Privacy Preserving EHR System Using Attribute-based Infrastructure

Shivaramakrishnan Narayan, Martin Gagné and Reihaneh Safavi-Naini
Department of Computer Science
University of Calgary, Alberta, Canada
{snarayan,mgagne,rei}@ucalgary.ca

ABSTRACT

Secure management of Electronic Health Records (EHR) in a distributed computing environment such as cloud computing where computing resources including storage is provided by a third party service provider is a challenging task. In this paper, we explore techniques which guarantees security and privacy of medical data stored in the cloud. We show how new primitives in attribute-based cryptography can be used to construct a secure and privacy-preserving EHR system that enables patients to share their data among healthcare providers in a flexible, dynamic and scalable manner.

Categories and Subject Descriptors

E.3 [**Data Encryption**]: Public Key Cryptosystem

General Terms

Algorithms, Design, Security

Keywords

Cloud Computing, Attribute-based Cryptography, PEKS, Electronic Health Record System, Ciphertext Policy, Privacy

1. INTRODUCTION

An electronic health record is a collection of patients' health related information to allow efficient, consistent and universal sharing of health information. Because of the sensitivity of health related information, providing secure storage and access to EHR is the main challenge in today's EHR systems. Patients' privacy has been recognized in law, and privacy law such as the health insurance portability and accountability act (HIPAA) [1] privacy and security rules, and personal information protection and electronic documents act for health data [2] are aimed at ensuring sufficient care is given to handling such data.

An increasingly popular approach in managing health data puts users at the centre of such systems and allows users to store and manage access to their own health information. *Patient-centric EHR systems* enable patients to selectively give access to their health data to healthcare providers and other. Following the electronic health record system of Szolovits et al. [15], numerous patient-centric EHR such as Indivo [12], Google Health [8] and Microsoft Healthvault [13] have been proposed.

Cloud computing makes on-demand computing resources a reality and provides the required computing infrastructure for storage of EHR. However, a cloud environment introduces an even greater risk to security and privacy of sensitive data. Data stored in cloud may reside on computers that are located in dispersed geographic locations and can be seen by many in transit and in their stored form.

In this paper, we consider a secure design for a patient-centric EHR management system where data is saved in a storage provided by a cloud provider. We assume the cloud provides a reliable storage for data but the stored data can be seen (and copied) by the cloud provider. This means that it is the responsibility of the user to provide mechanisms that ensure security and privacy of their information. Users store their data in encrypted form in the cloud and grant access to portion of the data in accordance with the requesters' identity information. The storage provider will not be able to see data, or associated metadata, therefore confidentiality and privacy of data will be guaranteed. The scheme presented can also be applied to other general security-sensitive database applications.

1.1 Our proposal

Users of the system are patients and healthcare providers (e.g. doctors, laboratories, pharmacist).

We require the EHR system to guarantee:

- confidentiality of health data in storage and transit. By confidentiality we mean the cloud provider or an adversary will not be able to read patients' files.
- privacy of data. By privacy, we mean that the cloud provider will not be able to infer information about the file's content other than what is revealed through the file size, attributes, the identity of the requester, and the sequence of operations on the EHR.

The centrepiece of our design is the use of *attribute-based encryption (ABE)* [14]. An ABE system is a public key encryption system in which each user's key is labeled with a set of attributes and the ciphertext is associated with an access policy. The secret key of a user (the key is attached with an attribute set) can decrypt a particular ciphertext only if the attribute set of the user's key satisfies the access policy associated with the ciphertext. Such an ABE is

CCSW'10, October 8, 2010, Chicago, Illinois, USA.
Copyright 2010 ACM 978-1-4503-0089-6/10/10...$10.00.

known as ciphertext-policy ABE (*cp-ABE*) [5]. Recently, the use of attribute-based cryptography to provide secure cloud storage was also considered in [9].

The solution we propose is based on the following assumptions:

- There is a trusted authority (TA) who generates keys for users of the system. There is also a public directory that is used by the TA to publish the system public values (such as public keys) and parameters that are needed for cryptographic operations.
- A user is associated with (i) a unique identifier (ID), and (ii) a set of attributes (ω).
  Each user has a public key and a private key. The private key is generated and issued by the TA after verifying the user's attributes.
- The health record database is hosted on a cloud storage. The cloud server is trusted for performing the requested operation but should not be able to do other unspecified operations such as reading patients' data. Therefore the health information on the storage must be kept in secured form.

An EHR can be seen as a directory of files stored in subfolders. We assume an electronic health record has the following structure:

- (i) patients' health data in the form of files (in encrypted form), (ii) a table consisting of entries corresponding to the files.
- an entry for a file X in the table contains the following:

  (a) Meta data describing the file and its location, all in encrypted form. This data contains the following information and is encrypted using the broadcast ciphertext-policy attribute-based encryption.

    i. file description outlining the file content;

    ii. a random *locator-tag* that is used as a label for the file. This label is the file name used by the cloud server to find locate the file.

    iii. a symmetric key used to encrypt the health data.

  (b) an access policy in plain text form, which specifies who can decrypt the encrypted data of both the entry and file. The access policy is not encrypted.

  (c) a search-index (SI) for keywords within the encrypted file used for keyword search. The search-index is the encryption of the keyword.

A user that satisfies the access policy of a file, can decrypt the meta data entry associated with the file, find the location of the file and also a key to decrypt the file. The user submits the locator tag to the system to obtain the encrypted file and uses the key to decrypt it.

The system allows the users to share their health data with healthcare providers selectively by encrypting the data using the attributes of the healthcare providers. Note that the actual data is encrypted using efficient symmetric key cryptography, and the attribute-based encryption is used for making the symmetric keys accessible to authorized users. We use a variation of ABE known as *broadcast ciphertext-policy attribute-based encryption (bABE)*. A *bABE* system has the same functionality as ABE in making a plaintext accessible to users that satisfy the policy attached to the ciphertext, together with the added functionality of user revocation: that is users' keys can be revoked. The EHR system works as follows:

- **Store File:** A patient storing a new file with locator-tag loc_F would perform $Store\langle F\rangle$. The file stored on the server has two parts: $(\langle F\rangle,\ \langle Entry_F\rangle)$, where

$$\langle F\rangle := Enc_K(\mathrm{Data_F}),$$
$$\langle Entry_F\rangle := (bABE((Desc_F, K,\ loc_F), \mathbb{A}_0),\ \mathbb{A}_0,\ SI),$$

where $\mathbb{A}_0$ is an access policy that only allows the patient to decrypt the file.

- **Set Access:** A patient granting access to $\langle F\rangle$ to a healthcare provider U_B performs $Grant\langle F, U_B\rangle$. This creates a new policy $\mathbb{A}_1$ which grants access to U_B in addition to the existing rules in the current policy of F, and updates the entry of F to:

$$\langle Entry_F\rangle := (bABE((Desc_F, K,\ loc_F), \mathbb{A}_1),\ \mathbb{A}_1,\ SI).$$

Note that this can be done without re-encrypting the whole ciphertext.

- **Revoke Access:** A patient removing the access of the healthcare provider U_B to the encrypted file F would perform $Revoke\langle U_B, F\rangle$. This creates a new policy $\mathbb{A}'_1$ which does not allow access to U_B, but maintains all other rules of the current policy, and updates the entry of F to:

$$\langle Entry_F\rangle := (bABE((Desc_F, K,\ loc_F), \mathbb{A}'_1),\ \mathbb{A}'_1,\ SI).$$

- **Delegate:** A healthcare provider U_B with attribute set ω_B can create a private key for a subset $\omega_C \subseteq \omega_B$ for healthcare provider U_C using $Delegate\langle\omega_B, \omega_C\rangle$. This creates a private key for ω_C.

- **Keyword Search:** A patient can provide searchability of their records for a search term to a healthcare provider, by providing them with a search key. The healthcare provider U_B will then be able to search the EHR of U_A for a keyword w. The search is performed by the cloud provider on the encrypted data such that the cloud provider learns nothing about w.

The EHR system provides the following functionalities:

Adding User-access: The system allows a patient to add health care providers attributes to provide access to the patient's health data.

Revoking User-access: The system allows a patient to revoke a health care provider's access to the patient's health data without having to re-encrypt the data.

Access Delegation: The system provides health care providers the ability to delegate access by generating the private key for any subset of attributes they hold.

Keyword Search: The system allows the health care providers to search a patient's record based on a keyword such that the cloud server learns nothing about the keyword. The server returns the entries that match the query.

In addition, the EHR must provide security and privacy to the users, meaning:

Security: The system guarantees the confidentiality of health data even if the data server is compromised. Data is always in encrypted form and only users who are authorized by the patient can decrypt the EHR data.

Privacy: The system ensures that the cloud server will not learn anything about the file content from the ciphertexts or the keyword searches, other than what is leaked by the size of the files and inferences based on the identity and attributes of the requester.

1.2 Related Work

The concept of patient-centered health information systems was initially introduced by Szolovits et al., in 1994 [15]. In their work, the authors proposed an individual-based health information system which integrates all health-related information about an individual. Following their work, a patient-centric health record management system named Indivo [12] was proposed. This web-based system enables a patient to assemble, maintain and manage a secure copy of his or her medical record. The security of the health record is based on policies set by the users, and using encryption where the data is encrypted such that only the trusted Indivo server which has access to the private keys can decrypt the data before sending it to the users. Recently, Microsoft [13] and Google [8] introduced their patient-centric web-based electronic health record system. These systems allow patients to maintain a copy of their health record, access the health data whenever needed, and share the data with other users based on the access policies set by the patient.

Many EHR systems use cryptographic protocols to provide security and privacy. In [7], Hu at al. proposed the use of public key infrastructure (PKI) for privacy and security of EHR. They use PKI for mutual authentication and distribution of health data, and use of symmetric encryption to preserve the confidentiality of the health data. However, in their system, the trust and security management is delegated to the medical provider much like in a paper-based EHR. In [16] Yu and Chekhanovoskiy presented a patient-centric medical content protection system based on the use of tamper resistant hardware and cryptographic protocols for authentication and encryption of the health data. The drawback of their system is the use of dedicated hardware device for EHR content protection by the users, further the authors do not provide the security analysis of the proposed solution. In [10], a cryptographic key management approach to provide security and privacy of EHR was proposed. The solution was based on smartcard technology which requires the users of the EHR system to have a dedicated smartcard.

Recently, Benaloh et al. proposed an EHR using patient-controlled encryption [4]. Their work is closest to ours. They use hierarchical identity-based encryption (HIBE) and searchable encryption to construct a privacy-preserving EHR system. The Benaloh et al., EHR system provides the advantage of implementing the EHR system over any third party storage device (such as cloud storage). The drawbacks of their proposed solution however are the need to create and manage multiple keys by patients and healthcare providers, the absence of an efficient user revocation mechanism, the need for an external key escrow agent, and the need for patients to verify the healthcare provider's credentials.

In our system, the key management problem is solved by using users' attributes to encrypt the data. As a result, every user has only one private key corresponding to their attribute set which is used to decrypt the entry which in turn provides access to the encrypted files. The credentials of healthcare providers are verified only by the trusted authority when keys are generated, and the TA can also be used as an escrow agent in emergencies. We use a more secure and more flexible protocol to provide the keyword search functionality. Finally, the use of broadcast ABE allows patient to revoke access to healthcare providers.

2. PRELIMINARIES

2.1 Health Record Structure

A patient's health record is composed of various health data pertaining to different areas such as dentistry, cardiology, dermatology, mental health, physical data summary, etc. The data in each area can be of different types such as a SOAP (subjective, objective, assessment, and plan) note for a medical condition, consultation report, lab reports such as x-ray, ultrasound and blood test, discharge summaries, and operative reports. Contrary to [4], where the access policies were determined by the nature of the data contained in the files, we set our access policies based on the attributes of the users who are allowed access.

A patient may want to share his record to a doctor but may not want to allow others (e.g. pharmacist) to read any more information than strictly necessary. Therefore, we propose a system in which a patient can grant access to specific portions of his health data. Initially, the patient's data is encrypted with access to patient only. Granting user access to the already encrypted data would involve addition of the relevant attributes to the ciphertext. For example, if a patient wants a pharmacist to access his medical prescription, the patient adds the type-identifier pharmacy, and pharmacist attributes such as pharmacy-id and pharmacy-location to the medical prescription ciphertext.

2.2 User Attributes

The user/subject attributes consists of type-identifier (e.g. patient, doctor, pharmacy), and attributes that defines the identity and characteristics of the subject such as name, ID, location, specialization, etc. In the proposed EHR system, the patient decides on who can access his health data and thereby determines the attribute set under which the resource is encrypted.

2.3 Access Policies

The access policies for our system consist of monotone boolean formulas on the attributes – that is, boolean formulas that use only the logical 'or' and logical 'and' gates. For example, the policy '$\mathsf{doctor} \wedge \mathsf{ID1234}$' states that only a user who possess the attributes doctor and $\mathsf{ID1234}$ is allowed access. Since our system assumes the existence of a trusted key generator who verifies the user attributes before issuing private keys, the patient can ensure that his file can be accessed by user 1234 only if he is a doctor by setting the access policy above.

2.4 Attribute-based Cryptography

Attribute-based encryption (ABE) was introduced in 2005 by Sahai and Waters as an extension of their work called fuzzy identity-based encryption [14]. Attribute-based systems allow security functionalities to be provided based on 'attributes' of users and objects, and not their individual identities (though, individual identities can still be used as one of the attributes).

Our EHR system uses an adaptive chosen ciphertext (CCA-2) secure broadcast ciphertext-policy attribute-based encryption [3]. This scheme is secure in so called *selective attribute* model; but recent developments in ABE [11] indicate that adaptively secure broadcast schemes will soon be available. Broadcast ciphertext-policy attribute-based encryption enables us to achieve direct revocation of user access to a data without having to re-encrypt the data or refreshing the system parameters. A patient allows access to the data by adding the subject's type-identifier, subject-ID and other necessary subject attributes, based on the required access-policy the patient seeks. The ciphertext contains an index set consisting of the unique subject-ID and if the subject-ID of the receiver is not in the index set he cannot decrypt the data although he holds other attributes.

A broadcast ciphertext-policy attribute-based encryption scheme consists of four algorithms namely Setup, Extract, Encrypt and Decrypt, and an optional algorithm Delegate.
bABE-Setup(1^λ)**:** Given a security parameter 1^λ, the algorithm outputs the system public key *params* and master secret key *msk*. The global system parameters *params* is a set of public information.
bABE-Ext$(params, msk, \omega, ID)$**:** Given an attribute set ω, user-index ID, public key *params*, and the master key *msk* as input, the algorithm outputs the private key $sk_{ID,\omega}$ corresponding to ω.
bABE-Enc$(params, M, S, \mathbb{A})$**:** Given a message M, public key *params*, user-index set S and an access structure $\mathbb{A}$, the encrypt algorithm produces the ciphertext C containing the encrypted message.
bABE-Dec$(params, C, sk_{ID,\omega_r})$**:** The decrypt algorithm takes as input the ciphertext C and the private key sk_{ID,ω_r} for the receiver attribute set ω_r. It decrypts the encrypted message if $\omega_r \in \mathbb{A}$ and $ID \in S$, if successful it returns the message M, else returns $\perp$.
bABE-Del$(params, sk_{x,y}, (x', y'))$**:** The delegate algorithm takes as input a private key $sk_{x,y}$ and a new set (x', y'). It outputs the private key $sk_{x',y'}$ for any $y' \subseteq y$ and for any $x' \in \mathcal{U}$, where $\mathcal{U}$ is the user-index universe.
Advantages: A CCA-2 secure broadcast ciphertext-policy encryption ensures both confidentiality of the health data. This property follows directly from the security property of the encryption scheme. The broadcast ciphertext-policy attribute-based encryption also provides direct revocation capability which in turn provides the user revocation functionality of the proposed EHR system. The *delegate* algorithm of the encryption scheme is used to provide the access delegation functionality.

2.5 Keyword Search over Encrypted Data

We provide the keyword search functionality using a primitive called *Secure Channel Free Public-Key Encryption with Keyword Search* (*PEKS*) [6]. This scheme enables users to perform private searches for matching keywords over encrypted data without revealing the keywords or partial matches to the server.

A PEKS consists of the following algorithms:
PEKS-KG$_P(1^\lambda)$**:** Given a security parameter 1^λ, the algorithm outputs the system common parameter *cp*.
PEKS-KG$_S(cp)$**:** Taking a common parameter *cp* as input, this algorithm generates a private and public key pair (sk_S, pk_S) of the server.
PEKS-KG$_R(cp)$**:** Taking a common parameter *cp* as input, this algorithm generates a private and public key pair (sk_R, pk_R) of the receiver.
PEKS(cp, pk_S, pk_R, w)**:** Taking a common parameter *cp*, a server's public key pk_S, a receiver's public key pk_R and a keyword w as input, this algorithm returns a S_{cf}PEKS ciphertext S_w which is a searchable encryption of w.
PEKS-Td(cp, sk_R, w)**:** Given a common parameter *cp*, receiver's secret key sk_R and a keyword w, the algorithm computes a trapdoor T_w for w.
PEKS-Test(cp, S_w, T_w, sk_S)**:** The Test algorithm takes as input a common parameter *cp*, ciphertext S_w, trapdoor $T_{\omega'}$, and server's secret key sk_S. It outputs 1 if successful, else it outputs 0.
Advantages: The PEKS provides the keyword search functionality of the EHR system such that the server does not learn anything about the keyword on which the search is performed. This is because the server performs the **Test** function which takes as input the common parameter, ciphertext, trapdoor and server's secret key, none of these reveal any information about the keyword.

3. BUILDING A PATIENT-CENTRIC EHR SYSTEM

In this section we outline the algorithms needed to construct a patient-centric EHR (PEHR) system satisfying the properties mentioned in Section 1.

We use a CCA2-secure broadcast attribute-based encryption to maintain the confidentiality of the health records. The key idea is to allow a patient encrypt his medical records under the health care provider's attributes such that only those who satisfy the access policy set by the patient are able to decrypt the encrypted records. The authenticity of health care providers are verified by a trusted key generation authority who upon successful verification of attributes issues them the private key. We assume that all the users of the system trust the key generation authority. A user is allowed to delegate access to others by issuing a private key for a subset of the user's attributes. Further, each user has a user-index which is used to provide direct revocation of user access to an encrypted data. The encryption scheme of our PEHR system consists of algorithms:
PEHR-KG(1^λ)**:** Run **bABE-Setup**(1^λ) to obtain the system public parameter *params* and the master secret *msk* which is kept secret by the trusted key generation authority.
PEHR-Ext(ω, ID)**:** Run **bABE-Ext**$(params, msk, \omega, ID)$ and output the private key $sk_{ID,\omega}$.
Enc$_K(M)$**:** Perform a symmetric encryption of M using the key K.
bABE$(M, \mathbb{A})$**:** Run **bABE-Enc**$(params, M, S, \mathbb{A})$, and output the resulting ciphertext C.
Dec$_{sk_{ID,\omega}}(C)$**:** Run **bABE-Dec**$(params, C, sk_{ID,\omega})$, and output the resulting message M.
Dec$_K()$**:** Perform a symmetric decryption using the key K.
Delegate(x', y')**:** Run **bABE-Del**$(params, sk_{x,y}, (x', y'))$ and return the secret key $sk_{x',y'}$.

3.1 Adding Searchability

To provide keyword search within a health record, we combine the attribute-based broadcast encryption with a searchable encryption scheme. For this, we propose the use of secure channel-free public key encryption with keyword search [6]. The main idea is when a medical document is uploaded, each keyword pertaining to the uploaded document is encrypted using the PEKS scheme, and the encrypted keyword known as search-index is stored in the entry corresponding to the encrypted document. The owner of the decryption key can generate a trapdoor key for a particular keyword which will allow the server to test whether a particular search-index is a match without knowing the keyword.

In our system, the patient generates and issues a public encryption key and private decryption key to the health care providers, thus enabling the health care providers to generate the trapdoor key. The searchable encryption scheme is as follows:

Srch-KGen(1^λ): Run **bABE-Setup(1^λ)** to obtain the common parameter *cp*. Let *msk* be the master secret of the system known only to the trusted key generation authority.

Srch-SGen(): Run **PEKS-KG$_S$(cp)** to obtain the public and private key pair (pk_S, sk_S) of the server. This is run by the trusted key generator who creates the key pair using the common parameter and the master secret key *msk*. The public key of the server pk_S is added to the search common parameter *cp* which is made public to all users.

Srch-RGen(): Run **PEKS-KG$_R$(cp)** to generate a private and public key pair (sk_R, pk_R) of the receiver. This is done by the patients who generate and issue the key pair to the health care providers. The health care providers can search for keywords if the search-index is produced using the public key pk_R issued by the patient.

Srch-IGen(pk_R, w): Run **PEKS(cp, pk_S, pk_R, w)** and output a PEKS ciphertext S_w which is a search index for the keyword w.

Srch-Td(sk_R, w): Run **PEKS-Td(cp, sk_R, w)** and output a trapdoor T_w for the keyword w.

Search(T_w): Run **PEKS-Test(cp, S, T_w, sk_S)** and output 1 if successful, else output 0.

3.2 Analysis

The proposed scheme has the following advantages compared to the Benaloh et. al, scheme of [4].

Efficient Key Management. Benaloh et. al's scheme requires the patient to send a large number of keys to healthcare providers. In our scheme, the decryption of the medical data relies on the private key of the health care providers. Here, the patients encrypt the data under the attributes of the health care providers, thus the patients need not generate and issue keys unlike in [4] thus reducing the communication cost.

Further, each time a health provider accesses a patient's record, the health provider needs to fetch the subkey which involves decrypt operation in order to decrypt the entries and the encrypted file. In our scheme, access to a patient's record does not require the health care provider to fetch a subkey.

Our scheme has a slightly higher computational cost for the patients due to re-encryption of records when updating access policies, but this is largely offset by the reduction in communication.

Direct Revocation. In our scheme a patient can revoke the access of a health care provider without having to refresh the public parameter or re-encrypt the health data. This is achieved using the *ID* of the health care provider whose access needs to be revoked. Direct revocation of access is not addressed in [4].

Key Escrow. In [4], the patients have to set up an external key escrow service or informally share the keys to the family in order to be able to access the health record in case of emergency. Our scheme naturally supports key escrow due to the presence of a trusted key generation authority who is able to generate private keys to access a patient's health record in case of emergency. A downside of this approach is that the trusted key generator has the capability to access all encrypted files.

Usability. Benaloh et. al's scheme requires the patients to verify the health provider credentials before issuing the keys. In our scheme, the private keys are issued by a trusted key generator who verifies the health provider attributes before issuing the private key.

4. PRIVACY PRESERVING EHR IN PRACTICE

We now describe the interactions between, a doctor, a patient, and the health record database which stores the patient's health record.

Setup: The user (both patient and doctor) connects to the web server and generates an account username and password which will be used to authenticate the user to the server for logging in to the system. Once a user has registered for the service, he downloads a client application which will run locally on his machine. Once this client application is installed, the user requests the private key corresponding to his attribute set from the trusted authority.

A patient then sets up his health record by importing the medical data, encrypting each file with a different symmetric key and storing the encrypted files in the cloud storage. Along with each encrypted file, the patient uploads a corresponding entry consisting of encrypted meta-data, access policy and search-index.

Note that at the end of every session, both on the patient's and doctor's end, the client application clears all the data - keys and decrypted materials - used during the session.

Initiating a Contact: Prior to a medical appointment, both the patient and the doctor need to exchange their username. To view the patient's record, the doctor sends a request for access to the health data of the patient. The request message is sent to the patient by encrypting it under the patient's public key which is downloaded by the client application given the patient's username.

The patient provides access to his health data by retrieving the entries corresponding to the requested files. He then updates the policy of the entries thereby enabling the doctor to access the corresponding files. Finally the patient sends an encrypted message to the doctor confirming that the access has been granted.

Accessing Health Record: After being granted access by the patient, the doctor makes an access request to the server on the patient's record. First, the doctor's client software retrieves the encrypted entries and decrypts them using the doctor's private key to access the file information and locator tags. The doctor then sends the locator tags of the corresponding files he wants to access to the server, which then returns the encrypted files. This encrypted file is decrypted at the client side and is available for reading.

Post Appointment: Update Health Record: If the doctor wants to update files or upload new data, the doctor sends to the patient the authenticated medical data file encrypted under the public key of the patient. In case of a fresh upload, the patient decrypts the message containing the authenticated medical data and creates a new encrypted file and a corresponding entry such that only the patient and the doctor are given access. For an update, the patient encrypts the file with a new key and updates the corresponding entry while preserving the existing access policy.

5. EHR SECURITY

Our proposal ensures the confidentiality of health data by the means of using strong encryption primitive. In this case, we use adaptive chosen ciphertext secure broadcast ciphertext-policy attribute-based encryption and public-key searchable encryption. This implies if the health record database is compromised the adversary learns nothing about the health data contained in the server without the relevant private keys. Since the system requires a trusted authority to issue the keys, the security of the system relies on the trust and reliability of the authority.

The security of the keys are handled by the direct revocation capability of the system. If a patient believes that a health provider is not trustworthy the user can directly revoke the access instead of refreshing the system public parameters. Since updated files are encrypted using new keys, a revoked provider having access to locator-tag and previous key cannot read the new file. Further, to ensure the data written is authenticated, we assume that the health providers use attribute-based signing of the data.

The security of the proposed system is based on the following assumptions:

◇ The trusted authority only issues the private key on the successful verification of user attributes.

◇ The private key is communicated to the users over a secure link such as SSL thereby preventing eavesdropper to learn anything about the key.

◇ After each session, the client application deletes all the data used during this session. This ensures an adversary cannot retrieve any data from the machine's temporary storage location.

◇ Trusted authority overrides the access to a patient's record only during the case of emergency.

6. CONCLUSION AND FUTURE WORK

Attribute-based cryptographic primitives provides flexible policies which can be used to build secure infrastructure for designing privacy preserving electronic health record system. In this paper, we presented a proposal which combines attribute-based cryptography and public-key encryption with keyword search to provide privacy preserving electronic health record management system.

We plan to develop a prototype of the proposed EHR system using Indivo [12], an open source software developed for patient-centric health record management. We also intend to extend our proposed solution to support multiple authorities to reduce the trust requirement in the key generators.

7. ACKNOWLEDGEMENTS

This research was funded in part by Alberta Innovates - Technology Futures, Alberta, Canada. The authors would like to thank the reviewers of ACM CCSW 2010 for providing helpful comments and suggestions.

8. REFERENCES

[1] Health insurance portability and accountability act of 1996. U.S. Government Printing Office, 1996.

[2] Recommendations for the interpretation and application of the personal information protection and electronic documents act (s.c.2000, c.5) in the health research context. Technical report, Canadian Institutes of Health Research, November 2001.

[3] N. Attrapadung and H. Imai. Conjunctive broadcast and attribute-based encryption. In *Pairing '09: The 3rd International Conference on Pairing-Based Cryptography*, volume 5671 of *Lecture Notes in Computer Science*, pages 248–265. Springer-Verlag, 2009.

[4] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter. Patient controlled encryption: Ensuring privacy in medical health records. In *ACM CCSW 2009*, 2009.

[5] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy, 2007, SP '07*, pages 321–334. IEEE Xplore, 2007.

[6] L. Fang, W. Susilo, C. Ge, and J. Wang. A secure channel free public key encryption with keyword search scheme without random oracle. In *CANS '09: Proceedings of the 8th International Conference on Cryptology and Network Security*, pages 248–258. Springer-Verlag, 2009.

[7] J. Hu, H. Chen, and T. Hou. A hybrid public key infrastructure solution (HPKI) for HIPAA privacy/security regulations. *Computer Standards and Interfaces*, 32(5-6):274–280, 2009.

[8] Google Inc. Google health. https://www.google.com/health/, 2009.

[9] S. Kamara and K. Lauter. Cryptographic cloud storage. In *Financial Cryptography: Workshop on Real-Life Cryptographic Protocols and Standardization - 2010*, volume 6052 of *Lecture Notes in Computer Science*. Springer-Verlag, 2010.

[10] W. B. lee and C. D. Lee. A cryptographic key management solution for HIPAA privacy/security regulations. *IEEE Transactions on Information Technology in Biomedicine*, 12:34–41, 2008.

[11] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology-EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer-Verlag, 2010.

[12] K. Mandl, W. Simons, W. Crawford, and J. Abbett. Indivo: a personally controlled health record for health information exchange and communication. *BMC Medical Informatics and Decision Making*, 7(1):25, 2007.

[13] Microsoft. Microsoft healthvault. http://www.healthvault.com/personal/ websites-overview.html, 2009.

[14] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Advances in Cryptology-EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer Berlin/Heidelberg, 2005.

[15] P. Szolovits, J. Doyle, W. J. Long, I. Kohane, and S. G. Pauker. Guardian angel: Patient-centered health information systems. Technical report, 1994.

[16] W. D. Yu and M. A. Chekhanovskiy. An electronic health record content protection system using smartcard and PMR. In *9th International Conference on e-Health Networking, Application and Services, 2007*, pages 11–18. IEEE Xplore, 2007.