PERSONAL HEALTH RECORDS SHARING IN CLOUD COMPUTING ENVIRONMENT

D.Jennifer #1 M.Saranya*

#Assistant Professor, Dept of Computer Science and Engineering
Apollo Engineering College, Chennai, Tamil Nadu,

*PG Scholar, Department of Computer Science and Engineering
Apollo Engineering College, Chennai, Tamil Nadu,

ABSTRACT

Personal Health Record (PHR) is very important for the patient. Personal health record (PHR) is an patient-centric model of health information exchange stored at a third party cloud providers. While maintaining the PHR in the outsourced environment such as cloud, it may not secure. Encryption is must before store the data in cloud. 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. The overall control is maintained by the data owner it increases the traffic overhead risk. To avoid this over head without disturb the security the proposed method introduces Attribute Based Encryption (ABE). It provides the secure outsourcing. 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. We are using Advanced Encryption Standard (AES) algorithm for securing the Each Patient Records.

1. INTRODUCTION

A PHR allows a patient to create, manage, and control her personal health data in one place through web. This PHR services has made the storage, retrieval, and sharing of the the medical information more efficient. Especially, each patient has the full control of his/her medical records and can share his/her health data with a wide range of users, including healthcare providers, family members or friends. But there are many security and privacy risks. 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 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. 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 by using Advanced Encryption standard (AES) with Attribute Based Encryption (ABE).

2. RELATED WORKS

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 scheme 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.

2.1 ABE for Fine-grained Data Access Control

There has been an increasing interest in applying ABE to secure electronic healthcare records (EHRs). Recently proposed 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 ciphertext length grows linearly with the number of unrevoked users. In a variant of ABE that allows delegation of access rights is proposed for encrypted EHRs applied ciphertext policy ABE (CP-ABE) to manage the sharing of PHRs, and introduced the concept of social/professional domains. there are several common drawbacks of the above works. First, they usually assume the use of a single trusted authority (TA) in the system. but also suffers from the key escrow problem since the TA can access all the encrypted files, opening the door for potential privacy exposure. Our idea of conceptually dividing the system into two types of domains is similar with that however a key difference is in a single TA is still assumed to govern the whole professional domain. On the other hand, Chase and Chow proposed a multiple-authority ABE (CC MA ABE)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.

3. ARCHITECTURE OF PHR

The Main Aim 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. 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 security domains, we use ABE to realize cryptographically enforced, patient-centric PHR access.

![Fig 1. Architecture of PHR](image)

In this architecture PUD uses multi-authority ABE which there are multiple “attribute authorities” (AAs), each having a disjoint subset of attributes. 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. They 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 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. 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.

4. PROPOSED SCHEME

4.1 Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is an encryption algorithm for securing sensitive data. AES is the Advanced Encryption Standard, a United States government standard algorithm for encrypting and decrypting data. AES is a symmetric block cipher with a block size of 128 bits. Key lengths can be 128 bits, 192 bits, or 256 bits; called AES-128, AES-192, and AES-256, respectively. AES-128 uses 10 rounds, AES-192 uses 12 rounds, and AES-256 uses 14 rounds. The main loop of AES performs the following functions:

4.2 Functions Of Advanced Encryption Standard

• SubBytes()
• ShiftRows()
• MixColumns()
• AddRoundKey()

The first three functions of an AES round are designed to cryptanalysis via the methods of “confusion” and “diffusion.” The fourth function actually encrypts the data. Diffusion means patterns in the plaintext are dispersed in the ciphertext. Confusion means the relationship between the plaintext and the ciphertext is obscured.

A simpler way to view the AES function order is:

1. Scramble each byte (SubBytes).
2. Scramble each row (ShiftRows).
3. Scramble each column (MixColumns).
4. Encrypt (AddRoundKey).
A term associated with AES is “the State,” an ‘intermediate cipher,’ or the ciphertext before the final round has been applied. AES formats plaintext into 16 byte (128-bit) blocks, and treats each block as a 4x4 State array. It then performs four operations in each round. The arrays contains row and column information used in the operations, especially MixColumns() and Shiftrows().

**SubBytes()**

SubBytes() adds confusion by processing each byte through an S-Box. An S-Box is a substitution table, where one byte is substituted for another, based on a substitution algorithm.

![SubBytes Diagram](image1)

**Fig 2. Sub Bytes**

To complete an S-Box operation on an example string of “ABC,” take the hexadecimal value of each byte. ASCII “A” == hex 0x42, “B” == 0x43 and “C” == 0x44. Look up the first (left) hex digit in the S-Box column and the second in the S-Box row. 0x42 becomes 0x2c; 0x43 becomes 0x1a, and 0x44 becomes 0x1b.

**ShiftRows()**

ShiftRows() provides diffusion by mixing data within rows. Row zero of the State is not shifted, row 1 is shifted 1 byte, row 2 is shifted 2 bytes, and row 3 is shifted 3 bytes, as shown in the FIPS illustration that follows:

In the ShiftRows step, bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs for each row.

![ShiftRows Diagram](image2)

**Fig 3. Shift Rows**
The ShiftRows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left. Similarly, the third and fourth rows are shifted by offsets of two and three respectively. For blocks of sizes 128 bits and 192 bits, the shifting pattern is the same. Row n is shifted left circular by n-1 bytes. In this way, each column of the output state of the ShiftRows step is composed of bytes from each column of the input state. (Rijndael variants with a larger block size have slightly different offsets). For a 256-bit block, the first row is unchanged and the shifting for the second, third and fourth row is 1 byte, 3 bytes and 4 bytes respectively—this change only applies for the Rijndael cipher when used with a 256-bit block, as AES does not use 256-bit blocks. The importance of this step is to avoid the columns being linearly independent, in which case, AES degenerates into four independent block ciphers.

**Mix Columns()**

In the MixColumns step, each column of the state is multiplied with a fixed polynomial $c(x)$.

In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher.

**Fig 4. Mix Columns**

During this operation, each column is multiplied by a fixed matrix:

$$
\begin{bmatrix}
2 & 3 & 1 & 1 \\
1 & 2 & 3 & 1 \\
1 & 1 & 2 & 3 \\
3 & 1 & 1 & 2 \\
\end{bmatrix}
$$

Matrix multiplication is composed of multiplication and addition of the entries, and here the multiplication operation can be defined as this: multiplication by 1 means no change, multiplication by 2 means shifting to the left, and multiplication by 3 means shifting to the left and then performing XOR with the initial unshifted value. After shifting, a conditional XOR with 0x1B
should be performed if the shifted value is larger than 0xFF. (These are special cases of the usual multiplication in \( \text{GF}(2^8) \).) Addition is simply XOR.

In more general sense, each column is treated as a polynomial over \( \text{GF}(2^8) \) and is then multiplied modulo \( x^4+1 \) with a fixed polynomial \( c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02 \). The coefficients are displayed in their hexadecimal equivalent of the binary representation of bit polynomials from \( \text{GF}(2)[x] \). The MixColumns step can also be viewed as a multiplication by the shown particular MDS matrix in the finite field \( \text{GF}(2^8) \). This process is described further in the article Rijndael mix columns.

AddRoundKey()

In the AddRoundKey step, each byte of the state is combined with a byte of the round subkey using the XOR operation (\( \oplus \)).

Fig 5. Add Round key

In the AddRoundKey step, the subkey is combined with the state. For each round, a subkey is derived from the main key using Rijndael's key schedule; each subkey is the same size as the state. The subkey is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR.
5. RESULTS AND DISCUSSION

This scheme uses the advanced encryption standard for encrypting the attributes of all the patient details. Attribute Based Encryption is used to separate the patient records into each and every attribute and all this attributes are encrypted with the help of (AES) algorithm. The patient records are maintained without any leakage of data by combining using Attribute Based Encryption (ABE) technique and Advanced Encryption Standard (AES) algorithm together.

![PHR Patient Analysis]

6. CONCLUSION

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.
7. REFERENCES

[5] “At risk of exposure – in the push for electronic medical records, concern is growing about how well privacy can be safeguarded.”