are required in the ciphertext, the CC MA-ABE can actually support owner-specified document access policies with some degree of flexibility (such as the one in Fig. 4), i.e., it functions similar to CP-ABE2.

In order to allow the owners to specify an access policy for each PHR document, we exploit the fact that the basic CC MA-ABE works in a way similar to fuzzy-IBE, where the threshold policies (e.g., k out of n) are supported. Since the threshold gate has an intrinsic symmetry from both the encryptor and the user's point of views, we can pre-define the formats of the allowed document policies as well as those of the key-policies, so that an owner can enforce a file access policy through choosing which set of attributes to be included in the ciphertext.

By enhancing the key-policy generation rule, we can enable more expressive encryptor's access policies. We exploit an observation that in practice, a user's at-tributes/roles belonging to different types assigned by the same AA are often *correlated* with respect to a *primary* attribute type. In the following, an attribute tuple refers to the set of attribute values governed by one AA (each of a different type) that are correlated with each other.

Definition 5 (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 not intersect with each other, as long as their primary attribute types are distinct.

Definition 6 (Enhanced Encryption Rule): In addition to the basic encryption rule, as long as there are multiple attributes

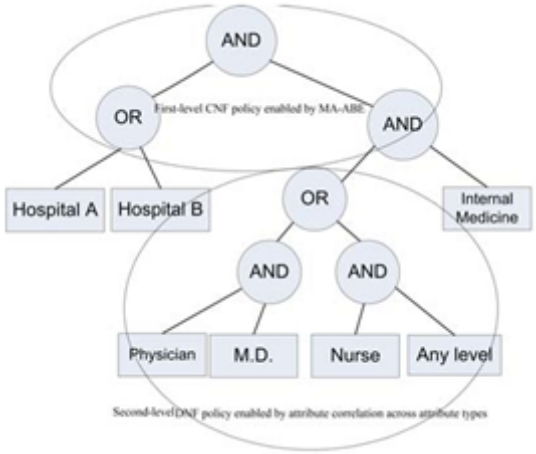
of the same primary type, corresponding non- intersected attribute tuples are included in the ciphertex- t's attribute set.

This primary-type based attribute association is illus- trated in Fig. 3. Note that there is a "horizontal asso- ciation" between two attributes belonging to different types assigned to each user. For example, in the first AA (AMA) in Table 2, "license status" is associated with "profession", and "profession" is a primary type. That means, a physician's possible set of license status do not intersect with that of a nurse's, or a pharmacist's. An "M.D." license is always associated with "physician", while "elderly's nursing licence" is always associated with "nurse". Thus, if the second level key policy within the AMA is "1 out of n_1 ∧ 1 out of n_2", a physician would receive a key like "(physician OR *) AND (M.D. OR *)" (recall the assumption that each user can only hold at most one role attribute in each type), nurse's will be like "(nurse OR *) AND (elderly's nursing licence OR *)". Meanwhile, the encryptor can be made aware of this correlation, so she may include the attribute set:

{physician, M.D., nurse, elderly's nursing licence} during encryption. Due to the attribute correlation, the set of users that can have access to this file can only possess one out of two sets of possible roles, which means the following policy is enforced: "(physician AND M.D.) OR (nurse AND elderly's nursing licence)". The direct consequence is it enables a *disjunctive normal form* (DNF) encryptor access policy to appear at the second level. If the encryptor wants to enforce such a DNF policy under an AA, she can simply include all the attributes in that policy in the ciphertext.

Furthermore, if one wants to encrypt with wildcard attributes in the policy, say: "(physician AND M.D.) OR (nurse AND any nursing license)" the same idea can be used, i.e., we can simply correlate each "profession" attribute with its proprietary "*" attribute. So we will have "$*_{\text{nursing license}}$, $*_{\text{physician license}}$" etc. in the users' keys. The above discussion is summarized in by an example encryptor's policy.

If there are multiple PUDs, then $\Pi = \cup_{PUDj} \{\Pi_{PUDj}\}$, and multiple sets of ciphertext components needs to be included. Since in reality, the number of PUDs is usually small, this method is more efficient and secure than a straightforward application of CP-ABE in which each organization acts as an authority that governs all types of attributes [1], and the length of ciphertext grows linearly with the number of organizations. For efficiency, each file is encrypted with a randomly generated file encryption key (FEK), which is then encrypted by ABE.



Remarks. We note that, although using ABE and MA- ABE enhances the system scalability, there are some limitations in the practicality of using them in building PHR systems. For example, in workflow-based access control scenarios, the data access right could be given based on users' identities rather than their attributes, while ABE does not handle that efficiently. In those scenarios one may consider the use of attribute-based broadcast encryption [32]. In addition, the expressibility of our encryptor's access policy is somewhat limited by that of MA-ABE's, since it only supports conjunctive policy across multiple AAs. In practice, the credentials from different organizations may be considered equally effective, in that case distributed ABE schemes [33] will be needed. We designate those issues as future works.

III. SECURITY ANALYSIS

It achieves data confidentiality (i.e., preventing unauthorized read ac- cesses), by proving the enhanced MA-ABE scheme (with efficient revocation) to be secure under the attribute- based selective-set model [21], [34]. We have the following main theorem.Our framework achieves forward secrecy, and security of write access control. For detailed security analysis and proofs, please refer to the online supplementary material of this paper. We also compare the security of our scheme with several existing works, in terms of confidentiality guarantee, access control granularity and supported revoca- tion method etc. We choose four representative state-of- the-art schemes to compare with: 1) the VFJPS scheme.From the system aspect, each data owner (patient) uses the YWRL ABE scheme for setup, key generation and revocation, uses both YWRL and enhanced MA- ABE for encryption. Each PSD user adopts the YWRL scheme for decryption, while each PUD user adopts the enhanced MA-ABE scheme for decryption. Each AA uses enhanced MA-ABE for setup, key generation and revocation. Next we provide estimations of computation times of each party in the system.

IV. CONCLUSION

Considering partially trustworthy cloud servers, we argue that to fully realize the patient-centric concept, patients shall have complete control of their own privacythrough 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 previousworks. We utilize ABE to encrypt the PHR data, 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.

REFERENCES

[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. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted personal health records in cloud computing," in ICDCS '11, Jun. 2011.

[4] "The health insurance portability and accountability act." [Online]. Available: http://www.cms.hhs.gov/HIPAAGenInfo/01Overview.asp

[5] "Google, microsoft say hipaa stimulus rule doesn't apply to them," http://www.ihealthbeat.org/Articles/2009/4/8/.

[6] "At risk of exposure – in the push for electronic medical records, concern is growing about how well privacy can be safeguarded," 2006. [Online]. Available: http://articles.latimes.com/2006/jun/26/health/he-privacy26

[7] K. D. Mandl, P. Szolovits, and I. S. Kohane, "Public standards and patients' control: how to keep electronic medical records accessible but private," BMJ, vol. 322, no. 7281, p. 283, Feb. 2001.

[8] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," in CCSW '09, 2009, pp. 103–114.

[9] 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.

[10] C. Dong, G. Russello, and N. Dulay, "Shared and searchable encrypted data for untrusted servers," in Journal of Computer Security, 2010.