CP-ABE for Selective Access with Scalable Revocation: A Case Study for Mobile-based Healthfolder

Divyashikha Sethia[1], Huzur Saran[2], and Daya Gupta[1]
(Corresponding author: Divyashikha Sethia)
Department of Computer Science and Engg., Delhi Technological University, India[1]
Shahbad Daulatpur, Main Bawana Road, Delhi, 110042
(Email: sethiadivya@gmail.com)
Department of Computer Science and Engg., Indian Institute of Technology, Delhi[2]
Hauz Khas, New Delhi, Delhi 110016
(Received Mar. 19, 2017; revised and accepted May 16 & June 5, 2017)

Abstract

With the recent advancement in computational and storage capabilities on mobile devices and Internet of Things (IoT), Ciphertext policy Attributed-based Encryption (CP-ABE) can provide confidentiality and direct selective fine-grained access control. There must be an ease of maintaining ciphertext, capability to share and protection against breach of trust. We present a novel revocation scheme Scalable Proxy-based Immediate Revocation for CP-ABE (SPIRC) which does not require prior revocation list, re-encryption and re-distribution of keys. It improves the Proxy-based Immediate Revocation of ATTribute based Encryption (PIRATTE) scheme for scalable revocation with reduced overheads for proxy data and master key generation. The paper also demonstrates the practical implementation of SPIRC for a case study of a portable Mobile-based Healthfolder on a patient mobile device for direct local access as well as sharing with medical professionals using reader application on their mobile devices. The performance evaluation on mid-range Android devices indicates acceptable overheads for access and security.
Keywords: CP-ABE; Mobility; RBAC; Scalable Revocation

1 Introduction

Attribute-Based Encryption (ABE) [15] provides fine-grained access control for sharing ciphertext with a group of users. It comprises of a set of plaintext attributes and an access policy to generate the ciphertext and decryption keys so that each user has a different decryption key. ABE has the advantage that users cannot aggregate their attributes together to decrypt the ciphertext and hence, it is collusion-free. There are several variations of ABE [22, 28] such as Key-Policy Attribute-Based Encryption (KP-ABE) scheme, Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and Hierarchical Attribute-Based Encryption scheme (HABE). The CP-ABE) [8] variation associates a set of attributes to the decryption key and the access policy to the ciphertext. A decryption key can decrypt the ciphertext only if it's associated attributes satisfy the access policy of the ciphertext. Users can be assigned different decryption keys, with each decryption key associated with a subset of attributes that satisfies the ciphertext's access policy. Since CP-ABE can provide Role-based access control (RBAC) by representing a set of attributes for a specific role, we choose it for selective sharing of ciphertext. It also supports revocation as well as collusion resistance. However, it lacks support for scalability as discussed in the survey comparison by Lee *et al.* [22]. CP-ABE has been used for several cloud-based data sharing applications such as for health [23] as well as proved feasible on resource-constrained portable devices such as mobile devices [3] and Internet of Things (IoT) [12].

Motivation: Cloud-based storage solutions are prone to security threats and may not provide 24/7 support in the case of an outage or lack of infrastructure. There is an increase in the penetration of smartphones across the globe. Hence personal portable mobile devices may retain highly available critical data such as that for health [29] and finance and share it directly with other users.

In this paper, we present a case study of a secure portable mobile-based healthfolder on a patient mobile device to store dispersed health data and share it directly with other health professionals. It is a future health management system which can improve availability, sharing capability and mobility to seek the right diagnosis and

treatment across various hospitals.

Health records may be dispersed due to patients visiting various hospitals and hence increase overheads for health management. Developing countries like India lack proper healthcare policies and infrastructure required for a centralised health system. Hence, people visit various hospitals for seeking specialised consultations and second opinions for a reliable diagnosis which leads to dispersed health records.

Health management systems in developed countries are well established. Patients are associated with a particular healthcare or insurance policy such as NPfIT system in U.K [31] and Taiwan Electronic Medical Record Template (TMT) suggested by Chen *et al.* [9]. However, records may be dispersed in the case of citizen mobility for work and tourism across the various states and countries. Also for an emergency situation, a patient may land in a hospital which is not under his health policy. Developed countries have strict and structured health policies which may cause challenges in integrating dispersed health records on cloud-based solutions. Hence, a portable device with health records can benefit patient for high availability as suggested by Anciaux *et al.* [4]. A mobile-based healthfolder can provide mobility to patients to seek efficient treatment and retain their health records securely on their personal devices for both developed and developing nations. Section 5 discusses the case study of the mobile-based healthfolder in detail.

Problem: Since the mobile device is vulnerable to security and privacy threats; it is important to maintain confidentiality and allow selective sharing with authorised users. This paper considers schemes based on Bethencourt *et al.*'s CP-ABE [8] to retain and share secure data on a mobile device since it has been implemented and proved feasible on mobile devices and IoT [3, 12]. The owner of the portable device must access it locally and directly share with other authorised users using selective access policy. There must also be protection from malicious users using a revocation scheme with minimal overheads. We identify the following requirements for retaining and selective sharing data on a portable device using CP-ABE:

R1: No prior knowledge of the revocation list.
There must be no prior requirement for a revocation list for encryption so that the ciphertext can be shared with multiple users.

R2: No re-encryption of ciphertext.
There must be no requirement for re-encryption of ciphertext after revocation so that the owner and other authorised non-revoked users can access it without interruption.

R3: No re-distribution of decryption keys.
There must be no requirement for re-distribution of decryption keys after revocation so that the owner and other non-revoked users can continue to access the ciphertext without interruption.

R4: Revoke a scalable number of users.
The owner must be able to share ciphertext with multiple users as well as revoke a scalable number of malicious users.

R5: Independent of the ciphertext.
No ciphertext specific data must be maintained for user revocation to reduce storage and revocation overheads.

There are several revocation schemes for sharing data on the cloud such as suggested by the survey by Liu *et al.* [24]. However, they do not consider the issues of avoiding re-encryption or key re-distribution after revocation. The revocation schemes can be categorised as direct, indirect and hybrid revocation as discussed by Pang *et al.* [28]. Unlike direct schemes, the indirect schemes do not require any prior knowledge of a revocation list and support broadcast of an intermediate key update, such that only non-revoked users can update their keys. Hence, they are suitable for portable devices to provide ease and flexibility to the owner. They also require a key update phase which can provide bottleneck for interaction with the Certified Authority (CA). Proxy-based Immediate Revocation of Attribute-based encryption (PIRATTE) [21] by Jahid *et al.* is an indirect revocation scheme for CP-ABE which satisfies all of the above requirements except *R4* for scalability. Hence, there is a need to improve PIRATTE for scalable revocation for secure storage and sharing of critical data on a portable device.

Our Contribution:

- Design and implementation of a novel scheme called *Scalable Proxy-based Immediate Revocation for CP-ABE (SPIRC)* which extends PIRATTE [21] for scalable user revocation. It fulfils all revocation requirements *R1-R5* for sharing of secure data from portable devices. The scheme requires a trusted proxy-based server which manages cryptographic credentials for registered owner and users and also provides proxy data to users to complete decryption. The proxy server updates the proxy data for a revoked user so that decryption fails. Section 3.1 describes the details of the trusted server in the intuition.

- Demonstration of a case study for a next-generation Portable Mobile-based Healthfolder [29] on a patient mobile device which retains dispersed health records from various hospitals. The patient can share it directly with health professionals based on their roles such as a doctor, nurse, lab technician and a pharmacist. The Healthfolder is encrypted using the SPIRC scheme for selective access by health professionals and scalable revocation. The health professionals access it directly with their mobile devices as shown in Figure 1.

- Practical implementation and evaluation of SPIRC for the prototype of the Mobile-based Healthfolder

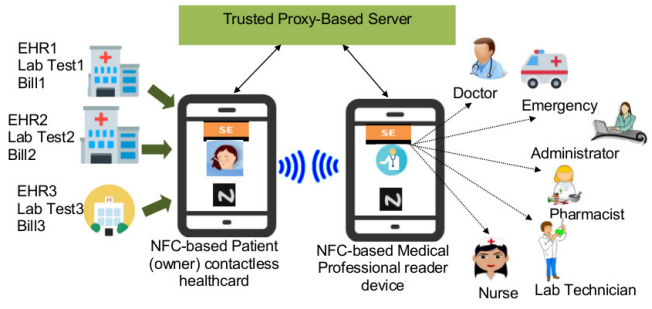

Figure 1: Mobile-based Healthfolder stakeholders: Patient, External users and trusted proxy-based server

on mid-range Android devices. Performance evaluation indicates acceptable delays for communication and security handshake. A comparison of different schemes for storage and computational overheads shows that SPIRC provides scalable revocation with lower overheads for proxy data and master key generation.

To the best of our knowledge, our work is the first novel attempt to address the issues of selective sharing and scalable revocation for a portable device using Bethencourt *et al.*'s CP-ABE scheme [8]. In future, we can work on scalable revocation schemes based on variations of CP-ABE with better performance such as Cheung *et al.*'s provably secure CP-ABE [10] and Lewko *et al.*'s CP-ABE scheme based on (Linear Secret Sharing Scheme) LSSS matrix [7].

The rest of the paper comprises of Related Work in Section 2, details of the new SPIRC scheme in Section 3 and Security Analysis of SPIRC in Section 4. Section 5 presents a Case Study for Selective Access for Portable Mobile-based Healthfolder along with its Security Analysis with SPIRC and Implementation and Performance Evaluation. It is followed by performance comparison of revocation schemes in Section 6. The paper finally concludes with Section 7 for Conclusion and Future work.

2 Related Work

The indirect revocation schemes for CP-ABE for portable devices must satisfy all revocation requirements *R1-R5* for the ease of portability, personal access for the owner and sharing data directly with other external authorised users.

CP-ABE techniques used in the cloud-based record sharing schemes such as those for health records are not directly suitable for portable devices. Narayan *et al.* [26] propose a broadcast variation of CP-ABE which has the limitation that the length of ciphertext grows proportionally with the number of revoked users. Hence, this may not be feasible for portable devices with limited storage. Liet *et al.* [23] suggest a scalable Electronic Health Record (EHR) scheme which uses revocation scheme by Worcester *et al.* [33] which requires re-encryption for revocation and violates requirement *R2* for a portable device.

Attrapadung and Imai [5] provide a hybrid revocation scheme which supports both direct and indirect modes. However, it has the drawback of longer user secret key length, which can be difficult to store on a portable mobile device. Ibraimi *et al.* [19] suggest an indirect revocation scheme which generates two portions of the private key one of which is retained by the user and the other with a mediator. The mediator sends the right portion of the key to a user only if it not revoked. However, it uses CP-ABE scheme by Cheung *et al.* [10] which has the drawback that there is an increase in the size of ciphertext and key with the increase in the total number of attributes in the access policy. Hence it is not suitable for a mobile device with limited storage. Modi *et al.* [25] propose a revocation scheme for secure file access on the cloud. However, it violates requirement *R1* needed for scalable sharing of ciphertext on a portable device. Hur *et al.* [17] propose an indirect revocation scheme to provide fine-grained attribute revocation with the limitation of requiring re-encryption of ciphertext and hence violates the requirement *R2*.

PIRATTE (Proxy-based Immediate Revocation of ATTribute based Encryption) [21] scheme by Jahid *et al.* is a variation of Bethencourt *et al.*'s CP-ABE [8], which provides indirect revocation without re-encryption of the ciphertext and key re-distribution. Users receive proxy data from the proxy-based server to complete decryption. PIRATTE scheme uses a polynomial P of degree $t+1$ in the master key. The trusted server divides the secret $P(0)$ into portions and provides a share to each user. During decryption, each user seeks a proxy key and t shares of the secret from the proxy-based server. It uses Lagrange's interpolation to combine the t secret portions with the user portion to generate the secret $P(0)$. If the user is non-revoked, the proxy-based server sends valid secret portions. Otherwise, it sends invalid secret portions, so that the user cannot generate the secret $P(0)$ and hence decryption fails. PIRATTE fulfils all revocation requirements, except for *R4* since it can revoke only limited t number of users.

A permanent revocation scheme (referred to as PERMREV in this paper) by Dolev *et al.* [13], modifies the Bethencourt *et al.*'s CP-ABE [8] scheme and associates a counter CTR with the ciphertext and a user state $State_i$ for the ith user $user_i$. It considers ciphertext to reside on a secure cloud-based system. For revocation of $user_i$, the secure server updates CTR, re-encrypts the ciphertext and sends the updated $State_i$ with new CTR only to the non-revoked users. Since revoked users do not receive any updated state, decryption fails. To avoid ciphertext re-encryption a Modified PERMREV scheme referred as M-PERMREV in this paper requires a server to broadcast $State$ to all users. For a revoked user, the server updates the CTR and the user state for only revoked users, which causes failure of decryption. However, M-PERMREV scheme does not fulfill requirement *R5* since it associates a constant CTR with the user's

state $State_i$ for every ciphertext.

CP-ABE can provide RBAC as suggested by role-based access control scheme (RACS) for sharing medical data on cloud by Tian *et al.* [32]. However it cannot be used for portable devices since it does not fulfill requirement *R2*.

The SPIRC scheme presented in this paper improves PIRATTE for scalable revocation and fulfils all requirements *R1-R5* for revocation.

3 Scalable Proxy-based Immediate Revocation For CP-ABE Scheme

Billinear pairings. Let G_1, G_2 and G_T be multiplicative cyclic groups of prime order p. Let g_1 and g_2 be a generator of G_1 and G_2 respectively. e is a bilinear map such that $e : G_1 \times G_2 \rightarrow G_T$. It has the following properties:

1) **Bilinearity:** for all u, v element of G_1, G_2 and a; b element of Z_p, $e(u^a, v^b) = e(u, v)^{ab}$.

2) **Non-degeneracy:** $e(g, g) \neq 1$.

Intuition: This paper looks into the issue of storing secure data on a portable device and sharing it using selective access control. It outsources encryption to a trusted proxy-based server. The owner decrypts the ciphertext locally on the personal portable device, and shares it directly with other users who decrypt it locally on their respective devices.

The trusted proxy-based server retains credentials and identities of registered users, as well as constants related to the proxy data.

Each $user_i$ registers with a trusted proxy-based server and is associated with a set of random parameters $S_i = \{\lambda_i, a_i, b_i\}$. The constants are associated with the decryption keys of the user as well as proxy data. For decryption, a user contacts the trusted server through a secure channel such as HTTPS to gather proxy data to complete the decryption process. The trusted server also maintains a revocation list RL which is populated by an authorized owner or an administrative personnel to protect portable device from malicious users on breach of trust or theft of device. To revoke a user, the proxy-based server updates S_i so that the proxy-based data is modified and causes decryption to fail.

The cloud-based service is contacted only for seeking proxy data and not for the actual ciphertext as in the cloud-based sharing applications [23]. The trusted proxy-server must comply to all requirements for Trusted Computing [30]. The trust between authorized users and proxy-server can be established through some of the existing techniques such as mutual authentication and remote attestation techniques as suggested in [6], to ensure that they are not compromised with any malicious software. Further authorized users can communicate with the trusted server using separate CP-ABE access policies for RBAC for allowing trusted revocation and configurations of credentials by users with administrative roles. This can ensure secure maintenance of credentials as well as revocation list on the proxy-server. The detailed design of the trusted proxy-based server are beyond the scope of the paper.

3.1 SPIRC Construction

The SPIRC scheme supports scalable user revocation without requiring re-encryption or re-distribution of keys. This paper modifies Jahid *et al.*'s PIRATTE [21] scheme for scalable revocation for infinite users. It comprises of the following algorithms:

Setup: Generates Public key PK and Master key MK.

Encrypt(PK, M, τ): Takes data M, Public key PK, and access policy τ to generate the ciphertext CT.

KeyGen(MK, S): Takes master key MK and set of attributes S and generates the secret key SK.

Proxy-Data(U_k, RL): Takes user identity u_k and the revocation list RL as input and generates the proxy data PXD. It also invokes CONVERT function to transform portion of the ciphertext C'_x for each attribute x satisfied by users u_k and generates the converted portion C''_x.

Decrypt(CT, SK): Decrypts the ciphertext CT to generate a plaintext M if the set of attributes S in SK satisfy the access policy τ that is used to generate ciphertext CT.

The details of the different phases are given below:

Setup. The trusted proxy-based server chooses G_1, G_2, g_1, g_2 and random elements α and $\beta \in Z_p$ to generate the public key PK and a master key MK.

$$\begin{aligned} PK &= G_1, G_1, g_1, g_2, h = g_1^{\beta}, e(g_1, g_2)^{\alpha} \\ MK &= \beta, g_2^{\alpha} \end{aligned}$$

Unlike PIRATTE, for master key MK, there is no generation of polynomial P of degree $t + 1$, where t is the number of users that can be revoked. Hence, it provides scalable revocation.

Encrypt(PK, M, τ): The tree structure τ represents the access policy with attributes at leaves and threshold of k-of-n gates at the interior nodes. q_x is the polynomial at node x with degree $d = k$ *-1*, where k is the threshold value of the node. For all OR nodes and leaf nodes, the polynomial degree is 0. The proxy-based server chooses a random secret $s \in Z_p$ for a message M, such that for root node $R, q_R(0) = s$. The secret is distributed from top to bottom for all other nodes, $q_x(0) = q_{parent(x)}(index(x))$, where $index(x)$ is a number associated with x between *1* and *num* (number of children of *parent(x)*). X is the

set of leaf nodes in the access tree τ. The ciphertext CT is: $CT = (\tau, \tilde{C} = Me(g_1, g_2)^{\alpha s}, C = h^s)$, $\forall x \in X : C_x = g_1^{q_x(0)}$, $C'_x = H(att(x))^{q_x(0)} = g_2^{h_x q_x(0)}$. $H : \{0,1\}* \rightarrow G_2$ is a hash function that maps a string attribute to a random element in G_2 and $h_x = \log_{g_2} H(att(x))$.

KeyGen(MK, S): It generates the secret key SK for $user_i$ for a set of attributes S. For each user i, it chooses a random number r along with set $S_i = (\lambda_i, a_i, b_i) \in Z_p$ and for each attribute j it chooses a random number $r_j \in Z_p$. $SK = (D = g_2^{(\alpha+r)/\beta}, \forall j \in S : D_j = g_2^r H(j)^{r_j(\lambda_i a_i + b_i)} = g_2^{r+h_j r_j(\lambda_i a_i + b_i)}, D'_j = g_1^{r_j}, D''_j = (D'_j)^{a_i} = g_1^{r_j a_i})$.

The portions of the secret key SK, D_j and D'_j for each attribute j contain random number r_j and D contains random number r which is specific to a user. Hence attributes from different users cannot be combined together and prevents collusion.

Proxy-Data($user_i$): Proxy-based server maintains a random set S_i for each user along with a revocation list. For the completion of decryption, $user_i$ seeks proxy data PXD from the proxy-based server which is unique for a user.

$PXD = \lambda_i$. The trusted server sends proxy data PXD to $user_i$, who also sends C'_x to the proxy-based server to return Convert C''_x as: $CONVERT(C''_x, b_i) = (C'_x)^{b_i} = g^{h_x q_x(0) b_i}$. The user secret SK is blinded by $(\lambda_i a_i + b_i)$ and needs C''_x along with C_x and C'_x. Proxy can revoke the user by updating the λ_i and b_i for $user_i$ in PXD and C''_x.

Decrypt. For a $user_i$, each leaf node x of the policy is an attribute, with $j = attr(x)$, if $j \in S$, (S is the set of attributes) then, $DecrytpNode = A_j$ is as follows:

$$
\begin{aligned}
A_j &= \frac{e(C_x, D_j)}{e(D''_j, C'_x)^{\lambda_i} e(D'_j, C''_x)} \\
e(C_x, D_j) &= e(g_1^{q_x(0)}, g_2^{r+h_j r_j(\lambda_i a_i + b_i)}) \\
&= e(g_1, g_2)^{q_x(0) r + q_x(0) h_j r_j(\lambda_i a_i + b_i)} \\
A_j &= \frac{e(g_1, g_2)^{q_x(0) r + q_x(0) h_j r_j(\lambda_i a_i + b_i)}}{e(g_1^{r_j a_j}, g_2^{h_j q_x(0)})^{\lambda_i} e(g_1^{r_j}, g_2^{h_j q_x(0) b_i})} \\
&= \frac{e(g_1, g_2)^{q_x(0) r + q_x(0) h_j r_j(\lambda_i a_i + b_i)}}{e(g_1, g_2)^{r_j a_i h_j q_x(0) \lambda_i} e(g_1, g_2)^{r_j h_j q_x(0) b_i}} \\
&= \frac{e(g_1, g_2)^{q_x(0) r + q_x(0) h_j r_j(\lambda_i a_i + b_i)}}{e(g_1, g_2)^{r_j a_i h_j q_x(0) \lambda_i + r_j h_j q_x(0) b_i}} \\
&= \frac{e(g_1, g_2)^{q_x(0) r + q_x(0) h_j r_j(\lambda_i a_i + b_i)}}{e(g_1, g_2)^{r_j h_j q_x(0)(\lambda_i a_i + b_i)}} \\
&= e(g_1, g_2)^{q_x(0) r}
\end{aligned}
$$

Each $user_i$ has associated constant values λ_i, a_i and b_i which are maintained on the proxy-based server. Whenever revocation is required, the proxy-based server updates λ_i or b_i, which are part of PXD and C''_x, and cause the DecryptNode function to fail and return $\perp$.

The rest of the decryption process is the same as in the Bethencourt *et al.*'s CP-ABE scheme [8] to obtain the original message M.

For each node z of a non-leaf node x, it calculates $F_z = e(g_1, e_2)^{r q_q(0)}$. If S_x is the set of children of x so that $F_z \neq \perp$. This is followed by the following decryption process:

$$
\begin{aligned}
F_x &= \prod_{i=1}^{S_x} F_z^{\lambda_i}, (i = index(z) \lambda_i calculated \forall z \in S_x) \\
&= \prod_{i=1}^{S_x} (e(g_1, g_2)^{r q_z(0)})^{\lambda_i} \\
&= \prod_{i=1}^{S_x} (e(g_1, g_2)^{r q_{parent(z)} index(z)})^{\lambda_i} \\
&= \prod_{i=1}^{S_x} (e(g_1, g_2)^{r q_x(i)})^{\lambda_i} \\
&= e(g_1, g_2)^{\sum_{i=1}^{S_x} r q_x(i) \lambda_i} \\
&= e(g_1, g_2)^{r q_x(0)}
\end{aligned}
$$

Let $A = e(g_1, g_2)^{r q_R(0)} = e(g_1, g_2)^{r q_R(0)} = e(g_1, g_2)^{rs}$ at root node R. Decryption can be done as follows, $\frac{\tilde{C}}{\frac{e(C,D)}{A}} = Me(g_1, g_2)^{\alpha s} \frac{e(g_1, g_2)^{rs}}{e(g_1, g_2)^{\alpha s + rs}} = M$.

4 Security Analysis For SPIRC

The definitions for user-based revocation are as per [21].

4.1 Security Game

In the security game between an adversary and a challenger, the encryption remains secure even when the adversary compromises the proxy and obtains it's key after a recent revocation.

Setup. A challenger runs the SETUP and provides public parameters PK to the adversary. Challenger also generates a proxy data PXD.

Phase 1. The adversary performs repeated queries for KEYGEN to obtain keys for multiple user $u_1, \cdots, u_{q_1}$ with different sets of attributes $S_1, \cdots, S_{q_1}$. The adversary also contacts the proxy for the $CONVERT(\{C'_1, \cdots, C'_r\}, u_k)$ for $C'_i \in G_1$. Simultaneously challenger also computes CONVERT with the stored values. The adversary contacts proxy server to get the proxy data PXD. In the meanwhile challenger updates the proxy data PXD.

Challenge. The adversary submits messages M_0 and M_1 of equal lengths and an access structure A* such that either u_k is to be revoked or S_k does not satisfy A*.

The challenger flips a coin to obtain a random bit b and returns M_b encrypted with the access policy A*. It also runs Proxy-Data and returns the proxy data PXD to the adversary.

Phase 2. The adversary makes repeated queries to the KEYGEN to obtain keys for users $u_{q_1+1}, \cdots, u_{q_2}$ with attributes $S_{q_1+1}, \cdots, S_{q_2}$. The new keys are such that if $u_k \notin$ revocation list RL, then S_k does not satisfy A*.

Guess. The adversary outputs a guess b' of b.

The adversary has an advantage defined as $Pr[b' = b] - \frac{1}{2}$. As in PIRATTE even if an adversary $user_j$ finds Proxy portions of another $user_i$, the portions will not help him with the decryption since each user has a different set of random constants values. The SPIRC scheme provides forward secrecy since a revoked user cannot decrypt any previously recorded ciphertext.

4.2 Security Proof

Asymmetric Groups Similar to PIRATTE [21] for user i and attribute j, different groups are used for C'_j and D'_j. The user sends C'_j to convert and receive C''_j, where $C''_j = C_j^{'b_i}$. If both C'_j and D'_j belong to the same group and user sends D'_j to convert, then user will get $D_j^{'b_j} = g_2^{r_j b_j}$. User will also get λ_j and can get $D_j^{''\lambda_j} = g_2^{r_j \lambda_j a_j}$. Combining these two terms by multiplication will provide $g_2^{r_j(\lambda_j a_j + b_j)}$. User can use this to decrypt any ciphertext without using the proxy-based server for revocation. Hence asymmetric pairing is used with different groups for C_j and D'_j.

Similar to PIRATTE [21], SPIRC is based on the generic asymmetric bilinear group model, which considers a asymmetric pairing of $e : G_1 \times G_2 \rightarrow G_T$, with the assumption that their is no isomorphism from G_1 to G_2. Both are based on CP-ABE [8] scheme, and hence are secure against Chosen Plaintext Attack (CPA). Other variations of CP-ABE such as Cheung *et al.*'s CP-ABE [10] are secure against Chosen Ciphertext Attack (CCA). However, this paper focuses on only Bethencourt et.al's CP-ABE scheme which has been proven feasible on mobile devices and IoT devices [3, 12].

Theorem 1. *The construction of SPIRC scheme is secure under the generic bilinear group model. It assumes that there is unexpected collisions between asymmetric groups.*

The paper assumes that in the security game, $A*$ contains single attribute A_j for some attribute j. After Phase 2, the adversary has the following elements for each user u_k and A_j from S_k: $G_1 : g_1, g_1^{\beta}$, $C = g_1^{\beta s}$, $C_j = g_1^s$, $D'_j = g_1^{r_{ukj}}$, $D''_j = g_1^{r_{ukj} a_k}$.

Secret s encrypts the message and $H(j) = g_2^{h_j}$. $G2 : g_2, D = g_2^{(\alpha+r)/\beta}, D_j = g_2^{r_u + h_j r_{ukj}(\lambda_k a_k + b_k)}, C'_j = g_2^{h_j s}$.

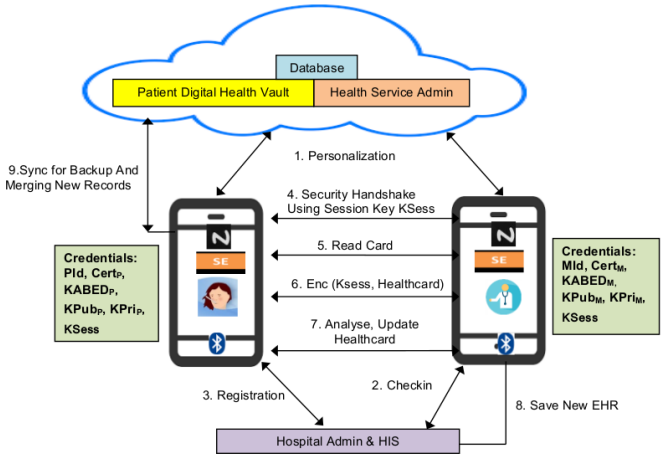

Figure 2: Selective access of a mobile-based healthfolder

$G_T : e(g_1, g_2)^{\alpha}$, $M.\ e(g_1, g_2)^{\alpha s}$.

Adversary only knows u_k for all revoked users in the revocation list $RL*$. However, secret s occurs only in elements of the ciphertext C, C_j and C'_j. To guess s, the adversary can compute $e(C, D^{(uk)}) = e(g_1, g_2)^{\alpha s + r_{uk} s}$. To determine $e(g_1, g_2)^{\alpha s}$, adversary must compute $e(g_1, g_2)^{rs}$. However, it is not feasible to compute it from D_j. Hence it is difficult for the adversary to determine the secret s in the security game. SPIRC is hence secure under the generic asymmetric bilinear group model.

5 Case Study: Selective Access Mobile-based Healthfolder

5.1 System Design

We present the system design for a Mobile-based Healthfolder on a patient device. The SPIRC scheme encrypts it and stores it on a secure storage with direct selective access. Figure 2 shows the system design. Our preliminary work in [29] is based on PIRATTE [21]. SPIRC scheme improves it for scalable revocation for enhanced portability and mobility of a patient across hospitals. The system comprises of a patient's mobile device with a Healthfolder containing different health data from dispersed hospitals.

The Mobile-based Healthfolder is retained as a large sized contactless card using NFC-based Host Card Emulation(HCE) [2]. A health professional accesses the software-based HCE contactless card by a tap of his mobile device, using IoT-based communication interfaces of NFC [11] and Bluetooth. It is supported by a cloud-based HealthSecure service which comprises of a trusted proxy-based server and a secure digital vault to store data sync. The proxy-based server maintains cryptographic credentials, unique user identities and support for SPIRC proxy decryption. The service can be managed by government intuitions, insurance companies or chain of well-known

Table 1: Main terms for mobile-based healthfolder

Term	Description
Idp	Identifier for Patient
Idm	Identifier for Health Professional
H/H'	Unencrypted/Encrypted Healthfolder
U	User (P-Patient/M-Health professional)
CU	**User's Credentials on SE**
{KUpub/KUpri	User's Public/Private RSA keys
Certu	User certificate for {Idu,KUpub}
KDRUabe	User CP-ABE Read decryption key
KDWUabe }	User CP-ABE Write decryption key
$Section_i$	Healthcard ith section, i = 1-7
ri	Random number for $Section_i$
rei	Encrypted ri for ith section E(KEWabe,ri)
RW={re1..re6}	Write policy encrypted random nos.
$Update_i$	Update for $Section_i$
Ksym	Symmetric Session key

Table 2: Healthfolder organization

Sections	1	2	3	4	5	6	7
Stakeholder	Basic Vitals	Allergies /Disease	Advan. Vitals	Medic	Lab/ Immun	Emerg / Admin	Non-clinical
Doctor	RW	RW	RW	RW	RW	R	R
Nurse	RW	R	R	R	R	R	R
Pharmacist	--	--	--	RW	--	R	--
Lab Tests	--	--	--	--	RW	R	--
Emergency	RW	RW	RW	RW	RW	RW	RW
Patient	R	R	R	R	R	R	RW
Admin	R	R	R	R	R	RW	R
Read Policy	ACRB	ACRB	ACRB	ACRM	ACRL	ACRALL	ACRB
Write Policy	ACWBV	ACWSP	ACWSP	ACWM	ACWL	ACWADM	ACWNC

Table 3: Healthfolder CP-ABE write access policies

Name	Section	Policy
ACWBV	Basic vitals	AND(wobb,OR(doctor,nurse,emerg))
ACWSP	Special	AND(wobs,OR(AND(doctor,department),emerg))
ACWM	Medication	AND(wmed,OR(doctor,pharm,emerg))
ACWL	Lab tests	AND(wlab,OR(doctor,pharm,emerg))
ACWADM	Admin	AND(wadm,OR(admin,patient))
ACWNC	Non-clinical	AND(wnoncl,OR(emerg,patient))

hospitals and must have a policy that complies with the requirements for Trusted computing [30].

Both devices of the patient and health professional register with the HealthSecure service and store secure credentials and identity on tamper resistant Secure Element (SE) in the form of a microSD card. It can be accessed internally through applications compiled with special libraries on the processor. The SE utilises Java Card [27] technology which enables Java-based applets to execute with limited memory and processing capabilities.

After an NFC tap between the mobile devices, they mutually authenticate and establish an asymmetric session key *Ksym*. The patient mobile device automates Bluetooth setup over HCE for higher throughput. All subsequent communication is encrypted using *Ksym*. A health professional reads and writes to set of sections on the Mobile-based Healthfolder over Bluetooth and terminates it after the transfer is complete.

Due to the high computational costs of bilinear pairing, the Mobile-based Healthfolder outsources CP-ABE encryption to the HealthSecure service. However, both the Patient and Medic mobile device locally decrypt to view the Healthfolder. The health professional evaluates the past records and provides diagnosis and treatment for the current medical condition. All new updates are written securely to the Mobile-based Healthfolder. Hence a patient retains upto date health records. Table 1 describes the main notations for the case study.

5.2 Selective RBAC with SPIRC

The Mobile-based Healthfolder retains different health records such as prescriptions, reports, medication details from various hospitals in standard formats such as HL7 [16] for interoperability. It organises each department record into different subsections. Various authorised health professionals access them as per their roles with selective RBAC as shown in Table 2.

For each section, a read access policy encrypts it, and a write access policy encrypts a section specific random number *ri* as *rei*. Table 2 shows the different access policies for each section on the Healthfolder. A stakeholder stores two decryption keys: a read decryption key *KDRUabe* and a write decryption key *KDWUabe* to access the authorised sections. A CP-ABE decryption key can decrypt all sections for which the attributes in the key can satisfy the section access policy.

A stakeholder first reads the Healthfolder and obtains the concerned sections by decrypting with his read decryption key *KDRUabe*. However, once he can read a section, he must be able to update it only if he has access according to the write access policy.

Figure 3 shows a sample read access policy *ACRALL* which permits all stakeholders to read. Each section has a different write access policy with a special set of associated attributes as shown in Table 3. For example, to read sections encrypted with *ACRM* and *ACRALL* read policies, a pharmacist must have a read decryption key with attributes that satisfy the related access policies. Similarly, to write to sections encrypted with *ACWM* write policy, the write decryption key must have attributes which satisfy the access policy. For example for a user pharmacist, the decryption key must have attributes *pharmacy*, time between and 4 and *wmed* to satisfy the access policy.

Write Access Policy. For each section *i* of the healthfolder, random number *ri* is encrypted with the write CP-ABE policy of the section as *rei*. When a stakeholder requests to write to section *i*, patient challenges it with the encrypted *rei* for the section. If the

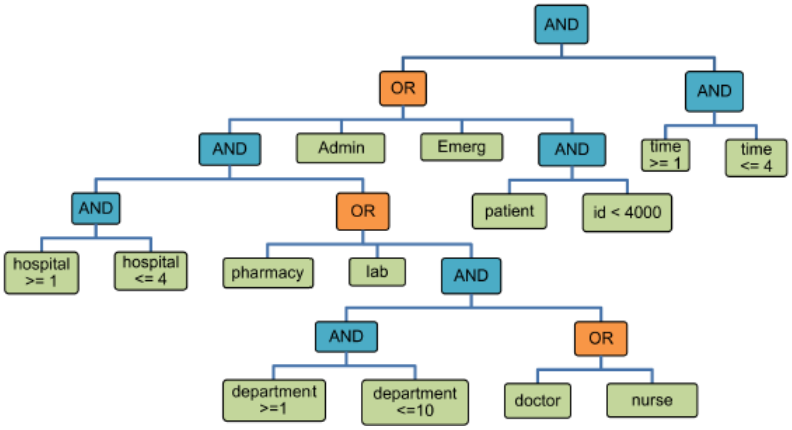

Figure 3: CP-ABE read access policy ACRALL

Table 4: Sequence for SPIRC-based selective access and scalable revocation

S.No	Messages	
1'.	Card:	Personalisation: ((KPpub, KPpri, KDRPabe, KDWPabe, RW=(re1..re7))
1".	Reader:	Personalisation: ((KMpub,KMpri) Non-emergency:KDRMabe, KDWMabe)
2.	Card ⟷ Reader:	Mutual Authentication to generate Ksym
3.	Card ← Reader:	Action: write/read, $Section_i$
4.	Card:	MP1=H' || rei
5.	Card → Reader:	E(Ksym,MP1)
6.	Reader ⟷ Server:	If Emergency personnel obtain BTG keys (KDRMabe, KDWMabe)
7.	Reader ⟷ Server:	Proxy-based server-based decryption H=D(KDRMabe,H'), ri=D(KDWMabe, rei)
8.	Server:	Revoke users in RL
9.	Server:	ri' = ri+1, $Access = hash(Update_i)$, $MM1 = ri'\|\|Update_i$
10.	Card ← Reader:	E(Ksym,MM1)
11.	Server:	If ri'== ri+1 then accept $Update_i$
12.	Card →: Server	$Update_i$ through HTTPS
13.	Server:	Revoke key if user is an Emergency personnel Sync $Update_i$ on digital vault, Re-encrypt H as H"
14.	Card ← Server:	H" through HTTPS

stakeholder has access to write, he can decrypt *rei* using his write decryption key *KDWUabe*. In response, he computes *ri'=ri+1* and sends it to the Mobile-based Healthfolder along with the update $Update_i$ for the section. The Mobile-based Healthfolder compares the received *ri'* and the locally computed value of $(ri+1)$. If they match, then the $Update_i$ is written on the healthfolder, else it is rejected.

Revocation. Healthsecure service associates time-based attributes with the decryption key and each stakeholder must renew it periodically. The *ACRALL* policy in Figure 3 shows the time-based attributes. Decryption keys with time attributes between *1* and *4* will only satisfy this policy, else decryption will fail. However, for a valid key time, the proxy-based server must be able to directly revoke a user using the SPIRC scheme and provide fine-grained access control.

Sequence Flow. Table 4 shows the sequence diagram for the access of the secure Mobile-based Healthfolder. The patient and health professionals personalise their device with credentials and identities on SE. After the HCE tap, they mutually authenticate each other and set a secure session key *Ksym*. The reader device requests to read or write to a $Section_i$. The card device sends the encrypted $Section_i$ along with a challenge *rei*. In the case of an emergency, the emergency professional obtains the Break the Glass (*BTG*) CP-ABE decryption keys *(KDRMabe,KDWMabe)* from the Healthsecure service. The health professional uses the read and write decryption keys to read and write to $Section_i$. After the session terminates, the patient mobile device sends the update for data sync to the digital vault. It also re-encrypts the Healthfolder with the new $Update_i$. After the session, the proxy-based server revokes the *BTG* CP-ABE decryption key for emer-

gency professional the SPIRC scheme.

5.3 Security Analysis

This section presents the security analysis for selective RBAC for Mobile-based Healthfolder.

S1: Confidentiality.
The mobile-based Healthfolder is encrypted by SPIRC and assures selective access by only authorised health professionals to assure confidentiality. SPIRC supports forward secrecy so that on revocation, a revoked user cannot access Healthfolder with his credentials.

S2: Selective read and write access.
Authorised stakeholders access various sections through selective RBAC. Each health professional has a separate CP-ABE decryption key to read and write to different sections and can access them only if the CP-ABE attributes associated with the key satisfies the corresponding access policy.

S3: Revocation.
The SPIRC scheme satisfies all revocation requirements for portable ciphertext *R1-R5* and provides flexibility to retain the secure Mobile-based Healthfolder on patient's mobile device. If an adversary $user_j$ finds Proxy data of another $user_i$, it will not help him with the decryption since each user has a different set of random constants maintained on the proxy-based server. There are however overheads of maintaining a constant set s_i for each $user_i$ on the proxy-based server. With scalable revocation, a patient can share health records across various hospitals and hence get mobility.

S4: Theft of device.
On the loss or theft of a registered device, the proxy-based server revokes the old credentials. Hence adversary cannot use the device. It issues new credentials along with the copy of re-encrypted Healthfolder on the new patient mobile device.

Hence it allows portability of secure health with direct sharing with trusted stakeholders.

S5: Emergency Break The Glass Key.
An emergency person authenticates with the Mobile-based Healthfolder and gets temporary CP-ABE read and write decryption keys from the HealthSecure service to provide emergency care. Later the proxy-based server revokes the emergency keys.

5.4 Implementation and Performance Analysis

Hardware Requirements: SPIRC-based CP-ABE requires around 40 MB of RAM and 1 GHz processor. This

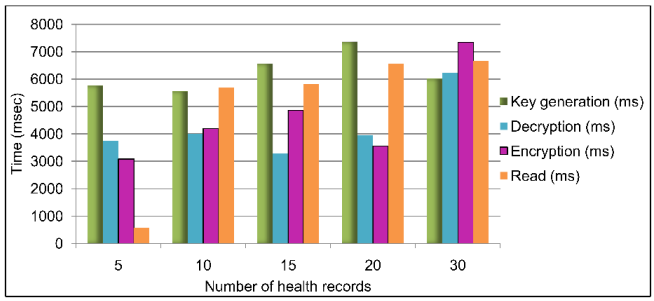

Figure 4: Impact of number of records

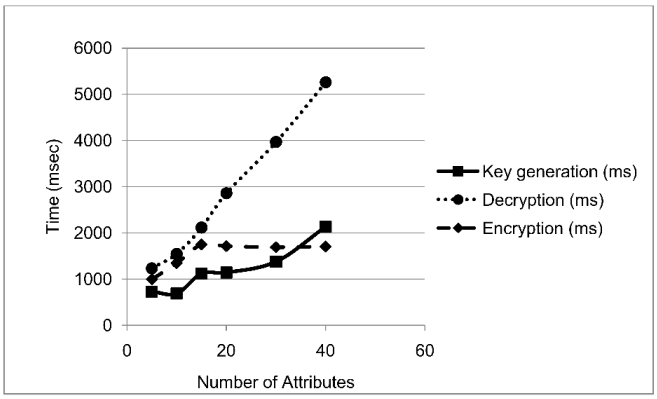

Figure 5: Impact of attributes on access time

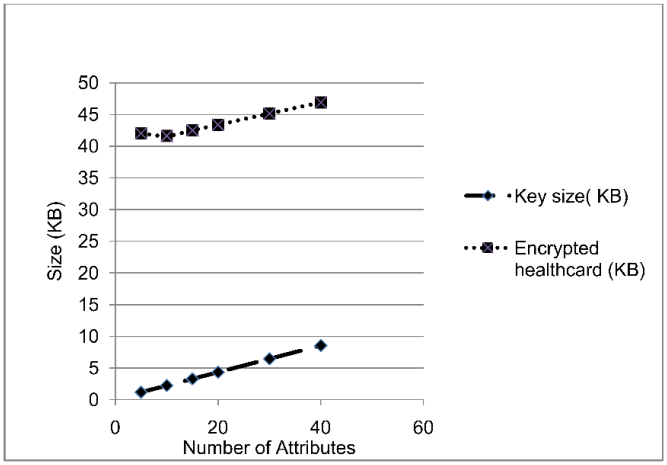

Figure 6: Impact of attributes on storage

Table 5: Average timings

Event	PIRATTE (ms)	SPIRC (ms)
Server Encryption	4197	4197
Proxy decryption	1864	450
Device decryption	2143	2143

Table 6: Comparison for revocation requirements

Requir.	PIRATTE [21]	M-PERMREV [13]	SPIRC (Proposed)
R1	No	Yes	Yes
R2	Yes	Yes	Yes
R3	Yes	Yes	Yes
R4	No	Yes	Yes
R5	Yes	No	Yes

configuration is already available in mid-range smartphones available in developing countries like India in the price range of 100-200 US dollars. CP-ABE has been implemented and tested successfully on Android-based mobile devices such as Samsung Galaxy Nexus device as well as IoT SBS (Single Board Computing) devices [12] such as Raspberry Pi and Intel Galileo.

The mobile-based healthfolder application discussed in the paper is based on NFC which is available currently in mid to high-end devices. NFC is primarily used for initial mutual authentication with the locality of reference as well as to automate pairing of Bluetooth. However, the application can also be deployed on the low-end devices without NFC, by using an alternate proximity technique of scanning secure QR-code using an inbuilt mobile camera to automate Bluetooth. Hence with the growing penetration of mobile devices across the world, the rollout of such a healthcare service in future can enable a rapid transition to health management.

The implementation of the Mobile-based Healthfolder comprises of a JavaScript Object Notation (JSON) file with a list of HL7 health records. The patient mobile device emulates an HCE-based card. A health professional accesses it directly using a reader application on the mobile device. Both mobile-based Healthfolder and the reader applications are implemented using:

- 2 Mid-range Android mobile device such as Sony Xperia M2 devices running Android 5.0.0 (Lollipop) which supports NFC-based HCE.
- Proxy-based CP-ABE scheme SPIRC scheme for selective access
- GO-Trust based secure microSD cards [14]. It includes Java card chip for SE on the microSD card, to store credentials and identities.
- Android SDK and Android Studio
- MongoDB and Python interpreter to maintain the HealthSecure service with Proxy-based Server for SPIRC.

The Health secure service outsources encryption to the proxy-based server. Decryption is performed partially on the user's mobile device and the Proxy-based server. When a patient visits an OPD for a department, typically a doctor reviews the previous health records. This paper assumes that a doctor would review approximate past 10 records at a time to gather the health history for a specific department. Hence, for the evaluation, 10 records are chosen, with an original size of 17KB and encrypted size of 57 KB. Table 5 shows the average timings for the Healthfolder encryption and decryption using PIRATTE and SPIRC scheme. It indicates that the overheads for the security computations for encryption and decryption of Healthfolder for both PIRATTE and SPIRC are similar with acceptable values for usage. SPIRC has lower overheads of proxy decryption as compared to PIRATTE since it associates proxy data with constant values instead of using Lagrange-based secret sharing in PIRATTE. Hence the total decryption time for SPIRC is lower as compared to PIRATTE.

Figure 4 shows the impact of the size of records for encryption, decryption and access time. It indicates that there is a significant increase in time to read as the number of records increase. However, it does not effect the encryption and decryption timings, since the size of ciphertext does not change. An AES key encrypts the Healthfolder, and the CP-ABE key is used to encrypt the AES key which remains constant. The read time comprises of communication time and decryption time to view the records. Since the communication time increases with the number of health records, the read time also increases. Figure 5 shows the increase in the timings for key generation, encryption and decryption with the increase in the number of attributes. Figure 6 illustrates that the increase in the number of attributes does not affect the storage size of encrypted Healthfolder. However, similar to CP-ABE, the key size increases with the increase in the number of attributes.

6 Performance Comparison

Table 6 illustrates the comparison of the different revocation techniques for the revocation requirements. Only SPIRC fulfils all the requirements *R1-R5*. Hence it is suitable for secure and selective access of a portable ciphertext and provides ease of usage to the owner and other non-revoked users.

Table 7 shows the comparison of CP-ABE techniques for overheads of storage and computational overheads of encryption and decryption. It also describes the terms used for comparison. The comparison assumes that all CP-

Table 7: Comparison of storage and performance

Scheme	CP-ABE [8]	PIRATTE [21]	M-PERMREV [13]	SPIRC (Proposed scheme)
Size of keys and ciphertext in different schemes				
PK	$2L_{G1} + L_{G2} + L_{GT}$	$2L_{G1} + L_{G2} + L_{GT}$	$2L_{G1} + L_{G2} + L_{GT}$	$2L_{G1} + L_{G2} + L_{GT}$
MK	$L_{G1} + L_{Zp}$	$L_{G1} + (1+t)L_{Zp}$	$L_{G1} + 2L_{Zp}$	$L_{G1} + L_{Zp}$
SK	$L_{G2} + (a + L_{G1} + L_{G2})\|A_U$	$L_{G2} + (a + L_{G1} + 2L_{G2})\|A_U$	$L_{G2} + (a + L_{G1} + L_{G2})\|A_U$	$L_{G2} + (a + L_{G1} + 2L_{G2})\|A_U$
CT	$(2\|A_C\| + 1)L_{G1} + L_{G2}$	$(2\|A_C\| + 1)L_{G1} + L_{G2}$	$(2\|A_C\| + 1)L_{G1} + L_{G2}$	$(2\|A_C\| + 1)L_{G1} + L_{G2}$
Broadcast	None	$tZ_p + \|A_U\|L_{G2} + Z_p$	$L_{G1} + L_{G2}$	$Z_p + \|A_U\|L_{G2}$
Comparison of computational overhead				
Encrypt.	$(2A_C + 1)G_1 + G_2$	$(2A_C + 1)G_1 + G_2$	$(2A_C + 1)G_1 + G_2$	$(2A_C + 1)G_1 + G_2$
Decrypt.	$2A_UC_e + (2\|S\| + 2)G_2$	$3A_UC_e + (2\|S\| + 2)G_2$	$2A_UC_e + (2\|S\| + 3)G_2$	$3A_UC_e + (2\|S\| + 2)G_2$

A_C: Attributes of ciphertext C; A_U: Attributes of user U; a: Length of an attribute; C_e: Number of bilinear pairings

G_i: Group or operations in group i, $i = 1$ or 2; S: Least interior nodes satisfying access structure (including root node); L*: Bit length of element in *; t number of users to be revoked

ABE schemes use asymmetric group pairing. All schemes have similar lengths for public key *PK*. However, the master key *MK* is shorter in SPIRC as compared to PIRATTE [21] since there is no generation of polynomial *P*. Both PIRATTE and SPIRC have similar lengths for private key *SK*, but which is longer as compared to the Bethencourt *et al.*'s CP-ABE [8] and M-PERMREV [13] schemes (both have same lengths for *SK*). *SK* is dependent on the number of attributes A_U allocated to the user *U*. The ciphertext length is dependent on the number of attributes of ciphertext A_C and is the same for all schemes. There is no broadcast overhead for Bethencourt *et al.*'s CP-ABE scheme [8]. The Broadcast overhead for PIRATTE is dependent on the number of revoked users and the number of attributes of a user A_U. The broadcast overhead of M-PERMREV is constant since it is only a state update for a user. However, it links a separate user state for each ciphertext and does not satisfy revocation requirement *R5*. SPIRC broadcasts a constant value for proxy data and is independent of the number of revoked users and dependent only on the number of attributes of a user A_U. Also unlike M-PERMREV, the proxy data is not be linked with the ciphertext such that there is an overhead of creating separate proxy data for each ciphertext for a user. The encryption time is dependent on the number of attributes in the ciphertext A_C and is similar to all schemes. The decryption time for PIRATE and SPIRC is higher as compared to the decryption time for CP-ABE and M-PERMREV, due to an extra bilinear pairing for proxy-base decryption. Due to simpler proxy data generated, the overall decryption time for SPIRC is smaller as compared to PIRATTE.

7 Conclusion and Future Work

Portable devices such as mobile devices can retain critical data encrypted with CP-ABE for fine-grained selective access control. This paper proposes a novel SPIRC scheme which improves PIRATTE [20] for scalable revocation. It satisfies all the revocation requirements *R1-R5* for ease of maintenance of ciphertext on a portable device. The overheads for generation of the master key and broadcast data as lower as compared to the PIRATTE scheme. Also, SPIRC does not associate the proxy data with the ciphertext as in the M-PERMREV scheme [13].

The paper presents a case study for using SPIRC for sharing secure portable Mobile-based Healthfolder with various health professionals over NFC as a contactless card. The Mobile-based Healthfolder is a next generation future Healthcard which can provide highly available and secure dispersed health records. The healthfolder can be shared with multiple health professionals using selective RBAC and provides mobility of patient across various hospitals. With the reduction in prices and the increase in penetration of mobile devices across the world, they can assist in the secure portable management of health data in emerging countries like India. The paper also successfully demonstrates the implementation and evaluation of a prototype of SPIRC for Mobile-based Healthfolder on mid-range Android devices with acceptable overheads for security and access.

Our work is the first novel attempt to address secure data on a portable device using Bethencourt *et al.*'s CP-ABE [8] with scalable user revocation. SPIRC can provide multi-user selective access to IoT devices such as users with different roles in a family access a car with their mobile devices to lock/unlock, configuration setup and access logs using selective RBAC.

The SPIRC scheme can be enhanced in future for supporting single key authority and multi-key authority delegation as well as attribute revocation. In future, we can compare SPIRC with other schemes such as those by Ibraimi *et al.* [19] using provably secure CP-ABE scheme [10] and by Lewko *et al.* [7] based on LSSS. Since mobile devices are vulnerable to security threats, the security scheme must assure that they are not susceptible to malware [1] and have trustful states. We can also use

secure smart card-based authentication [18] using Secure Element on a mobile device.

References

[1] A. Abdullah, Al-khatib, and A. W. Hammood, "Mobile malware and defending systems: Comparison study," *International Journal of Electronics and Information Engineering*, vol. 6, pp. 116–123, 2017.

[2] M. Alattar and M. Achemlal, "Host-based card emulation: development, security,and ecosystem impact analysis," in *Proceedings of IEEE High Performance Computing and Communications,* Aug. 2014.

[3] M. Ambrosin, M. Cont, and T. Dargahi, "On the feasibility of attribute-based encryption on smartphone devices," in *Proceedings of ACM IoT challenges in Mobile and Industrial Systems (IoY-Sys'15)*, pp. 49–54, 2015.

[4] N. Anciaux, M. Berthelot, L. Braconnier, *et al.*, "A tamper-resistant and portable healthcare folder," *International Journal of Telemedicine and Applications*, vol. 2008, 2008.

[5] N. Attrapadung and H. Imai, "Attribute-based encryption supporting direct/indirect revocation modes," in *Cryptography and Coding*, LNCS 5921, pp. 278–300, Springer, 2009.

[6] N. Aziz, N. Udzir, and R. Mahmod, "Extending TLS with mutual attestation for platform integrity assurance," *Journal of Communications*, vol. 9, pp. 63–72, 2014.

[7] A. Lewko, T. Okamoto, A. Sahai, *et al.*, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Advances in Cryptology (EUROCRYPT'10)*, LNCS 6110, pp. 62–91, Springer, 2010.

[8] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334, 2007.

[9] W. Chen, C. Hsu, Y. Lee, W. Jian, and H. Rau, "Developing electronic health records in taiwan," *IEEE IT Professional*, vol. 12, no. 2, pp. 17–25, 2010.

[10] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*, pp. 456Ũ465, 2007.

[11] V. Coskun, B. Ozdenizci, and K. Ok, "A survey on near field communication (nfc) technology," *Wireless Personal Communications*, vol. 71, pp. 2259–2294, 2013.

[12] A. Dmitrienko, Z. Hadzic, H. Löhr, A. Sadeghi, and M. Winandy, "On the feasibility of attribute-based encryption on internet of things devices," *IEEE Access*, vol. 36, pp. 25–35, 2016.

[13] S. Dolev, N. Gilboa, and M. Kopeetsky, "Permanent revocation in attribute based broadcast encryption," in *Proceedings of IEEE Cyber Security*, Dec. 2012.

[14] Go-trust, *Go-trustő Secure Microsd java*, Jan. 11, 2018. (`http://www.go-trust.com/products/microsd-java/`)

[15] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of ACM Computer and Communications Security (CCS'06)*, pp. 89–98, Oct. 2006.

[16] HL7, *Introduction to HL7 Standards*, Jan. 11, 2018. (`http://www.hl7.org/implement/standards/`)

[17] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, pp. 1214Ũ1221, 2011.

[18] C. Y. Tsai, C. S. Pan, M. S. Hwang, "An improved password authentication scheme for smart card," in *Proceedings of International Conference on Intelligent and Interactive Systems and Applications*, Nov. 2016.

[19] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *Information Security Applications*, LNCS 5932, pp. 309–323, Springer, 2009.

[20] S. Jahid, P. Mittal, and N. Borisov, "Easier: Encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pp. 411–415, 2011.

[21] S. Jahid, P. Mittal, and N. Borisov, ""PIRATTE: Proxy-based immediate revocation of attribute-based encryption"," in *arXiv preprint arXiv*, pp. 1–14, 2012.

[22] C. Lee, P. Chung, and M. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments," *International Journal of Network Security*, vol. 15, pp. 231–240, 2013.

[23] M. Li, Logan S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, pp. 131–143, 2012.

[24] C. Liu, W. Hsien, C. Yang, and M. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage," *International Journal of Network Security*, vol. 18, pp. 900–916, 2016.

[25] J. Modi, M. Prajapati, A. Sharma, R. Ojha, and D. Jinwala, "A secure communication model for expressive access control using cp-abe," *International Journal of Network Security*, vol. 19, pp. 193–204, 2017.

[26] S. Narayan, M. Gagné, and R. Safavi-Naini, "Privacy preserving ehr system using attribute-based infrastructure," in *Proceedings of ACM Workshop on Cloud Computing Security Workshop*, pp. 47–52, 2010.

[27] Oracle, *Java Card Platform Security*, Jan. 11, 2018. (`http://www.oracle.com/technetwork/java/javacard/`)

[28] L. Pang, J. Yang, and Z. Jiang, "A survey of research progress and development tendency of attribute-based encryption," *The Scientific World Journal*, vol. 2014, 2014.

[29] D. Sethia, D. Gupta, and H. Saran, "Security framework for portable nfc mobile based health record system," in *Proceedings of the 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'16)*, Oct. 2016.

[30] TCG, *Trusted Computing Group*, Jan. 11, 2018. (`https://www.trustedcomputinggroup.org/`)

[31] The National Archives, *U.k. National Health Service (NHS). Spine - NHS, Connecting for Health*, Jan. 11, 2018. (`http://www.connectingforhealth.nhs.uk`)

[32] Y. Tian, Y. Peng, G. Gao, and X. Peng, "Role-based access control for body area networks using attribute-based encryption in cloud storage," *International Journal of Network Security*, vol. 19, pp. 720–726, 2017.

[33] S. Y. Worcester, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of IEEE 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS'20)*, pp. 261–270, 2010.

Biography

Divyashikha Sethia Received B.Tech degree in Computer Science and Engineering from Maharaja Sayajirao University of Baroda in 1997 and M.Tech degree in Computer Science and Engineering from IIT Delhi in 2006. She has previously worked as a Software Engineer in the telecom industry in California, USA. She is currently an Assistant Professor in Department of Computer Science and Engineering at Delhi Technological University. Her research interests include distributed systems, computer networks, mobile computing and telemedicine

Huzur Saran Received B.Tech degree in Electrical Eng., Indian Institute of Technology, Delhi in 1983 and PhD in the Computer Science University of California, Berkeley in 1990. Dr Saran has also led many other entrepreneurial technology development efforts such as Solidcore Inc., which was purchased by Mcafee for their Dynamic Whitelisting technology developed entirely in India under Prof Saranąęs Leadership. He is currently professor of Computer Science at IIT Delhi and Heads the Amar Nath and Shashi Khosla School of Information Technology, at IIT Delhi. His research interests include Computer Systems, Security and Algorithms.

Daya Gupta received her M.Sc.(Computer Science) at I.I.T Delhi, and PhD in Computer Science from the University of Delhi. She is Professor in Department of Computer Science and Engineering at Delhi Technological University. She is a Senior Member, I.E.E.E. Her research interests include Information Security, Requirements Engineering, Swarm Intelligence, Nature Inspired Algorithm, Data Warehouse and Data Mining.