Smart, Secure and Seamless Access Control Scheme for Mobile Devices

Yogachandran Rahulamathavan*, Muttukrishnan Rajarajan * and Raphael C.-W. Phan†
*School of Engineering and Mathematics and Computer Science, City University London, U.K.
†Faculty of Engineering, Multimedia University, Malaysia
Email:*{yogachandran.rahulamathavan.1, r.muttukrishnan}@city.ac.uk, †raphael@mmu.edu.my,

***Abstract*—Smart devices capture users' activity such as unlock failures, application usage, location and proximity of devices in and around its surrounding environment. This activity information varies between users and can be used as digital fingerprints of the users' behaviour. Traditionally, users are authenticated to access restricted data using long term static attributes such as password and roles. In this paper, in order to allow secure and seamless data access in mobile environment, we combine both the user behaviour captured by the smart device and the static attributes to develop a novel access control technique. Security and performance analyse show that the proposed scheme substantially reduces the computational complexity while enhances the security compared to the conventional schemes.**

I. Introduction

Recently Google introduced a new authentication technique called *smart lock* for its Android Lollipop mobile operating system. This new feature provides seamless access to the device without compromising the security i.e., devices are secured from intruders but also easy to unlock when a security check has been made. This smart lock allows Android based smart phones to bypass lock screen security when it is connected to *trusted devices* via Bluetooth or NFC or located in *trusted locations*[1]. Smart devices are now embedded with numerous sensors and capable of capturing more sophisticated information about the user's day-to-day activity on top of the trusted locations and trusted devices such as app usage, charging pattern, temperature, lighting, etc [1], [2].

The aim of this paper is to exploit the above sensor data to develop a novel seamless data sharing technique for mobile devices. In general, the data sharing involves three parties i.e., owners, (mobile) users, and storage server. Traditionally, we assume these three are in the same domain and also that the server is fully trusted. In the recent trend towards cloud computing and outsourcing environments, we cannot rely on the traditional data sharing setup since the data confidentiality is not guaranteed within third party cloud environment. To overcome this challenge, the data confidentiality in a cloud environment is achieved via attribute based encryption (ABE) technique [3], [4], [5], [6].

ABE is considered as a promising cryptographic technique and supports both the data confidentiality and access control simultaneously [3], [4], [5], [6]. Using ABE, the data owners can encrypt the data using fine-grained access policies. For instance, let us assume, an employer uploads encrypted file to the cloud using ABE, where access policy of that file is defined using the following attributes and functions AND and OR: "Manager" OR "Finance Office" AND "Company A". Hence, an employee who is a "Manager" employed at "Company A" can decrypt the file. The data confidentiality in the ABE schemes relies only on predefined static attributes such "Manager", "Finance Office", and "Company A". Similar to the smart lock feature, we note that it is also possible to incorporate the device sensor data i.e., trusted locations, devices etc (lets call them as dynamic attributes) within ABE to provide seamless data sharing within mobile environment. This feature not only improves the user experience but also the security. Because if we consider the previous example, where an employee has the long term credentials for the following attributes: "Manager", and "Company A". Hence, she can access the encrypted file while she is traveling in public transport using her personnel mobile device. However, the risk level associated in this context is high. In fact, people in her proximity might easily see confidential data via shoulder surfing. It is also possible for an adversary to steal the employee's mobile device, and get unauthorized access to the corporate data if there is no real-time verification (assuming that the credentials for static attributes are stored within mobile). Hence, evaluating the data collected by smart device's sensors in real-time provides additional layer of security.

[1]https://support.google.com/nexus/answer/6093922?hl=en-GB

II. Related Work

ABE was firstly proposed by Sahai and Waters in [3]. At high level, ABE can be divided into two: single authority and multi-authority. In a single authority scenario, there is only one attribute authority (AA) which monitors all attributes of users and issues encryption and decryption credentials for the data owners and users. This single authority becomes a fully trusted party to which the users have to prove their attributes in order to obtain the decryption credentials. In such a case, the AA has too much power and it can decrypt all the data and knows about all the users' attributes. This is one of the limitations in single authority ABE scheme. In order to overcome this single point-of-failure, multi-authority ABE schemes were proposed in [7], [8]. Chase et. al. [7] presented a MA-ABE scheme, which allows any polynomial number of independent authorities to monitor attributes and distribute decryption keys. However [7] requires a central authority to monitor all other AAs; hence vulnerable for single point of failure similar to single AA schemes. In order to fully

decentralise the MA-ABE scheme, Chase and Chow proposed a decentralised scheme by removing the central authority [8]. However the scheme in [8] is not fully decentralised since if an AA joins or leaves the system then the whole ABE system needs to reboot by assigning new keys to users. Lewko and Waters [9] proposed a fully decentralized ABE scheme, where users could have zero or more attributes from each AA and do not require a trusted server. In their work, the AA can join and leave freely without re-initializing the system.

However, the above (centralised) ABE schemes do not support user privacy i.e., users need to reveal their identities to AAs before retrieving credentials for their attributes. In order to overcome this limitation, a privacy-preserving MA-ABE scheme was proposed by Han et al. in [11]. In contrast to the existing decentralized ABE schemes, the scheme in [11] preserves the user privacy. In [11], the Global Identifier (GID) of the user is used to tie all the decryption keys together, where a blind key generation protocol has been used to issue the decryption keys. Hence, corrupted AAs cannot pool the users' attributes by tracing the GIDs' of the users from the decryption keys.

None of the above ABE schemes exploit the features of smart phones or devices to support smart access control framework. The work in [10] combined the centralised ABE schemes [7], [8] and features of smart devices to enhance the data access in the mobile environment. Since the centralised ABE schemes do not preserve the users' privacy, the same privacy vulnerability exist in [10]. Since the smart devices extract substantial amount users' activity information, any solution without privacy protection will endanger the adoption of new technology. In order to overcome this issue, we propose a new scheme in this paper which incorporates smart devices' dynamic attributes with the static attributes of privacy-preserving decentralised ABE scheme. To achieve this, we modify the scheme in [11] in order to appropriately incorporate the dynamic attributes and to avoid user collusion [12]. Later in this paper, we perform extensive numerical analysis and security analysis to show that the proposed scheme is quicker and secure.

A. Preliminaries

Bilinear Pairings: Let $\mathbb{G}_1, \mathbb{G}_2$ be two multiplicative groups of prime order q and let g_1 and g_2 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Let us denote a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. The map has the following three properties:

1) Bilinearity: $\forall x \in \mathbb{G}_1, \forall y \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_q$, there is $\hat{e}(x^a, y^b) = \hat{e}(x, y)^{ab}$.
2) Non-degeneracy: For $\forall x \in \mathbb{G}_1, \forall y \in \mathbb{G}_2$, there is $\hat{e}(x, y) \neq 1$.
3) Computability: $\hat{e}$ is an efficient computation.

Lagrange Interpolation: Shamir's secret share uses Lagrange interpolation technique to obtain the secret from shared-secrets. Suppose that $p(x) \in \mathbb{Z}_p[x]$ is a $(k-1)$ degree polynomial and secret $s = p(0)$. Let us denote $S = \{x_1, x_2, \ldots, x_k\}$ and the Lagrange coefficient for x_i in S as

$$\Delta_{x_i,S}(x) = \prod_{x_j \in S,\ x_j \neq x_i} \frac{x - x_j}{x_i - x_j}.$$

For a given k different number of values $p(x_1)$, $p(x_2)$, ..., $p(x_k)$, the polynomial $p(x)$ can be reconstructed as follows,

$$p(x) = \sum_{x_i \in S} p(x_i) \prod_{x_j \in S,\ x_j \neq x_i} \frac{x - x_j}{x_i - x_j} = \sum_{x_i \in S} p(x_i)\Delta_{x_i,S}(x),$$

hence the secret s can be obtained as:

$$s = p(0) = \sum_{x_i \in S} p(x_i) \prod_{x_j \in S,\ x_j \neq x_i} \frac{0 - x_j}{x_i - x_j}.$$

Mapping: We consider a linear comparison function namely mapping denoted as M. This function takes two inputs: one from smart device sensors (i.e., location) and the second one from the data owner. Data owner must embed the required sensor data and boolean operations. The output of the mapping function is "yes" or "no" by comparing both the inputs. For example, this function can extract data from smart device's GPS module and compare with locations in the data owner's input and output "yes" or "no" i.e., $M(\text{“}data\ from\ GPS\text{”} = \text{“}London\text{”}) = no$. It should be noted that this function can be embedded securely within any smartphone apps which calls required sensors at device level. Similar to the XACML policy language, the data owner can even define a range of values for sensor data. However the data owner needs to embed a boolean function with the data to expedite the comparison process.

III. Problem Statement

A. System Architecture

The proposed system consists of the following four components: mobile users (or employees), data owner (or organizations), cloud service provider, and AAs.

Mobile Users are equipped with one or more smart devices.

Data Owners upload the encrypted data to the cloud storage and define access policies. In our scheme, the data owner defines access policies based on static attributes obtained from AA together with dynamic attributes. Thus, the data owner can decide who is able to decrypt the data from where and in what circumstances.

Cloud Service Providers provide cloud storage and computational power to both the users and data owners. In our scheme, we assume that data owner will upload the encrypted data to the storage while the user will download the encrypted data from the storage.

Attribute Authorities manage and maintain static attributes of the users. Different authorities manage different sets of attributes. A user needs to prove her attributes to the authorities in order to receive the decryption key for each attribute from corresponding authority. In the proposed scheme, it should be noted that for the dynamic attributes, there is no need to have an AA to distribute encryption and decryption credentials e.g., for location attribute the device uses GPS module to obtain longitude and latitude.

B. Decentralized Attribute Based Encryption

In this section, we present a variant of the decentralized KP-ABE scheme [11]. In a decentralized scheme, it is not necessary to maintain a fixed number of AAs. Any AA can join and/or leave the system at any time without rebooting the

system. First of all, we will explain sub-algorithms followed by the privacy-preserving decentralized KP-ABE in Fig. 1.

C. Sub-algorithms

The decentralised ABE contains five sub-algorithms namely global setup, authority setup, key issuing, encryption and decryption. Let us briefly explain the functionalities of each sub-algorithm.

Global Setup: This algorithm takes a security parameter as input and output system parameters. These system parameters can be used by authorities who join the system.

Authority Setup: Each AA uses the system parameters obtained from the global setup to generate public and private keys for the attributes it maintains.

Key Issuing: User and AA interact via anonymous key issuing protocol in order to determine a set of attributes which belongs to the user. Then AA generates decryption credentials for those attributes and sends them to the user.

Encryption: The encryption algorithm takes a set of attributes maintained by AA and the data as input. Then it outputs the ciphertext of the data.

Decryption: The decryption algorithm takes the decryption credentials received from AAs and the ciphertext as input. The decryption will be successful if and only if the user attributes satisfy the access structure.

Detailed description of each step involved in Fig. 1 is provided after Fig. 2.

D. Security and Privacy Threats

There are a number of known security and privacy threats which hinder the access control schemes. Let us provide a list of potential attacks and relate them to the requirements of the system.

- **Identity-related threats**: The main threat we need to consider is related to the identities of the elements involved in the protocol. The adversary might impersonate as one of the entities and try to establish a connection to a legitimate entity.
- **Collusion attacks**: Each AA manages a set of attributes in our system, hence, authorities can collude with each other to infer user attributes. This will allow the malicious authorities to profile the user based on different sets of attributes that the user has with the malicious authorities. Similarly, two users can collide and get access to data which are not accessible by the users individually.
- **Dynamic attribute cheating**: Smart devices capture dynamic attributes such as unlock failures, app usage, location and near by devices using the local sensors. It is crucial for the data owner to ensure that the user's device is not modified (rooted) or tampered in order to feed false dynamic attributes. Hence, we need to consider a set of technologies in order to guarantee that no attribute cheating is possible.

We explain how our new algorithm mitigates all these threats in the security and privacy analysis section later in this paper.

IV. Dynamic and Static Attribute-Based Encryption Scheme

A. Proposed Scheme

In contrast to the conventional ABE scheme described in Fig. 1, we will show in this section how to efficiently incorporate the dynamic attributes to the conventional ABE scheme, where the data owner can encrypt the data by not only using the credentials obtained from the AA, but also using dynamic attributes such as location, time, temperature, noise, light, the presence of other devices, a particular interaction between the user and the smartphone, or a combination of these. Similar to the conventional ABE, the proposed algorithm is also composed of four sub-algorithms namely setup, key issuing, encryption, and decryption. The proposed algorithm is given in Fig. 2.

Let us consider a system which contains N number of AAs (i.e., we denote them as $A_1, \ldots, A_N$). The attribute set managed by the authority A_k is denoted as $\widetilde{A_k} = \{a_{k,1}, \ldots, a_{k,n_k}\}\ \forall k$. Each AA is also assigned a value d_k i.e., user must have at least d_k number of attributes of this authority in order to retrieve the private key associated with this AA. Initially, for a given security parameter λ, global setup algorithm ($\mathcal{GS}$) generates the bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order p i.e., $\{\mathbb{G}_1, \mathbb{G}_2\} \leftarrow \mathcal{GS}(1^\lambda)$. The authority setup algorithm ($\mathcal{AS}$) is executed by each AA to randomly generate public keys (PK) and the corresponding private keys (SK). The public-private key pairs for A_k is given as $\{(Y_k, Z_k, [T_{k,1}, \ldots, T_{k,n_k}]), (\alpha_k, \beta_k, [t_{k,1}, \ldots, t_{k,n_k}])\}$.

Let us denote the attribute set belonging to user u as $\widetilde{A_u}$ and the common attribute set between user u and authority k as $\widetilde{A_u^k}$ i.e., $\widetilde{A_u^k} = \widetilde{A_u} \bigcap \widetilde{A_k}$. Key generation ($\mathcal{KG}$) algorithm will be used to issue decryption keys to the user u with a set of attributes $\widetilde{A_u}$ as shown in Fig. 2.

Let us denote the set of static attributes used to encrypt message m as $\widetilde{A_m}$ and the common attribute set between message m and the authority k as $\widetilde{A_m^k}$ i.e., $\widetilde{A_m} = \{\widetilde{A_m^1}, \ldots, \widetilde{A_m^k}, \ldots, \widetilde{A_m^N}\}$. Denote the dynamic attribute as $A_C = \{a_{a,1}, \ldots, a_{a,n}\}$ where $a_{a,i}$. Let us also denote the index set of authorities involved in the ciphertext of message m as I_c. The encryption algorithm ($\mathcal{E}$) encrypts the message $m \in \mathbb{G}_2$ using an attribute set $\widetilde{A_m}$. The data owner also computes $M(a_{a,1})||M(a_{a,2})|| \ldots ||M(a_{a,n}$. In order to encrypt the message, the message owner randomly generates s and computes ciphertext $C = \left[C_1, C_2, C_3, C_{k,j}, \forall a_{k,j} \in \widetilde{A_m^k}\right]$. If user has decryption keys for the attributes of message m then he can obtain the message m from the ciphertext using the following four steps by executing the decryption algorithm ($\mathcal{D}$)as shown in Fig. 2.

The novelty in our scheme compared to the conventional ABE scheme lies in the encryption and the decryption sub-algorithms in Fig. 2. For the sake of simplicity, let us consider the following three dynamic attributes: $a_{a,1}$ ="location", $a_{a,2}$ ="risk level associated with her recent app usage" and $a_{a,3}$ ="unlock failures in last two days". Now the data owner defines $A_C = \{a_{a,1} = \text{“}LONDON\text{”}, a_{a,2} < \text{“3”}$ and $a_{a,3} < \text{“2”}\ \}$ and computes $C_1 =$

Global Setup $\mathcal{GS}$ For a given security parameter λ, $\mathcal{GS}$ generates the bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order p as follows: $\{\mathbb{G}_1, \mathbb{G}_2\} \leftarrow \mathcal{GS}(1^\lambda)$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ be a bilinear map and g, h and h_1 be the generators of $\mathbb{G}_1$ such that $\forall\ x, y \in \mathbb{G}_1$ and $\forall\ a,\ b \in \mathbb{Z}_p$, $e(x^a, y^b) = e(x, y)^{ab}$. There are N number of authorities $\{A_1, \ldots, A_N\}$: A_k monitors n_k attributes i.e. $\tilde{A}_k = \{a_{k,1} \ldots, a_{k,n_k}\}, \forall k$.
Authorities Setup $\mathcal{AS}$ Security parameters of A_k: $SK_k = \{\alpha_k, \beta_k,$ and $[t_{k,1}, \ldots, t_{k,n_k}]\} \xleftarrow{R} \mathbb{Z}_p, \forall k$. Public parameters of A_k: PK_k = $\{Y_k = e(g,g)^{\alpha_k}$, $Z_k = g^{\beta_k}$, and $[T_{k,1} = g^{t_{k,1}}, \ldots, T_{k,n_k} = g^{t_{k,n_k}}]\}$, $\forall\ k$. A_k specify m_k as minimum number of attributes required to satisfy the access structure ($m_k \leq n_k$).
Key Generation $\mathcal{KG}$ Collision-Resistant Hash Function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ to generate u from the user global identity. Attribute set of user is $\tilde{A}_u$: $\tilde{A}_u \cap \tilde{A}_k = \tilde{A}_u^k\ \forall\ k$. A_k generates $r_{k,u} \in_R \mathbb{Z}_p$ and polynomial q_x for each node x (including the leaves) $\mathbb{T}$. For each node x, the degree d_x of the polynomial q_x is $d_x = k_x - 1$ where k_x – threshold value of that node. Now, for the root node r, set $q_r(0) = r_{k,u}$. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$. Now decryption keys for the user u are generated as follows:
$$D = \left[D_{k,u} = g^{-\alpha_k} h^{\frac{\beta_k}{r_{k,u}+u}} h_1^{\frac{r_{k,u}}{\beta_k+u}}, D_{k,u}^1 = h^{\frac{1}{r_{k,u}+u}}, D_{k,u}^j = h_1^{\frac{q_{a_{k,j}}(0)}{(\beta_k+u)t_{k,j}}}, \forall a_{k,j} \in \tilde{A}_u^k \right].$$
Encryption $\mathcal{E}$ Attribute set for the message m is $\tilde{A}_m$: $\tilde{A}_m \cap \tilde{A}_k = \tilde{A}_m^k, \forall k$, i.e. $\tilde{A}_m = \{\tilde{A}_m^1, \ldots, \tilde{A}_m^k, \ldots, \tilde{A}_m^N\}$. The data owner of message m randomly chooses $s \in_R \mathbb{Z}_p$, and outputs the ciphertext as follows:
$$C = \left[C_1 = m. \prod_{k \in I_C} e(g,g)^{\alpha_k s}, C_2 = g^s, C_3 = \prod_{k \in I_C} g^{\beta_k s} \text{and} \left\{ C_{k,j} = T_{k,j}^s \right\}_{\forall k \in I_C, a_{k,j} \in \tilde{A}_m^j} \right]$$ where I_c denotes the index set of the authorities.
Decryption $\mathcal{D}$ In order to decrypt C, the user u, computes X, Y and Q_k as follows: $\left[X = \prod_{k \in I_C} e\left(C_2, D_{k,u}\right),\ Y = e\left(C_3, D_{k,u}^1\right) \text{and } S_k = \prod_{a_{k,j} \in A_m^k} e\left(C_{k,j}, D_{k,u}^j\right)^{\Delta_{a_{k,j}, \tilde{A}_m^j}(0)} \right]$. User then decrypts the message m as follows: $m = \frac{C_1 X}{Y \prod_{k \in I_C} S_k}$.

Fig. 1. The decentralized key-policy attribute-based encryption scheme [11].

Global Setup $\mathcal{GS}$ same as conventional method.
Authorities Setup $\mathcal{AS}$ same as conventional method.
Key Generation $\mathcal{KG}$ same as conventional method.
Encryption $\mathcal{E}$ Attribute set for the message m is $\tilde{A}_m$: $\tilde{A}_m \cap \tilde{A}_k = \tilde{A}_m^k, \forall k$, i.e. $\tilde{A}_m = \{\tilde{A}_m^1, \ldots, \tilde{A}_m^k, \ldots, \tilde{A}_m^N\}$.
Data owner chooses a set of dynamic attributes and computes $h(M(a_{a,1})||M(a_{a,2})|| \ldots ||M(a_{a,n}))$
Data owner of message m randomly chooses $s \in_R \mathbb{Z}_p$, and outputs the ciphertext as follows:
$C_1 = m.h(M(a_{a,1})||M(a_{a,2})|| \ldots ||M(a_{a,n})). \prod_{k \in I_C} e(g,g)^{\alpha_k s}$,
$C_2 = g^s, C_3 = \prod_{k \in I_C} g^{\beta_k s}$ and $\left\{ C_{k,j} = T_{k,j}^s \right\}_{\forall k \in I_C, a_{k,j} \in \tilde{A}_m^j}$
where I_c denotes the index set of the authorities.
Hence the date owner uploads the ciphertext $C = [C_1, C_2, C_{k,j}]$ into the cloud.
Decryption $\mathcal{D}$ User downloads C from the cloud and checks the required attributes to decrypt m.
In order to decrypt C, the user u, computes X, Y and Q_k as follows:
$X = \prod_{k \in I_C} e\left(C_2, D_{k,u}\right)$,
$Y = e\left(C_3, D_{k,u}^1\right)$ and $S_k = \prod_{a_{k,j} \in A_m^k} e\left(C_{k,j}, D_{k,u}^j\right)^{\Delta_{a_{k,j}, \tilde{A}_m^j}(0)}$.
Now the mobile device computes $h(M(a'_{a,1})||M(a'_{a,2})|| \ldots ||M(a'_{a,n}))$.
User can decrypt the data as follows (only if $h(M(a'_{a,1})||M(a'_{a,2})|| \ldots ||M(a'_{a,n})) = h(M(a_{a,1})||M(a_{a,2})|| \ldots ||M(a_{a,n}))$)
User then decrypts the message m as follows: $m = \frac{C_1 X}{h(M(a'_{a,1})||M(a'_{a,2})||\ldots||M(a'_{a,n}))Y \prod_{k \in I_C} S_k}$.

Fig. 2. The proposed scheme.

$m.h(M(a_{a,1})||M(a_{a,2})|| \ldots ||M(a_{a,n})). \prod_{k \in I_C} e(g,g)^{\alpha_k s}$. Let us assume that the risk level varies between 1 to 10 where higher risk denoted by larger value. However, different organizations may define the risk level based on their own standards. For example, if a particular document is highly classified then, the organization sets high risk value for that document rather than ordinary documents.

In the decryption phase the mapping function, M, which inputs data owners dynamic attribute requirements and smart device readings and output $''yes''$ or $''no''$, e.g., if the current risk level is less than the threshold defined by the data owner then $M(a_{a,2} < \text{“3”}) = yes$. This ensures that even if a user has all the credentials from AA, dynamic attributes enforced by the data owner must be satisfied before the decryption. If $h(measured\ value) = h(data\ owners\ predefined\ values)$ then decryption will be successful.

V. Performance Analysis

In this section, we compare the computation costs associated with the proposed scheme in Fig. 2 against the conventional scheme in Fig. 1. In the proposed or in the conventional scheme, the user is involved in the computation during the decryption step and the data owner is involved in the encryption step. We can ignore the computational costs involved in the setup and key-issuing steps since these can be done off line. Since the computational cost for hash function is negligible compared to pairing and exponentiation, let us denote the computational time (in ms) for one multiplication, one exponentiation, and one pairing as C_m, C_{ex}, and C_p, respectively.

For comparison, we use the benchmark time values given in the popular pairing-based cryptography library namely jPBC [2]. Table I shows the time values (in ms) for C_m, C_{ex}, and C_p

[2]The Java Pairing-Based Cryptography Library (JPBC), http://tinyurl.com/ll2p39t

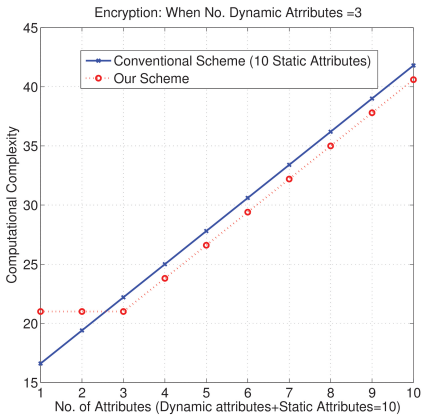

Fig. 3. Complexity comparison for encryption for three dynamic attributes.

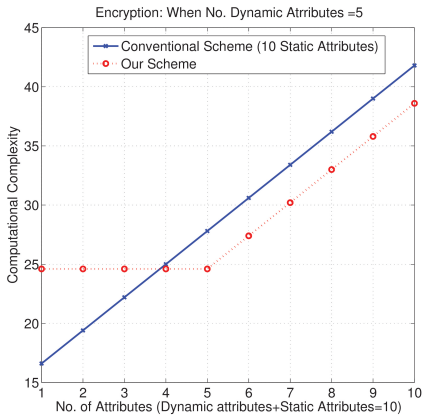

Fig. 4. Complexity comparison for encryption for five dynamic attributes.

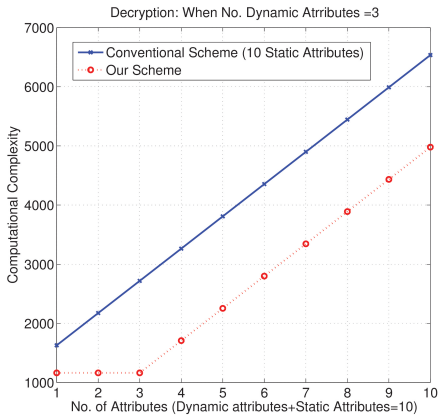

Fig. 5. Complexity comparison for decryption for three dynamic attributes.

for two different testbeds: testbed 1 uses Intel(R) Core(TM) 2 Quad CPU Q6600 with 2.40GHz and 3 GB memory running on Ubuntu 10.04 and testbed 2 uses HTC Desire HD A9191 smart phone running on Android 2.2. Let us assume that the data owner uses an environment similar to the testbed 1 for the encryption while user uses a mobile device similar to the testbed 2 for the decryption.

TABLE I
TIME COMPLEXITY MEASURES FOR TWO DIFFERENT TESTBEDS.

	Testbed 1 (ms)	Testbed 2 (ms)
C_p	14.6	491.2
C_{ex}	2.8	34.1
C_m	1.8	20

Let us denote the number of static attributes used for encryption as n per authority and the total number of dynamic attributes used by data owner as n_d. Table II shows the total time required for encryption (by the data owner) and for decryption (by the user) for the proposed and the conventional ABE schemes. In order to graphically visualize the actual difference between proposed and conventional algorithms, we plotted the computational complexities given in Table II by varying the number of attributes, $n + n_d$, in Figures 3-6.

The computational complexity is measured in terms of total time required for the data owner and the user to encrypt and decrypt the data, respectively. Figures 3-6 depict four different cases (a) encryption with three dynamic attributes and static attributes vary from 1 to 7, (b) encryption with five dynamic attributes and static attributes vary from 1 to 5, (c) decryption with three dynamic attributes and static attributes vary from 1 to 7, and (d) decryption with five dynamic attributes and static attributes vary from 1 to 5. With three and five dynamic attributes (see Fig. 3 and Fig. 4), the proposed algorithm performs 1ms and 3ms faster than the conventional scheme, respectively for encryption. This clearly shows that incorporating more dynamic attributes speed up the encryption. Similarly for the decryption with three dynamic attributes (Fig. 5), the proposed scheme performs nearly 1500ms faster than conventional scheme and for five dynamic attributes (Fig. 6), the proposed scheme performs

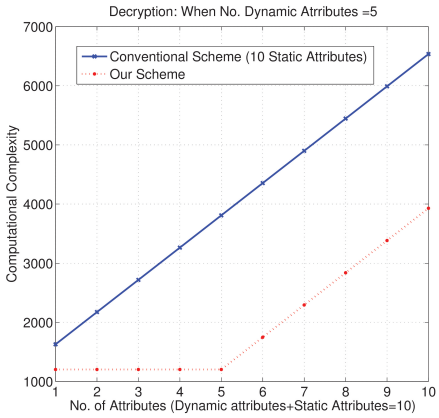

Fig. 6. Complexity comparison for decryption for five dynamic attributes.

nearly 2500ms faster than conventional schemes. This substantial reduction in decryption time enable seamless access to the data for mobile environment.

Remarks: One of the drawbacks of the existing ABE schemes is that the complexity increases linearly with the number of static attributes. Since our algorithm was built on top of the existing ABE scheme the same follows. If the data owner wants to use AAs to issue credentials for dynamic attributes then complexity will increase linearly since those dynamic attributes become static attributes. However, in our solution, as seen from Figures 3-6, any number of dynamic attributes can be added for negligible cost. Hence the complexity in the case of our scheme can be reduced by reducing the number of static attributes and adding more dynamic attributes used for encryption. For example, instead of including ten static attributes from AAs, it is possible in our scheme that the data owner can include five static attributes from AAs and another five dynamic attributes. This approach reduces the complexity by half. Yet, the proposed scheme adds additional layer of security on top of the conventional schemes. In a nutshell, the proposed schemes do not degrade the performance of conventional ABE while including the dynamic attributes to provide run-time security to the data

TABLE II
COMPARISON OF COMPUTATIONAL COST FOR THE DECENTRALIZED IN FIG. 1 AND THE PROPOSED SCHEME IN FIG. 2

	Decentralized ABE Scheme in Fig. 1	Proposed Scheme in Fig. 2
Encryption	$[(2+n)K+1]C_{ex}+(2K+1)C_m+(2K+1)C_m$	$[(2+n)K+1]C_{ex}+(2K+2+n_d)C_m+(2K+2+n_d)C_m$
Decryption	$[(n+1)K+1]C_p+nKC_e+[3+(2+n)K]C_m$	$[(n+1)K+1]C_p+nKC_e+[4+n_d+(2+n)K]C_m$

owner's data.

VI. SECURITY AND PRIVACY ANALYSIS

In Section III-D, we categorized the possible security and privacy threats to the proposed algorithms. In this section, we address each issue and validate that our algorithm is robust against those security and privacy threats.

Mitigate Identity Threat As shown in Fig. 2, public-keys associated with AAs will be published online and the corresponding private-keys are known only to the AAs. At the same time, according to the modulo arithmetic, it is infeasible to compute private-keys from public-keys. During the encryption and decryption, data owners and users use AAs. Data owner and users can verify the public-keys using well-known techniques such as certificates. Hence, impersonating AA is not possible. User device might be at the possession of an attacker where user static attribute credentials are stored on the device. However, adversary cannot get access in to the network without satisfying the dynamic attributes introduced in this paper. Adversary behavior may not be similar as the legitimate user, hence, mapping function running in the user device increases the risk level which will eventually alert the network to deny the service request.

Mitigate Collusion Attacks Two different types of collusion attacks are possible: 1) AAs can collude with each other and aggregate the user attributes, 2) users can pool their decryption keys to access the data which cannot be accessed by individual users. Since our schemes were built on top of the conventional ABE scheme, the proposed schemes are also collusion resistant against up to $(N-2)$ AAs. Hence, let us discuss the user collusion. During the key issuing sub-algorithms, due to the inherent anonymous key issuing protocol, user u will obtain only $D_{kj} = g_1^{R_{kj}} y_k^{x_j/(s_{kj}+u)}$ where user identity u is incorporated within decryption key by inverse exponentiation operation after adding u with random value s_{kj} (known only to authority). From the properties of modulo arithmetic, it is infeasible to infer $x_j/(s_{kj}+u)$ from $y_k^{x_j/(s_{kj}+u)}$. Moreover, the user identity was randomized by s_{kj}, it is infeasible to modify u with other user's identity.

Mitigate Dynamic Attribute Cheating The mapping function installed in the mobile device can be used to verify whether the current user is the legitimate user of the mobile device. However, since it is installed within user's device, malicious users might modify this in order to feed false information for dynamic attributes. Recent technology development in smart device industry already has some working prototype for this kind of security vulnerability i.e., Samsung's KNOX and Blackberry's BES [3][4]. These software platforms are capable of securely installing apps on the users' mobile devices and check for integrity of the installed apps without user interruption. Hence, modifying an app in order to feed false information can be easily detected by the data owner using either KNOX or BES.

[3]Samsung KNOX, http://tinyurl.com/me93jcv

[4]BlackBerry BES12, http://tinyurl.com/ls3yxh8

VII. CONCLUSIONS

In this paper, we proposed a smart access control technique which incorporates attributes generated by smart devices to secure the conventional access control framework without compromising the mobile users' privacy. In the proposed scheme, the data owner incorporates smart device's dynamic attributes together with predefined static attributes. This approach adds additional layer of security on top of the security available in conventional access control framework. We showed that the efficiencies of the proposed schemes are comparable to that of the conventional schemes while offering better security and flexibility for mobile computing network.

REFERENCES

[1] J. L. Hernandez-Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta, and L. Ladid. Toward a lightweight authentication and authorization framework for smart objects. in Selected Areas in Communications, IEEE Journal on , vol. 33, no. 4, pp. 690–702, Apr. 2015.

[2] L. Atzori, A. Iera, and G. Morabito. The Internet of things: A survey." Computer networks, vol. 54, no. 15, pp. 2787-2805, 2010.

[3] A. Sahai, and B. Waters, Fuzzy Identity-based encryption. Advances in Cryptology EUROCRYPT, vol. 3494, 2005.

[4] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Proc. 13th ACM Conf. Comp. Commun. Security, New York, USA, pp. 89–98, 2006.

[5] F. Li, Y. Rahulamathavan, M. Rajarajan, R. C.-W Phan. Low complexity multi-authority attribute based encryption scheme for mobile cloud computing. In Proc. IEEE 7th Int'l Symp. Service Oriented System Engineering (SOSE), San Francisco, USA, pp. 573–577, Mar. 2013.

[6] F. Li, Y. Rahulamathavan, and M. Rajarajan. LSD-ABAC: Lightweight static and dynamic attributes based access control scheme for secure data access in mobile environment. in Proc. IEEE Local Computer Networks (IEEE LCN 2014), Sept. 2014, Edmonton, Canada.

[7] M. Chase. Multi-authority attribute based encryption, In Lecture Notes of Theory of Cryptography in Computer Science, Berlin Heidelberg, pp. 515–534, 2007.

[8] M. Chase, and S. S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Proc. 16th ACM Conf. Comp. Commun. Security, New York, NY, USA, pp. 121–130, 2009.

[9] A. B. Lewko, and B. Waters. Decentralizing attribute-based encryption, in EUROCRYPT, ser. LNCS, K. G. Paterson, Ed., vol. 6632. Springer, pp. 568–588, 2011.

[10] F. Li, Y. Rahulamathavan, M. Conti, M. Rajarajan. Robust Access Control Framework for Mobile Cloud Computing Network. In (Elsevier) Computer Communications, vol. 68, pp. 61-72, Sep. 2015.

[11] J. Han, W. Susilo, Y. Mu, and J. Yan: Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption. In IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 11, pp. 2150–2162, Nov. 2012.

[12] Y. Rahulamathavan, S. Veluru, J. Han, R. Lu, F. Li, and M. Rajarajan. User Collusion Avoidance Scheme for Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption IEEE Transactions on Computers, to appear, 2016.