International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 9 Issue 2 –AUGUST 2014.

Scalable and Secured Sharing of Personal Health Record using Attribute based Encryption in Cloud Computing

G.Premalatha[1], S.Thirunavukarasu[2], Dr.A.Kumaravelu[3]

Department Of Information Technology, Bharath University

[1]prema.sekar07@gmail.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 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.

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 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.

While it is exciting to have convenient PHR services for everyone, there are many security and privacy risk 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, cloud providers are usually not covered entities.

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. 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 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 However, the goal of patient-centric privacy is often in conflict 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 professional 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 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 paper, we endeavor to study the patient centric, secure sharing of PHRs stored on semi-trusted servers, and focus on addressing the complicated an hallenging key management issues. In order to protect he 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 there are several main additional contributions:

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.

We clarify the proposed revocable MA-ABE scheme, and provide a formal security proof for it.

We carry out both real-world experiments and simulations to evaluate the performance of the proposed solution in this paper.

The rest of the paper will be organised as follows: In section 2, we see about the related works of the paper. In section 3, we discuss about the proposed method. The algorithms and simulation are shown in the section 4 and 5. The conclusion of our paper is in section 6.

II. LITERATURE REVIEW

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength.

Once these things r satisfied, ten next steps is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration r taken into account for developing the proposed system.We have to analysis the IEEE TRANSACTIONS ON Parallel and Distributed System.

Parallel and Distributed Computing:

Distributed systems are groups of networked computers, which have the same goal for their work. The terms "concurrent computing", "parallel computing", and "distributed computing" have a lot of overlap, and no clear distinction exists between them. The same system may be characterized both as "parallel" and "distributed"; the processors in a typical distributed system run concurrently in parallel. Parallel computing may be seen as a particular tightly-coupled form of distributed computing, and distributed computing may be seen as a loosely-coupled form of parallel computing. Nevertheless, it is possible to roughly classify concurrent systems as "parallel" or "distributed" using the following criteria:

In parallel computing, all processors have access to a shared memory. Shared memory can be used to exchange information between processors.

In distributed computing, each processor has its own private memory (distributed memory). Information is exchanged by passing messages between the processors.

The figure on the right illustrates the difference between distributed and parallel systems. Figure (a) is a schematic view of a typical distributed system; as usual, the system is represented as a network topology in which each node is a computer and each line connecting the nodes is a communication link. Figure (b) shows the same distributed system in more detail: each computer has its own local memory, and information can be exchanged only by passing messages from one node to another by using the available communication links. Figure (c) shows a parallel system in which each processor has a direct access to a shared memory.

The situation is further complicated by the traditional uses of the terms parallel and distributed algorithm that do not quite match the above definitions of parallel and distributed systems; see the section Theoretical foundations below for more detailed discussion. Nevertheless, as a rule of thumb, high-performance parallel computation in a shared-memory multiprocessor uses parallel algorithms while the coordination of a large-scale distributed system uses distributed algorithms.

History

The use of concurrent processes that communicate by message-passing has its roots in operating system architectures studied in the 1960s. The first widespread distributed systems were local-area networks such as Ethernet that was invented in the 1970s.

ARPANET, the predecessor of the Internet, was introduced in the late 1960s, and ARPANET e-mail was invented in the early 1970s. E-mail became the most successful application of ARPANET, and it is probably the earliest example of a large-scale distributed application. In addition to ARPANET, and its successor, the Internet, other early worldwide computer networks included Usenet and FidoNet from 1980s, both of which were used to support distributed discussion systems.

The study of distributed computing became its own branch of computer science in the late 1970s and early 1980s. The first conference in the field, Symposium on Principles of Distributed Computing (PODC), dates back to 1982, and its European counterpart International Symposium on Distributed Computing (DISC) was first held in 1985.

III. Proposed Method

We endeavour to study the patient centric, 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.

IV. Implementation

(1) Registration:

In this module normal registration for the multiple users. There are multiple owners, multiple AAs, and multiple users. The attribute hierarchy of files – leaf nodes is 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 revocable KP-ABE scheme is adopted for each PUD, our proposed revocable MA-ABE scheme.

PUD - public domains
PSD - personal domains
AA - attribute authority
MA-ABE - multi-authority ABE
KP-ABE - key policy ABE

(2) Upload Files:

In this module, users upload their files with secure key probabilities. The owners upload ABE-encrypted PHR files to the server. Each owner's PHR file encrypted both under a certain fine grained model.

(3) ABE For Fine-Grained Data Access Control:

In this module 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). An attribute-based infrastructure for EHR systems, where each patient's EHR files are encrypted using a broadcast variant of CP-ABE that allows direct revocation. However, the cipher text length grows linearly with the number of un revoked users.

In a variant of ABE that allows delegation of access rights is proposed for encrypted EHRs applied cipher text policy ABE (CP-ABE) to manage the sharing of PHRs, and introduced the concept of social/professional domains 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.

(4) Setup And Key Distribution:

In this module 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)

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.

(5) Break- Glass:

In this module 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 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. After the emergency is over, the patient can revoke the emergent access via the ED.

V. 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 works.

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. Furthermore, 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.

VI. REFERENCES

[1] User Interfaces in C#: Windows Forms and Custom Controls by Matthew MacDonald.
[2] Applied Microsoft® .NET Framework Programming (Pro-Developer) by Jeffrey Richter.
[3] Practical .Net2 and C#2: Harness the Platform, the Language, and the Framework by Patrick Smacchia.
[4] Data Communications and Networking, by Behrouz A Forouzan.
[5] Computer Networking: A Top-Down Approach, by James F. Kurose.
[6] Operating System Concepts, by Abraham Silberschatz.
[7] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. USB-EECS-2009-28, Feb 2009.
[8] "The apache cassandra project," http://cassandra.apache.org/.
[9] L. Lamport, "The part-time parliament," ACM Transactionson Computer Systems, vol. 16, pp. 133–169, 1998.
[10] N. Bonvin, T. G. Papaioannou, and K. Aberer, "Cost-efficient and differentiated data availability guarantees in data clouds," in Proc. of the ICDE, Long Beach, CA, USA, 2010.
[11] O. Regev and N. Nisan, "The popcorn market. online markets for computational resources," Decision Support Systems, vol. 28, no. 1-2, pp. 177 – 189, 2000.
[12] A. Helsinger and T. Wright, "Cougaar: A robust configurable multi agent platform," in Proc. of the IEEE Aerospace Conference, 2005.
[13] J. Brunelle, P. Hurst, J. Huth, L. Kang, C. Ng, D. C. Parkes, M. Seltzer, J. Shank, and S.Youssef, "Egg: an extensible and economics-inspired open grid computing platform," in Proc.of the GECON, Singapore, May 2006.
[14] J. Norris, K. Coleman, A. Fox, and G. Candea, "Oncall: Defeating spikes with a free-market application cluster," in Proc.of the International Conference on Autonomic Computing, New York, NY, USA, May 2004.
[15] C. Pautasso, T. Heinis, and G. Alonso, "Autonomic resource provisioning for software business processes," Information and Software Technology, vol. 49, pp. 65–80, 2007.

[16] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef, "Web services on demand: Wsla-driven automated management," IBM Syst. J., vol. 43, no. 1, pp. 136–158, 2004.

[17] M. Wang and T. Suda, "The bio-networking architecture: a biologically inspired approach to the design of scalable, adaptive, and survivable/available network applications," in Proc. of the IEEE Symposium on Applications and the Internet,2001.

[18] N. Laranjeiro and M. Vieira, "Towards fault tolerance in web services compositions," in Proc. of the workshop on engineering fault tolerant systems, New York, NY, USA, 2007.

[19] C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He, "Transparent symmetric active/active replication for servicelevel high availability," in Proc. of the CCGrid, 2007.

[20] J. Salas, F. Perez-Sorrosal, n.-M. M. Pati and R. Jim´enez-Peris, "Ws-replication: a framework for highly available web services," in Proc. of the WWW, New York, NY, USA, 2006,