International Journal For Technological Research In Engineering
Volume 1, Issue 6, February – 2014 *ISSN (Online): 2347 - 4718*

SCALABLE AND PROTECTED SHARING RECORDS IN CLOUD COMPUTING USING ABE

S. Senthilkumar
ME, Department of Computer Science and Engineering
A.S.L Paul's College of Engineering & Technology, Coimbatore
Anna University, Chennai, India.

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 paper, 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 attribute based encryption (ABE) techniques to encrypt each patient's PHR file. Different from previous works in secure data outsourcing, we focus on the multiple data owner scenario, and divide the users in the PHR system into multiple security domains that greatly reduces the key management complexity for owners and users. A high degree of patient privacy is guaranteed simultaneously by exploiting multi-authority ABE. Our scheme also enables dynamic modification of access policies or file attributes, supports efficient on-demand user/attribute revocation and break-glass access under emergency scenarios. Extensive analytical and experimental results are presented which show the security, scalability and efficiency of our proposed scheme.**

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

I. 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 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. 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 adoption. 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 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 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]. To ensure patient-centric privacy control over their own PHRs, it is essential to have fine-grained data acces control mechanisms that work with semi-trusted servers. A feasible and promising approach would be to encrypt 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]. However, the goal of patient-centric privacy is often inconflict with scalability in a PHR system. The authorized users may either need to access the PHR for personal use or professional purposes. Examples of the former are family member and friends, while the latter can be medical doctors, pharmacists, and researchers, etc. We refer to the two categories of users as *personal* and *professiona*users, respectively. The latter has potentially large scale; should each owner herself be directly responsible for managing all the professional users, she will easily be overwhelmed by the key management overhead. In addition, since those users' access requests are generally unpredictable, it is difficult for an owner to determine a list of them. On the other hand, different from the single data owner scenario considered in most of the existing works [8], [9], in a PHR system, there are *multiple owners* who may encrypt according to their own ways, possibly using different sets of cryptographic keys. Letting each user obtain keys from every owner who's PHR she wants to read would limit the accessibility since patients are not always online. An alternative is to employ a central authority (CA) to do the key management on behalf of all PHR owners, but this requires too much trust on a single authority (i.e., cause the key escrow problem). In this paperwe endeavor to study the patientcentric, secure sharing of PHRs stored on semi-trusted servers, and focus on addressing the complicated and challenging key management issues. In order to protect the personal health data stored on a semi-trusted server, we adopt attribute-based encryption (ABE) as the main encryption primitive. Using ABE, access policies are expressed based on the attributes of 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 attributes, without the need to know a complete list of users. The complexities per encryption, key generation and decryption are only linear with the number of attributes involved. However, to integrate ABE into a large-scale PHR system, important issues such as key management scalability, dynamic policy updates, and efficient on-demand revocation are non-trivial to solve, and remain largely open up-to-date. To this end, we make the following main contributions:

(1) We propose a novel ABE-based framework for patient-centric secure sharing of PHRs in cloud computing environments, under the multi-owner settings. To address 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 majority 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 applications' requirements, while incurring minimal key management overhead for both owners and users in the system. In addition, the framework enforces write access 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 personalized 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 implementing it on a modern workstation and performing experiments/simulations. Compared with the preliminary version of this paper [1], there are several main additional contributions: (1) we clarify and extend our usage of MA-ABE in the public domain, and formally show how and which types of user-defined file access policies are realized. (2) We clarify the proposed revocable MA-ABE scheme, and provide a formal security proof for it. (3) We carry out both real-world experiments and simulations to evaluate the performance of the proposed solution in this paper.

II. 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 Goyal et. al's seminal paper on ABE [11], data is 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. 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, Narayan et 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 ciphertext length grows linearly with the number of unrevoked users. In [17], a variant of ABE that allows delegation of access rights is proposed for encrypted EHRs. Ibraimi *et.al*. [18] Applied ciphertext policy ABE (CP-ABE) [19] to manage the sharing of PHRs, and introduced the concept of social/professional domains. In [20], Akinyele et al. investigated using ABE to generate self-protecting EMRs, which can either be stored on cloud servers or cellphones so that EMR could be accessed when the health provider is offline. However, there are several common drawbacks of the above works. First, they usually assume the use of a single trusted authority (TA) in the system. This not only may create a load bottleneck, but also suffers from the key escrow problem since the TA can access all the encrypted files, opening the door for potential privacy exposure. In addition, it is not practical to delegate all attribute management tasks to one TA, including certifying all users' attributes or roles and generating secret keys. In fact, different organizations usually form their own (sub) domains and become suitable authorities to define and certify different sets of attributes belonging to their (sub) domains (i.e., *divide and rule*). For example, a professional association would be responsible for certifying medical specialties, while a regional health provider would certify the job ranks of its staffs. Second, there still lacks an efficient and on-demand user revocation mechanism for ABE with the support for dynamic policy updates/changes, which are essential parts of secure PHR sharing. Finally, most of the existing works do not differentiate between the personal and public domains, which have different attribute definitions, key management requirements and scalability issues. Our idea of conceptually dividing the system into two types of domains is similar with that in [18], however a key difference is in [18] a single TA is still assumed to govern the whole professional domain.

Recently, Yu *et al*. (YWRL) applied key-policy ABE to secure outsourced data in the cloud [9], [15], where a single data owner can encrypt her data and share with multiple authorized users, by distributing keys to them that contain attribute-based access privileges. They also propose a method for the data owner to revoke a user efficiently by delegating the updates of affected ciphertexts and user secret keys to the cloud server. Since the key update operations can be aggregated over time, their scheme achieves low amortized overhead. However, in the YWRL scheme, the data owner is also a TA at the same time. It would be inefficient to be applied to a PHR system with multiple data owners and users, because then each user would receive many keys from multiple owners, even if the keys contain the same sets of attributes. On the other hand, Chase and Chow [21] proposed a multiple-authority ABE (CC MAABE) solution in which multiple TAs, each governing a different subset of the system's users' attributes, generate user secret keys collectively. A user needs to obtain one part of her key from each TA. This scheme prevents against collusion among at most $N - 2$ TAs, in addition to user collusion resistance. However, it is not clear how to realize efficient user revocation. In addition, since CC MA-ABE embeds the access policy in users' keys rather than the ciphertext, a direct application of it to a PHR system is non-intuitive, as it is not clear how to allow data owners to specify their file access policies. We give detailed overviews to the YWRL scheme and CC MAABE scheme in the supplementary material.

B. 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 MAABE. In addition, Ruj *et al*. [25] proposed an alternative\ solution for the same problem in our paper using Lewko and Waters's (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 MA- ABE. 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 ciphertext component to every non-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. The framework captures applicationlevel requirements of both public and personal use of a patient's PHRs, and distributes users' trust to multiple authorities that better reflects reality. We also propose a suite of access control mechanisms by uniquely combining the technical strengths of both CC MA-ABE [21] and the YWRL ABE scheme [9]. Using our scheme, patients can choose and enforce their own access policy for each PHR file, and can revoke a user without involving high overhead. We also implement part of our solution in a prototype PHR system.

III. 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. The main notations are summarized.

A. Problem Definition

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 [27], 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 [8], [20].

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.

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, usercontrolled read/write access and revocation are the two core security objectives for any electronic health record system, pointed out by Mandl et. 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 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, 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.

IV. 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. 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

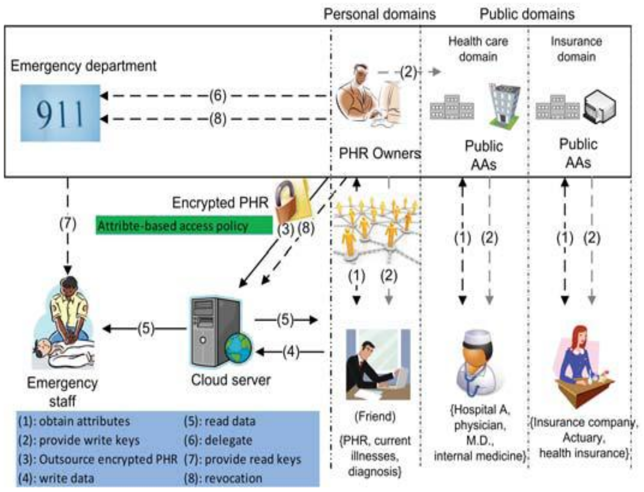

Fig. 1. The proposed framework for patient-centric, secure and scalable PHR sharing on semi-trusted storage under multi-owner settings.

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

A. Details of the Proposed Framework

In our framework, there are multiple SDs, multiple owners, multiple AAs, and multiple users. In addition

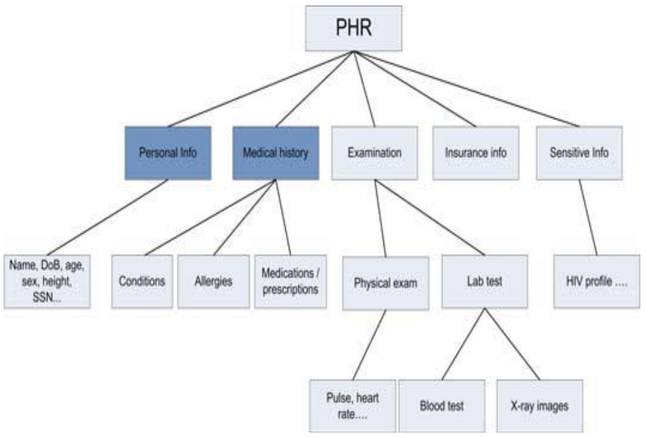

Fig. 2. The attribute hierarchy of files – leaf nodes are atomic file categories while internal nodes are compound categories. Dark boxes are the categories that a PSD's data reader have access to.

two ABE systems are involved: for each PSD the YWRL's revocable KP-ABE scheme [9] is adopted; for each PUD, our proposed revocable MA-ABE scheme (described in Sec. 4) is used. The framework is illustrated in Fig. 1. We term the users having read and write access as data readers and contributors, respectively.

B. System Setup and Key Distribution

The system first defines a common universe of data attributes shared by every PSD, such as "basic profile", "medical history", "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. For the PUDs, the system defines role attributes, and a reader in a PUD obtains secret key from AAs, which binds the user to her claimed attributes/roles. For example, a physician in it would receive "hospital A, physician, M.D., internal medicine" as her attributes from the AAs. In practice, there exist multiple AA seach governing a different subset of role attributes. Forinstance, hospital staffs shall have a different AA from pharmacy specialists. This is reflected by (1) in Fig. 1. MA-ABE is used to encrypt the data, and the concrete mechanism will be presented in Sec. 4. In addition, the AAs distribute write keys that permit contributors in their PUD to write to some patients' PHR ((2)).

V. 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.

A. Using MA-ABE in the Public Domain

For the PUDs, our framework delegates the key management 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.

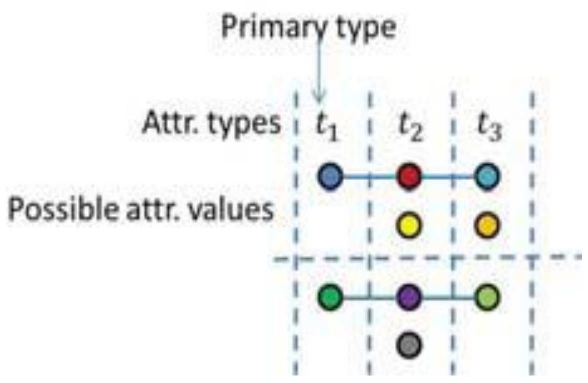

Fig. 3. Illustration of the enhanced key-policy generation rule. Solid horizontal lines represent possible attribute associations for two users.

Given theorem essentially states, the CC MAABE can be used in a fashion like CP-ABE when the document access policy is CNF. In practice, the above rules need to be agreed and followed by each owner and AA. It is easy to generalize the above conclusions to conjunctive forms with each term being a threshold logic formula, which will not be elaborated here.

Achieving More Expressive File Access Policies

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 attributes/ 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 nonintersected attribute tuples are included in the ciphertext's attribute set. This primary-type based attribute association is illustrated in Fig. 3. Note that there is a "horizontal association" 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 with in 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.)

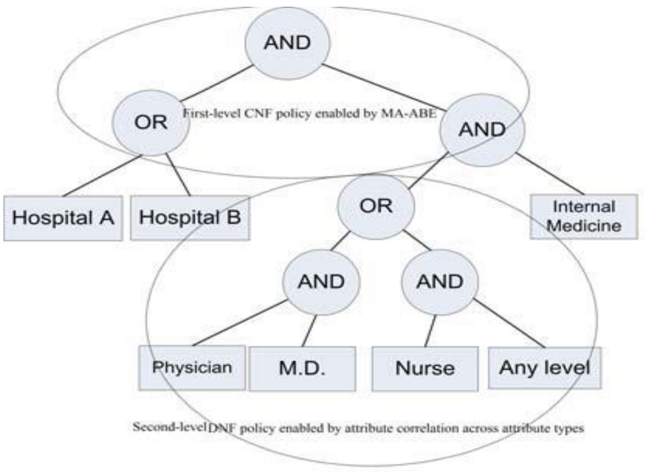
AND
OR
First-level CNF policy enabled by MA-ABE
AND
Hospital A
Hospital B
OR
Internal Medicine
AND
AND
Physician
M.D.
Nurse
Any level
Second-level DNF policy enabled by attribute correlation across attribute types

Fig. 4. An example policy realizable under our framework using MA-ABE, following the enhanced key generation and encryption rules.

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.

VI. SCALABILITY AND EFFICIENCY

Storage and Communication Costs

First, we evaluate the scalability and efficiency of our solution in terms of storage, communication and computation costs. We compare with previous schemes in terms of ciphertext size, user secret key size, public key/information size, and revocation (re-keying) message size. Our analysis is based on the worst case where each user may potentially access part of every owners' data. Table 4 is a list of notations, where in our scheme:
$|U| = |UD| + |UR|$, $tc = |AC\ PSD| + |AC\ PUD|$ (includes one emergency attribute), and $tu = |Au\ PSD| + |Au\ PUD|$ (a user could be both in a PSD and PUD). Note that, since the HN, NGS and RNS schemes do not separate PSD and PUD, their $|U| = |UR|$, $tc = |AC\ PUD|$, and $tu = |Au\ PUD|$.

However, they only apply to PHR access in the PUD. While in the VFJPS and BCHL schemes this is linear with *No*, since a user needs to obtain at least one key from each owner who's PHR file the user wants to access. For public key size, we count the size of the effective information that each user needs to obtain. The VFJPS scheme requires each owner to publish a directed acyclic graph representing her ACL along with key assignments, which essentially amounts to O (*Nu*) per owner. This puts a large burden either in communication or storage cost on the system. For re-keying, we consider revocation of one user by an owner in VFJPS and BCHL. In VFJPS, revoking one user from a file may need over-encryption and issuing of new public tokens for all the rest of users in the worst case. The NGS scheme achieves direct user revocation using ABBE, which eliminates the need of re-keying and re-encryption; however, attribute revocation is not achieved; and for the revocable ABBE in [32], either the ciphertext size is linear with the number of revoked users, or the public key is linear with the total number of users in the system4. For the RNS scheme, the main drawback is the large size of revocation messages to betransmitted to non-revoked users.

In our scheme, revocation of one user *u* requires revoking a minimum set of data attributes that makes her access structure unsatisfiable. From Table 5, it can be seen that our scheme has much smaller secret key size compared with VFJPS and BCHL, smaller rekeying message size than VFJPS, HN and RNS, the size of ciphertext is smaller than NGS while being comparable with HN and RNS. The public key size is smaller than VFJPS and BCHL, and is comparable with that of RNS; while it seems larger than those of HN and NGS, note that we can use the large universe constructions [21] to dramatically reduce the public key size. Overall, compared with non-ABE schemes, our scheme achieves higher scalability in key management. Compared with existing revocable ABE schemes, the main advantage of our solution is small re-keying message sizes. To revoke a user, the maximum re-keying message size is linear with the number of attributes in that user's secret key. These indicate our scheme is more scalable than existing works. To further show the storage and communication costs, we provide a numerical analysis using typical parameter settings in the supplementary material.

VII. 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 security. Through implementation and simulation, weshow that our solution is both scalable and efficient.

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/01 Overview.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.
[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *CCS '06*, 2006, pp. 89–98.
[12] M. Li, W. Lou, and K. Ren, "Data security and privacy in wireless body area networks," *IEEEWireless Communications Magazine*, Feb. 2010.
[13] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *ACM CCS*, ser. CCS '08, 2008, pp. 417–426.
[14] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Ciphertext-policy attribute-based threshold decryption with flexible delegation and revocation of user attributes," 2009.
[15] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *ASIACCS'10*, 2010.
[16] S. Narayan, M. Gagn´e, and R. Safavi-Naini, "Privacy preserving ehr system using attribute-based infrastructure," ser. CCSW '10, 2010, pp. 47–52.
[17] X. Liang, R. Lu, X. Lin, and X. S. Shen, "Patient self-controllable access policy on phi in ehealthcare systems," in *AHIC 2010*, 2010.
[18] L. Ibraimi, M. Asim, and M. Petkovic, "Secure

management of personal health records by applying attribute-based encryption," *Technical Report, University of Twente*, 2009.
[19] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE S& P '07*, 2007, pp. 321–334.
[20] J. A. Akinyele, C. U. Lehmann, M. D. Green, M. W. Pagano, Z. N. J. Peterson, and A. D. Rubin, "Self-protecting electronic medical records using attribute-based encryption," Cryptology ePrint Archive, Report 2010/565, 2010, http://eprint.iacr.org/.
[21] M. Chase and S. S. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *CCS '09*, 2009, pp. 121–130.
[22] X. Liang, R. Lu, X. Lin, and X. S. Shen, "Ciphertext policy attribute based encryption with efficient revocation," *Technical Report, University of Waterloo*, 2010.

S.Senthilkumar received the ME degree in computer Science at Annamalai University Chidhabaram .currently working assistant professor at the A.S.L Pauls College of Engineering & Technology ,Coimbatore .His research interest include reliability/security of cloud computing and storage, distributed systems and Networks.