Mediated Ciphertext-Policy Attribute-Based Encryption and its Application

Luan Ibraimi[1,2], Milan Petkovic[2], Svetla Nikova[1], Pieter Hartel[1], Willem Jonker[1,2]

[1] Faculty of EEMCS, University of Twente, the Netherlands
[2] Philips Research, the Netherlands

Abstract. In Ciphertext-Policy Attribute-Based Encryption (CP-ABE), a user secret key is associated with a set of attributes, and the ciphertext is associated with an access policy over attributes. The user can decrypt the ciphertext if and only if the attribute set of his secret key satisfies the access policy specified in the ciphertext. Several CP-ABE schemes have been proposed, however, some practical problems, such as attribute revocation, still needs to be addressed. In this paper, we propose a mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE) which extends CP-ABE with instantaneous attribute revocation. Furthermore, we demonstrate how to apply the proposed mCP-ABE scheme to securely manage Personal Health Records (PHRs).

1 Introduction

Modern distributed information systems require flexible access control models which go beyond discretionary, mandatory and role-based access control. Recently proposed models, such as attribute-based access control, define access control policies based on different attributes of the requester, environment, or the data object. On the other hand, the current trend of service-based information systems and storage outsourcing require increased protection of data including access control methods that are cryptographically enforced. The concept of Attribute-Based Encryption(ABE) fulfills the aforementioned requirements. It provides an elegant way of encrypting data such that the encryptor defines the attribute set that the decryptor needs to posses in order to decrypt the ciphertext. Since Sahai and Waters [1] proposed the basic ABE scheme, several more advanced schemes have been developed, such as most notably Ciphertext-Policy ABE schemes (CP-ABE) [2,3]. In these schemes, a ciphertext is associated with an access policy and the user secret key is associated with a set of attributes. A secret key holder can decrypt the ciphertext if the attributes associated with his secret key satisfy the access policy associated with the ciphertext. For example, consider a situation when two organizations, a Hospital and a University, conduct research in the field of neurological disorders. The Hospital wants to allow access to their research results to all staff from the University who have the role Professor and belong to the Department of Neurology (DN). To enforce the policy, the Hospital encrypts the data according to the access policy

$\tau_{Results}$=(University Professor $\wedge$ Member of DN). Only users who have a secret key associated with a set of attributes ω=(University Professor, Member of DN) can satisfy the access policy $\tau_{Results}$ and be able to decrypt the ciphertext.
The state-of-the-art CP-ABE schemes provide limited support for revocation of attributes, a feature, which is becoming increasingly important in modern access control systems. In general, attribute revocation may happen due to the following reasons: 1) an attribute is not valid because it has expired, for instance, the attribute "project manager-January 2009 " is valid until January 2009, or 2) a user is misusing her secret key associated with a set of attributes, for instance, Alice might give a copy of her secret key to Bob who is not a legitimate user. In particular, attribute revocation is an important requirement in the domain of access control to personal health data, which is our application field for attribute-based encryption.
Contribution. In this paper, we propose a new scheme for attribute revocation in CP-ABE called mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE). Previous CP-ABE systems proposed to use a system where attributes are valid within a specific time frame [4]. However, the drawback of this approach is that there is no way to revoke an attribute before the expiration date. In our scheme the secret key is divided into two shares, one share for the mediator and the other for the user. To decrypt the data, the user must contact the mediator to receive a decryption token. The mediator keeps an attribute revocation list (ARL) and refuses to issue the decryption token for revoked attributes. Without the token, the user cannot decrypt the ciphertext, therefore the attribute is implicitly revoked. In our scheme we assume that each user has a unique identifier I_u (in CP-ABE the user is identified only with a set of attributes) and may have many attributes. The identifier is used by the mediator to check if there are revoked attributes related to I_u. Different users having different identifiers, may have the same attribute set. For example, Alice with an identifier I_{Alice}, and Bob with an identifier I_{Bob}, may have the same attribute set $\omega = (att_1, att_2)$. The technique of splitting the attribute components of the secret key into two shares, and the technique of using an identifier I_u for each user, helps us to achieve the following attribute revocations: i) revoking an attribute from a single user without affecting other users, and ii) revoking an attribute from the system where all users are affected.
We also define a security model for the proposed scheme which formalizes the security attacks and provide a security proof under the generic group model. Finally, we demonstrate the applicability of the proposed scheme to securely manage Personal Health Records (PHRs).

1.1 Related Work

Attribute-Based Encryption. Sahai and Waters in their seminal paper [1] introduce the concept of ABE. There are two types of ABE schemes: Key-Policy ABE schemes (KP-ABE) [5] and Ciphertext-Policy ABE schemes (CP-ABE) [2,3]. In KP-ABE, a ciphertext is associated with a set of attributes and a user secret

key is associated with an access policy. A secret key holder can decrypt the ciphertext if the attributes associated with the ciphertext satisfy the access policy associated with the secret key. Related to KP-ABE is the technique of searching on encrypted data [6,7,8,9]. In CP-ABE the idea is reversed. A ciphertext is associated with an access policy and the user secret key is associated with a set of attributes. A secret key holder can decrypt the ciphertext if the attributes associated with the secret key satisfy the access policy associated with the ciphertext.

Mediated Cryptography. Boneh et al.[10,11] introduce a method for fast revocation of public key certificates and security capabilities in a RSA cryptosystem called mediated RSA (mRSA). The method uses an online semi-trusted mediator (SEM) which has a share of each users secret key, while the user has the remaining share of the secret key. To decrypt or sign a message, a user must first contact SEM and receive a message-specific token. Without the token, the user cannot decrypt or sign a message. Instantaneous user revocation is obtained by instructing SEM to stop issuing tokens for future decrypt/sign requests. Thus, in mediated cryptography the Trusted Authority (TA) responsible to generate a user key pair, does not deliver the full decryption key to users, but it delivers only a share of it. This method achieves faster revocation of user's security capabilities compared to previous certification techniques such as Certificate Revocation List (CRL) and Online Certificate Status Protocol (OCSP). Libert and Quisquater [12] show that the architecture for revoking security capabilities can be applied to several existing public key encryption schemes including the Boneh-Franklin scheme, and several signature schemes including the GDH scheme. Nali et al. [13] present a mediated hierarchical identity-based encryption and signature scheme. The hierarchical nature of the schemes and the instant revocation capability offered by the SEM architecture allows to enforce access control cryptographically in hierarchically structured communities of users whose access privileges change dynamically. Nali et al. [14] also show how to extend the Libert and Quisquater mediated identity-based cryptographic scheme to allow the enforcement of role-based access control (RBAC).

Revocation. Credential revocation is a critical issue for access control systems. For ABE systems, Pirretti et al. [4] propose to use user attributes for a limited time period. After a specific time period the attribute would become invalid. However, in such systems an attribute cannot be revoked before the expiration date. This approach also requires the list of keys that correspond to attributes to be updated regularly, which would also require the users secret keys to be updated regularly. Boldyreva et al. [15] proposes a revocable IBE scheme. The proposed idea for the revocation is based on binary tree data structure, proposed previously in the PKI setting [16,17]. Boldyreva approach is an improvement to Boneh and Franklin [18] idea, however, when the number of revoked users increases, then the advantage of the proposed scheme is lost over that proposed by Boneh and Franklin, especially when the number of revoked users r becomes close to to the total number of users n in the system. Even if r is less than n, still the key update complexity is bounded by $O(r \log(\frac{n}{r}))$ while in a realistic

solution, the key update complexity should depend on the number of revoked users. Ostrovsky et al.[19] proposes a Key-Policy Attribute-Based Encryption scheme, where the user secret key may be associated with a non-monotonic access policy. The non-monotonic access policy can be represented by a boolean formula such as AND, OR, NOT, and Out Of (threshold) operations. The main drawback of the scheme is that the size of attributes in the ciphertext is fixed, which restricts the expressivity of the scheme. We note that the concept of revoking an attribute is similar to the concept of revoking an identity. Hence, one can revoke an identity of the user instead of revoking an attribute in an access structure. In this paper we propose a mediated CP-ABE scheme which is not limited to the fixed size of attributes which can be revoked.

Organization. The rest of this paper is organized as follows. Section 2 provides background information. In Section 3 we give a formal definition of the mCP-ABE scheme. Section 4 describes the construction of mCP-ABE scheme. In Section 5 we apply the mCP-ABE scheme and describe a general architecture for secure management of Personal Health Records (PHRs). The last section concludes the paper.

2 Background

In this section, we briefly review the basics of bilinear pairing and the security proof in the generic group model, and give a formal definition of CP-ABE.

2.1 Bilinear Pairing

Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative groups of prime order p, and let g be a generator of $\mathbb{G}_0$. A pairing (or bilinear map) $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ satisfies the following properties [20]:

1. Bilinear: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p^*$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
2. Non-degenerate: $\hat{e}(g, g) \neq 1$.

$\mathbb{G}_0$ is said to be a bilinear group if the group operation in $\mathbb{G}_0$ and the bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ can be computed efficiently. Note that the map is symmetric since $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab} = \hat{e}(g^b, g^a)$.

2.2 Security in the Generic Group Model

We prove the security of the scheme based on the generic group model, introduced by Shoup [21]. A proof in the generic group model is based on the fact that the discrete logarithm and the Diffie-Hellman problem are hard to solve as long as the order of the group is a large prime number. The same applies to a group with bilinear pairing where finding the discrete logarithm is a hard problem. In the generic group model group elements are encoded as unique random strings, in such a way that the adversary cannot test any property other than equality.

We prove the security of the mCP-ABE scheme based on the argument that no adversary that acts generally on the groups can break the security of our scheme. This means that if there is an efficient adversary who can discover vulnerabilities in our scheme, then these vulnerabilities can be used to exploit mathematical properties of groups used in the scheme. In the generic model, the adversary has access to the oracles that compute group operations in $\mathbb{G}_0$, $\mathbb{G}_1$, and to the oracle that performs non-degenerate paring $\hat{e}$, while the adversary can test the equality by itself. While it is preferred to prove the security of the scheme by reducing the problem of breaking the scheme to a well studied mathematical problem, a proof in the generic model gives high confidence in the security of the scheme.

2.3 Formal Definition of CP-ABE

The building block of our construction is a CP-ABE scheme. In CP-ABE a message is encrypted under an access policy τ over the set of possible attributes, and a user secret key sk_ω is associated with an attribute set ω. A secret key sk_ω can decrypt the message encrypted under the access policy τ, if and only if the user attribute set ω satisfies the access policy τ. CP-ABE scheme consists of two entities: a trusted authority (TA) and users. The four algorithms: Setup, Keygen, $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ are defined as follows [2]:

- $\mathsf{Setup}(k)$: run by the TA, this algorithm takes as input a security parameter k and outputs the public key pk and a master key mk.
- $\mathsf{Keygen}(\omega, mk)$: run by the TA, this algorithm takes as input the master key mk and a set of attributes ω. The algorithm outputs a secret key sk_ω associated with ω.
- $\mathsf{Encrypt}(m, \tau, pk)$: run by the encryptor, this algorithm takes as input a public key pk, a message m, and an access policy represented by an access tree τ. The algorithm returns the ciphertext c_τ such that only users who have the secret key shares associated with attributes that satisfy the access tree τ will be able to decrypt the message.
- $\mathsf{Decrypt}(c_\tau, sk_\omega)$: run by the decryptor, this algorithm takes as input a ciphertext c_τ, a secret key sk_ω associated with ω, and it outputs a message m, or an error symbol $\perp$ when the attribute set ω does not satisfy the access tree τ.

3 Mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE)

In this section, first, we give a formal definition of our proposed scheme, and later we give the security model in which our scheme is proven to be secure.

3.1 Formal Definition of mCP-ABE

The mCP-ABE scheme consists of three entities: a trusted authority (TA), a mediator and users. The TA uses the master key to generate a user secret key,

which is then divided into two shares such that the first share of the user secret key is sent to the mediator and the second share of the user secret key is sent to a user. The mediator has to stay online all the time, while the TA can be put off-line once it has generated secret keys for all users. The mCP-ABE scheme consists of five algorithms: Setup, Keygen, Encrypt, m-Decrypt, and Decrypt (the Setup and Encrypt algorithms are same as in CP-ABE scheme):

- Keygen(mk, ω, I_u): run by the TA, this algorithm takes as input the master key mk, the user attribute set ω, and the user identifier I_u. The algorithm outputs two secret key shares associated with ω and I_u : $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$. The first share of the secret key $sk_{\omega I_u,1}$ is delivered to the mediator, and the second share of the secret key $sk_{\omega I_u,2}$ is delivered to the user. The secret key shares are delivered through a secure channel to the mediator and to the user.
- m-Decrypt$(c_\tau, I_i, sk_{\omega I_i,1})$: run by the mediator, this algorithm takes as input a ciphertext c_τ, the identifier I_i and the secret key $sk_{\omega I_i,1}$, and outputs a message $\hat{c}_\tau$, or an error symbol $\perp$ when the non-revoked attributes from the set ω do not satisfy the access tree τ.
- Decrypt$(\hat{c}_\tau, sk_{I_i\omega,2})$: run by the message receiver, this algorithm takes as input a ciphertext $\hat{c}_\tau$, and a secret key $sk_{I_i\omega,2}$, and outputs a message m, or an error symbol $\perp$ when the non-revoked attributes from the set ω does not satisfy the access tree τ.

In practice, there might be multiple entities acting as mediators, and a global entity acting as TA. For example, a healthcare organization may choose Proxy_1 as its mediator and a government organization may choose Proxy_2 as its mediator, where each mediator has the first share of the secret key for registered users in the hospital organization, respectively in the government organization. Vanrenen et al. [22] propose the use of peer-to-peer networking (P2P) which would allow users to require a decryption token from every mediator, such as the mediator either tries to compute a decryption token by itself, or forwards the request to its neighbors.

3.2 Security Model

In our scheme the TA is a fully trusted entity which stores securely the master key. We skip discussions about the key escrow problem, since different existing threshold schemes [23,24] can be applied to solve this problem. A mediator is a semi-trusted entity, namely, it should issue decryption tokens to users, but it is not trusted in the sense that it should not obtain information about the plaintaixt.
We define semantic security of mCP-ABE scheme following the security model of Libert and Quisquater [12]. For an encryption scheme to be semantically secure the adversary must not learn anything about the plaintext when the ciphertext and the public key used to create the ciphertext are given. In the security game, the challenger simulates the game and answers adversary $\mathcal{A}$ queries as follows:

1. Setup. The challenger runs the Setup algorithm to generate (pk, mk) and gives the public key pk to the adversary $\mathcal{A}$.
2. Phase1. $\mathcal{A}$ performs a polynomially bounded number of queries:
   - $\mathsf{Keygen}^1(\omega, I_u)$. $\mathcal{A}$ asks for a secret key for the attribute set ω and identifier I_u, and receives the mediator share of the secret key $sk_{\omega I_u,1}$.
   - $\mathsf{Keygen}^2(\omega, I_u)$. $\mathcal{A}$ asks for a secret key for the attribute set ω and identifier I_u, and receives the user share of the secret key $sk_{\omega I_u,2}$.
3. Challenge. $\mathcal{A}$ sends to the challenger two messages m_0, m_1, and the challenge access policy τ^*, such that none of the full secret keys $sk_{\omega I_u}$ (both $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$) generated from the interaction with Keygen^1 and Keygen^2 oracles satisfies τ^*. The challenger picks a random bit $b \in (0, 1)$ and returns $c_{\tau^*} = \mathsf{Encrypt}(m_b, \tau^*, pk)$.
4. Phase2. $\mathcal{A}$ can continue querying with the restriction that none of the full secret keys $sk_{\omega I_u}$ generated from the interaction with Keygen^1 and Keygen^2 oracles satisfies τ^*.
5. Guess. $\mathcal{A}$ outputs a guess $b' \in (0, 1)$.

Definition 1. *The mCP-ABE scheme is said to be semantically secure if any polynomial-time adversary has only a negligible advantage in the security game, where the advantage is defined to be* $|\Pr[b' = b] - \frac{1}{2}|$.

Note that the security game formally captures the following security requirements:

- Resistance against secret key collusion, where different users cannot combine their attribute sets to extend their decryption power. For example, suppose there is a message encrypted under the access tree $\tau = (a_1 \wedge a_2 \wedge a_3)$. Suppose Alice has a secret key $sk_{\omega_A I_A}$ associated with an attribute set $\omega_A = (a_1, a_2)$, and Bob has a secret key $sk_{\omega_B I_B}$ associated with an attribute set $\omega_B = (a_3, a_4)$. Neither Alice's secret key, nor Bob'secret key satisfies the access tree τ. But, if Alice and Bob combine their attribute sets $\omega_A \cup \omega_B = (a_1, a_2, a_3, a_4)$, then the combined attribute sets satisfies the access tree τ. Therefore in the security game we allow the adversary to make secret key queries associated with different attribute sets, say ω_1 and ω_2, such that neither ω_1, nor ω_2 alone can satisfy the challenge access policy τ^*, but $\omega_1 \cup \omega_2$ can satisfy τ^*.
- Resistance against malicious cooperation between the mediator and some users to decrypt the ciphertext associated with an access policy, when the users secret key does not satisfy the access policy. For example, even if a user with attribute set $\omega = (a_1, a_2)$ collude with the mediator, the user should not be capable to decrypt a ciphertext encrypted under a challenge access policy $\tau^* = (a_1 \wedge a_2 \wedge a_3)$, since ω does not satisfy τ^*. Therefore in the security game the adversary is allowed to ask the mediator share (first share of the secret key $sk_{\omega I_u,1}$) and the user share of a secret key (second share of the secret key $sk_{\omega I_u,2}$) for any set of attributes which does not satisfy the challenge access policy τ^*.

4 mCP-ABE scheme

In this section, we give a description of the access policy associated with the ciphertext, and then we give the construction of the scheme. We analyze the security of the scheme and describe how to revoke user attributes. Finally, we show how to extend the proposed scheme to a multi-authority setting.

4.1 Access Policy

In mCP-ABE scheme, an access policy is represented by an access tree τ, in which inner nodes are either $\wedge$ (and) or $\vee$ (or) boolean operators, and leaf nodes are attributes. The access tree τ specifies which combination of attributes the decryptor needs to posses in order to decrypt the ciphertext. Figure 1 presents an example of an access tree τ representing an access policy: $(a_1 \wedge a_4) \vee (a_3 \vee a_5)$.

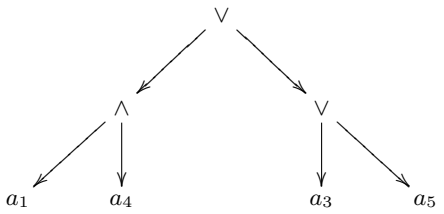

Fig.1. Access tree $\tau = (a_1 \wedge a_4) \vee (a_3 \vee a_5)$

To decrypt an encrypted message under the access tree τ, the decryptor must possess a secret key which is associated with the attribute set which satisfies τ. Attributes are interpreted as logic variables, and possessing a secret key associated with an attribute makes the corresponding logical variable *true*. There are several different sets of attributes that can satisfy the access tree τ presented in Figure 1, such as the attribute set (a_1, a_4), the attribute (a_3), or the attribute (a_5). In our scheme, we assume that attributes are ordered in the access tree e.g index(a_1)=1, index(a_4)=2, index(a_3)=3 and index(a_5)=4.

4.2 Main Construction

1. $\mathsf{Setup}(k)$: On input of the security parameter k, the algorithm generates a group $\mathbb{G}_0$ of prime order p with a generator g and a bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$. The algorithm generates the system attribute set $\Omega = (a_1, a_2, \ldots a_n)$, for some integer n, and for each $a_j \in \Omega$ chooses a random elements $t_j \in \mathbb{Z}_p$. Let $y = \hat{e}(g,g)^{\alpha}$, where α is chosen at random from $\mathbb{Z}_p$, and $\{T_j = g^{t_j}\}_{j=1}^{n}$. The public key is published as:

$$pk = (g, y, \{T_j\}_{j=1}^{n})$$

The master secret key consists of the following components:

$$mk = (\alpha, \{t_j\}_{j=1}^{n})$$

2. $\mathsf{Keygen}(mk, \omega, I_u)$: To generate a secret key for the user with an attribute set ω and an identifier I_u, the Keygen algorithm performs as follows:
   (a) Compute the base component of the secret key: $d_0 = g^{\alpha - u_{id}}$ where $u_{id} \in_R \mathbb{Z}_p$ (for each user with an identifier I_u a unique random value u_{id} is generated).
   (b) Compute the attribute component of the secret key. For each attribute $a_j \in \omega$, choose $u_j \in_R \mathbb{Z}_p$ and compute $d_{j,1} = g^{\frac{u_j}{t_j}}$ and $d_{j,2} = g^{\frac{u_{id} - u_j}{t_j}}$.

   The secret key of the form: $sk_{\omega I_u,1} = \{d_{j,1}\}_{a_j \in \omega}$ is delivered to the mediator, and the secret key of the form: $sk_{\omega I_u,2} = (d_0, \{d_{j,2}\}_{a_j \in \omega})$ is delivered to the user.
3. $\mathsf{Encrypt}(m, \tau, pk)$: To encrypt a message $m \in \mathbb{G}_1$ the algorithm proceeds as follows:
   – Select a random element $s \in \mathbb{Z}_p$ and compute:

$$c_0 = g^s$$
$$c_1 = m \cdot y^s = m \cdot \hat{e}(g, g)^{\alpha s}$$

   – Set the value of the root node of τ to be s, mark all nodes as un-assigned, and mark the root node assigned. Recursively, for each assigned non-leaf node, suppose its value is s, do the following.
     • If the symbol is $\wedge$ and its child nodes are marked un-assigned, let n be the number of child nodes, set the value of each child node, except the last one, to be $s_i \in_R \mathbb{Z}_p$, and the value of the last node to be $s_n = s - \Sigma_{i=1}^{n-1} s_i \mod p$ (i represents the index of an attribute in the access tree). Mark this node assigned.
     • If the symbol is $\vee$, set the values of its child nodes to be s. Mark this node assigned.
   – For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = T_j^{s_i}$.

   Return the ciphertext $c_\tau = (\tau, c_0, c_1, \{c_{j,i}\}_{a_{j,i} \in \tau})$.
4. $\mathsf{m\text{-}Decrypt}(c_\tau, sk_{\omega I_i,1}, I_i)$: when receiving the ciphertext c_τ, the recipient I_i firstly chooses the smallest set $\omega' \subseteq \omega$ that satisfies τ and forwards to the mediator (c_τ, ω', I_i). The mediator checks the Attribute Revocation List (ARL) if any $a_j \in \omega'$ is revoked either from system attribute set Ω or from the user attribute set ω.
   (a) If an attribute is revoked, the mediator returns an error symbol $\perp$ and does not perform further computations.
   (b) If no attribute is revoked, the mediator computes $\hat{c}_\tau$ as follows:

$$\hat{c}_\tau = \prod_{a_j \in \omega'} \hat{e}(T_j^{s_i}, g^{\frac{u_j}{t_j}})$$
$$= \hat{e}(g, g)^{\sum_{a_j \in \omega'} u_j s_i}$$

   Sends $\hat{c}_\tau$ to the recipient.
5. $\mathsf{Decrypt}(\hat{c}_\tau, sk_{\omega I_i,2})$: To decrypt the ciphertext the recipient proceeds as follows:

(a) compute:

$$\begin{aligned} c''_\tau &= \prod_{a_j \in \omega'} \hat{e}(T_j^{s_i}, g^{\frac{u_{id}-u_j}{t_j}}) \\ &= \prod_{a_j \in \omega'} \hat{e}(g^{t_j s_i}, g^{\frac{u_{id}-u_j}{t_j}}) \\ &= \hat{e}(g,g)^{\sum_{a_j \in \omega'}(u_{id}-u_j)s_i} \end{aligned}$$

(b) compute:

$$\begin{aligned} \hat{e}(c_0, d_0) \cdot \hat{c}_\tau \cdot c''_\tau &= \hat{e}(g^s, g^{\alpha - u_{id}}) \cdot \hat{e}(g,g)^{\sum_{a_j \in \omega'} u_j s_i} \cdot \hat{e}(g,g)^{\sum_{a_j \in \omega'}(u_{id}-u_j)s_i} \\ &= \hat{e}(g^s, g^{\alpha - u_{id}}) \cdot \hat{e}(g,g)^{u_{id} s} \\ &= \hat{e}(g^s, g^\alpha) \end{aligned}$$

(c) return m, where

$$\begin{aligned} m &= \frac{c_1}{\hat{e}(g^s, g^\alpha)} \\ &= \frac{m \cdot \hat{e}(g,g)^{\alpha s}}{\hat{e}(g^s, g^\alpha)} \end{aligned}$$

Efficiency. Our scheme is similar to the work of Cheung and Newport [3] on ciphertext-policy attribute-based encryption, however we make major changes in the Key Generation phase, Encryption phase and Decryption phase in order to improve the expressivity of the scheme (the scheme in [3] supports only access policies with logical conjunction), and we improve the efficiency of the scheme (in [3] the size of the ciphertext and secret key increases linearly with the total number of attributes in the system). In our proposed scheme, the size of the shares of the secret key $sk_{\omega I_u,1}$ and $sk_{\omega I_u,2}$ depend on the number of attributes the user has and consists of $|\omega| + 1$ group elements in $\mathbb{G}_0$ ($|\omega|$ is the cardinality of a set ω). The size of the ciphertext c_τ depends on the size of the access policy τ and has $|\tau| + 1$ group elements in $\mathbb{G}_0$, and one group element in $\mathbb{G}_1$. In the m-Decrypt phase, the mediator has to compute ω' pairing operations, where $\omega' \subseteq \omega$ is the attribute set which satisfies the access policy τ. In the decryption phase, to reveal the message, the user has to compute $\omega' + 1$ pairing operations. For the sake of simplicity, we mentioned only access policies which consist of $\wedge$ (and) and $\vee$ (or) nodes. Note that our scheme, in addition to $\wedge$ (and) and $\vee$ (or) nodes, can support threshold nodes or Out Of nodes. For example, the encryptor may specify the access policy 2 Out Of (a_1, a_2, a_3), which implies that the user must have at least two attributes in order to satisfy the access policy and be able to decrypt. If the access policy contains threshold nodes, then the attribute shares s_i can be generated using threshold secret sharing techniques e.g. using Shamir's secret sharing technique. This would require, to use Lagrange basis polynomials in the decryption phase in order to reconstruct the value s.

4.3 Security Analysis

We give a brief discussion about the security of the proposed scheme. A full formal security proof using the generic group model is given in Appendix A. To decrypt a ciphertext without satisfying the access policy, the adversary has to construct $\hat{e}(g^s, g^\alpha)$, and then divide c_1 with $\hat{e}(g^s, g^\alpha)$ to obtain m. To obtain $\hat{e}(g^s, g^\alpha)$, the adversary must first obtain $\hat{e}(g, g)^{su_{id}}$, which can be calculated by pairing the components of the secret key $g^{\frac{u_{id}-u_j}{t_j}}$ with the components of the ciphertext $g^{t_j s_i}$, and then multiply the result with the decryption token $\hat{e}(g, g)^{\sum_{a_j \in \omega'} u_j s_i}$ received from the mediator. However, $\hat{e}(g, g)^{su_{id}}$ can be computed only if the adversary has enough attributes which satisfy the access policy, otherwise this would not be possible. Also note that, if a user acting as an adversary is revoked, then the user will not get the decryption token $\hat{e}(g, g)^{\sum_{a_j \in \omega'} u_j s_i}$ from the mediator, and as a result of this the user cannot reconstruct $\hat{e}(g, g)^{su_{id}}$ even if the user has a secret key with attributes which satisfy the access policy. If we assume that the adversary is able to compromise the mediator, then the adversary will be able to learn the mediator share of user secret key $sk_{\omega I_u,1}$, and be able to compute the decryption token $\hat{e}(g, g)^{\sum_{a_j \in \omega'} u_j s_i}$. However, the decryption token will not help the adversary to decrypt ciphertext which are satisfied by a set of attributes ω. The reason is because the adversary does not know the second share of the user secret key $sk_{\omega I_u,2}$.
The very important security of mCP-ABE scheme is a collusion resistance of user secret keys - it should not be possible for different users to combine their secret keys in order to extend their decryption power. Therefore, to prevent collusion, the Keygen algorithm of our scheme generates a random value u_{id} for each user, which is embedded in each component of the user secret key. Users cannot combine components of the secret key since different users have different random value in their secret keys.

4.4 Attribute Revocation

As already mentioned in section 1, there can be many reasons why an attribute can be revoked. We assume that the mediator maintains an Attribute Revocation List (ARL) which simply has information about attributes revoked from the system attribute set Ω, and attributes revoked from user attribute set ω.
The basic idea is that, when an attribute a_j is revoked from the system attribute set Ω, the TA removes a_j from the system attribute set, and notifies the mediator to stop performing decryption tasks for all users whose attribute secret key involves a_j. When an attribute is revoked from a specific user I_u, the trusted authority notifies the mediator to stop helping the user I_u to perform decryption tasks for the attribute a_j. Therefore, the attribute revocation is achieved immediately after the revocation decision is made.
We assume that there is a policy of revocation authorization maintained by the mediator that describes who is responsible to revoke system or user attributes. At least, the TA should be able to revoke the system and user attributes, and

the owner of the attribute should be able to revoke its attribute because the owner may be the first to notice the compromise of her secret key.

4.5 Multi-Authority mCP-ABE

Ideally, we would like to have multiple independent authorities which would manage user attributes and distribute secret keys. Assume that the Attribute Authority (AA) from Hospital A manages the attribute set $\Omega_{HospitalA}$, and that the AA from Hospital B manages the attribute set $\Omega_{HospitalB}$. In the multi-authority setting, the encryptor has the flexibility to chose different attributes from different authorities in the access policy of the ciphertext, such that only users who have attributes from the given authority can decrypt the ciphertext. For instance, a patient may want to encrypt her health data, such that a user who has the attribute General Practitioner received from Hospital A or the attributes General Practitioner and Pediatrician received from Hospital B can decrypt the ciphertext.
Chase [25] gives the construction of the first multi-authority attribute-based encryption (ABE), which allows multiple independent authorities to monitor user attributes. We can apply the same idea to extend the scheme presented in section 4.2 to support multi-authority mCP-ABE. The main requirement that we have is that each AA should use the same function which takes as input I_u and outputs u_{id}, where u_{id} is used to connect the base component of the secret key with the attribute component of the secret key. The component of the master secret key α is part of the base component of the secret key but is not included in the attribute component of the secret keys, therefore, there is no need for attribute authorities to know α. However, there should be an entity who will manage with α. Thus, in addition to AA, a central authority (CA) is needed. We extend the single-authority mCP-ABE scheme presented in section 4.2 to a multi-authority mCP-ABE as follows (only changes from the scheme in section 4.2 are presented):

1. Setup :
   (a) Central Authority: Generates a group $\mathbb{G}$ of prime order p with a generator g and a bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$. Set the component of the master secret key $\alpha \in_R \mathbb{Z}_p^*$, and the component of the public key $\hat{e}(g,g)^\alpha$.
   (b) Attribute Authority(AA)$-l$: Generate the attribute set $\Omega_l = (a_{l,1}, a_{l,2}...a_{l,n})$. For each $a_{l,j} \in \Omega_l$ set the attribute secret key: $t_{l,1}...t_{l,n}$, and the attribute public key $\{T_{l,j} = g^{t_{l,j}}\}_{j=1}^n$.
2. Keygen
   (a) Central Authority : Compute the base component of the secret key: $d_0 = g^{\alpha - u_{id}}$
   (b) Attribute Authority(AA)$-l$: Suppose a user with an identifier I_u applies for the set of attributes ω to the AA l. The AA l computes the attribute secret key as follows: for each $a_{l,j} \in \omega$, compute $d_{l,j,1} = g^{\frac{u_{l,j}}{t_{l,j}}}$ and $d_{l,j,2} = g^{\frac{u_{id} - u_{l,j}}{t_{l,j}}}$, where $u_{l,j} \in_R \mathbb{Z}_p$.

5 Applying mCP-ABE in Practice

In this section we describe an application of mCP-ABE. We propose to use mCP-ABE to securely manage Personal Health Records (PHRs). This application demonstrate the practicality and usefulness of our scheme.

5.1 Using mCP-ABE to Securely Manage PHRs

Issues around the confidentiality of health records are considered as one of the primary reason for the lack of the deployment of open interoperable health record systems. Health data is sensitive: inappropriate disclosure of a record can change a patient's life, and there may be no way to repair such harm financially or technically. Although, access to health data in the professional medical domain is tightly controlled by existing legislations, such as the U.S. Health Insurance Portability and Accountability Act (HIPAA) [26], private web PHR systems stay outside the scope of this legislation. Therefore, a number of patients might hesitate to upload their sensitive health records to web PHR systems such as the Microsoft Health Vault, Google Health or WebMD. The scheme presented in this paper allows patients to store their sensitive health records on web PHR systems in an encrypted form, while still giving them control to share their data with healthcare providers and/or with their family members. The reason is because the data is encrypted according to an access policy, and the policy moves with the encrypted data. Thus, even if the server which stores health records gets compromised, the confidentiality of the data is preserved since the data is encrypted, and the attacker cannot decrypt the encrypted data without having a secret key. Figure 2 illustrates a general architecture of a PHR system that uses mCP-ABE. The architecture consists of a publishing server, a data repository that includes a security mediator (Proxy), a trusted authority and several data users. The publishing server can be implemented on a home PC of the data source (a patient) or as a trusted service. Its role is to protect and publish health records. The data repository stores encrypted health records, while Proxy is used in the data consumption phase for revocation. The TA is used to set up the keys. Note that the TA and the publishing server do not have to be always online (the TA is needed only in the set-up phase while the publishing server can upload the protected data in an ad-hoc way). There are three basic processes in the management of PHRs:

1. Setup: The steps of this phase are depicted with number 1 in Figure 2. In this phase, the TA distributes the keys to the patients, users and Proxy.
2. Data protection (upload of data to the PHR): When a patient wants to upload protected data to the repository, she contacts the TA to check which attributes are allowed to be used as a policy. Then she creates her access control policy and encrypts the data with the keys corresponding to that policy. Then the data is uploaded to the repository. If she wants to change the policy she can re-encrypt the data and update the repository. All this can be done by a publishing server on behalf of the patient who specifies the access control policy.

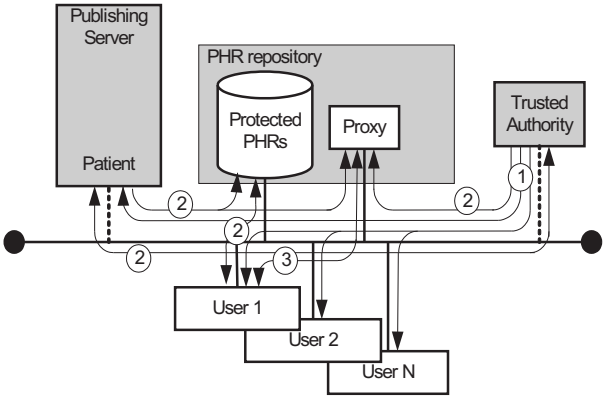

Fig. 2. Secure Management of PHRs

3. Data consumption (doctor's request-response) and revocation: When a user wants to use patient data he contacts the PHR repository and downloads encrypted data. The user makes a request to Proxy for a decryption token. The request contains the encrypted data and a set of user attributes which satisfy the access policy associated with the encrypted data. Proxy checks if any attribute from the user request is not revoked, and, if so, Proxy generates the decryption token and send it to the user. After receiving the decryption token the user decrypts the patient data using the keys corresponding to the appropriate attributes which satisfy the access tree. The steps of this phase are depicted with number 3 in Figure 2.

An addition advantage of an online semi-trusted mediator (Proxy) is that the mCP-ABE scheme can be used to enforce context attributes such as: system date and time or the location from where the request comes from. This is useful for healthcare applications which require context-aware access control where access to patients data depends not only on user roles, but also on the context information. Suppose there is an access policy $\tau = (Location = Hospital \wedge (A \wedge B))$ which says that a doctor (we assume that a doctor is identified with attributes A and B) can view patients health records only if doctor's request comes from inside the hospital. Outside the hospital, no user should be able to decrypt the ciphertext encrypted under the access tree τ, even if the user may own a secret key associated with the attributes (A, B). Using mCP-ABE scheme, the enforcement of τ, which contain context attributes, is done as follows:

– a patient encrypts her health record according to the access policy $(A \wedge B)$, and then uploads his data to a PHR repository. Hence, part of the access policy is enforced in the Encryption phase by the patient.

– a doctor download encrypted data and request from Proxy a decryption token.
– Proxy checks the context attribute inside τ and issues decryption token only if the request comes from inside the hospital (e.g. only if the request comes from a specific IP address), therefore the context attribute is enforced by Proxy in the m-Decrypt phase.

Note that the involvement of an online semi-trusted mediator (Proxy) plays a crucial role in the enforcement of context attributes, as it is very hard or rather impossible to enforce these attributes without the involvement of an online component.
The mCP-ABE scheme can also support the off-line use of data. Then the architecture is slightly changed in a way that Proxy is distributed to the users or their domains within which the data will be used. As a consequence there will be a number of proxies which will be coordinated by the central Proxy. The above defined process will not fundamentally change, except that the central Proxy will update the local ones and that in the data consumption phase, the user will contact only the local Proxy.

6 Conclusion

We propose a mediated Ciphertext-Policy Attribute-Based Encryption (mCP-ABE) scheme that supports revocation of user attributes. If an attribute is revoked, the user cannot use it in the decryption phase. The scheme allows the encryptor to encrypt a message according to an access policy over a set of attributes, and only users who satisfy the access policy and whose attributes are not revoked can decrypt the ciphertext. Furthermore, we demonstrate how to use the proposed scheme to solve very important problems in managing Personal Health Records (PHRs). A possible extension to this work would be to provide a scheme which would have a security proof under standard complexity assumptions.

References

1. A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Advances in Cryptology–Eurocrypt 2005*, volume 3494, pages 457–473. Springer, 2005.
2. J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-Policy Attribute-Based Encryption. *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, pages 321–334, 2007.
3. L. Cheung and C. Newport. Provably secure ciphertext policy ABE. *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 456–465, 2007.
4. M. Pirretti, P. Traynor, P. McDaniel, and B. Waters. Secure attribute-based systems. *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 99–112, 2006.

5. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 89–98, 2006.
6. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Advances in Cryptology – EUROCRYPT 2008*, volume 4965, pages 146–162. Springer, 2008.
7. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. *Journal of Cryptology*, 21(3):350–391, 2008.
8. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search. In *Advances in Cryptology – EUROCRYPT 2004*, volume 3027, pages 506–522. Springer, 2004.
9. X. Boyen and B. Waters. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In *Advances in Cryptology – CRYPTO 2006*, volume 4117, pages 290–307. Springer, 2006.
10. D. Boneh, X. Ding, G. Tsudik, and C.M. Wong. A method for fast revocation of public key certificates and security capabilities. In *Proceedings of the 10th conference on USENIX Security Symposium-Volume 10 table of contents*, pages 22–22. USENIX Association Berkeley, CA, USA, 2001.
11. D. Boneh, X. Ding, and G. Tsudik. Fine-Grained Control of Security Capabilities. *ACM Transactions on Internet Technology*, 4(1):60–82, 2004.
12. B. Libert and J.J. Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 163–171. ACM New York, NY, USA, 2003.
13. D. Nali, A. Miri, and C. Adams. Efficient Revocation of Dynamic Security Privileges in Hierarchically Structured Communities. In *Proceedings of the 2nd Annual Conference on Privacy, Security and Trust (PST 2004), Fredericton, New Brunswick, Canada*, pages 219–223, 2004.
14. D. Nali, C. Adams, and A. Miri. Using Mediated Identity-Based Cryptography to Support Role-Based Access Control. In *Information Security*, volume 3225, pages 245–256. Springer, 2004.
15. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 417–426. ACM, 2008.
16. W. Aiello, S. Lodha, and R. Ostrovsky. Fast digital identity revocation (extended abstract). In *Advances in Cryptology – CRYPTO 98*, pages 137–152, 1998.
17. M. Naor and K. Nissim. Certificate revocation and certificate update. In *In USENIX Security Symposium*, pages 561–570. IEEE, 1998.
18. D. Bonehl and M. Franklin. Identity-Based Encryption from the Weil Pairing. *Advances in Cryptology-Crypto 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, 2001.
19. A. Ostrovksy, A. Sahai, and B. Waters. Attribute Based Encryption with Non-Monotonic Access Structures. In *ACM conference on Computer and Communications Security*, pages 195–203. ACM, 2007.
20. M.Franklin D.Boneh. Identity-based encryption from the Weil pairing. *LNCS*, 2139:213–??, 2001.
21. V. Shoup. Lower Bounds for Discrete Logarithms and Related Problems. *LNCS*, pages 256–266, 1997.
22. G. Vanrenen and S. Smith. Distributing security-mediated PKI. In *Public Key Infrastructure*, volume 3093, pages 218–231. Springer, 2004.

23. A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
24. Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Advances in Cryptology – CRYPTO' 89 Proceedings*, volume 435, pages 307–315. Springer, 1990.
25. M. Chase. Multi-authority Attribute Based Encryption. In *Theory of Cryptography*, volume 4392, pages 515–534. Springer, 2007.
26. The US Department of Health and Human Services. Summary of the HIPAA Privacy Rule, 2003.

A Security Proof

Theorem 1. *Let γ_0 be a random encoding for group $\mathbb{G}_0$ (generic bilinear group), and a γ_1 be a random encoding for group $\mathbb{G}_1$. A random encoding map elements of additive group $\mathbb{Z}_p$ into a bit strings. The advantage of the adversary in the security game issuing at most q queries to the oracles for group operation in $\mathbb{G}_0$ and $\mathbb{G}_1$, to the oracle for computing pairing operation $\hat{e}$, to the oracle for key generation, and to the oracle for encryption is bounded by $O(\frac{q^2}{p})$.*

Proof. Consider groups $\mathbb{G}_0$ and $\mathbb{G}_1$, and generators g of group $\mathbb{G}_0$, and a generator $\hat{e}(g,g)$ of group $\mathbb{G}_1$. In generic group model, group elements are encoded as unique random strings, in such a way that the adversary can not test any property other than equality. In our proof, we use γ_0 as a random encoding for group $\mathbb{G}_0$ (generic bilinear group), and γ_1 as a random encoding for group $\mathbb{G}_1$. Thus, for example, the group element $g^s \in \mathbb{G}_0$ will be encoded as $\gamma_0(s)$, and the group element $\hat{e}(g,g)^\alpha \in \mathbb{G}_1$ will be encoded as $\gamma_1(\alpha)$. Each random encoding is associated with a rational function $f = \frac{\xi}{\varpi}$ over the variables:

$$\Upsilon = \{\alpha, s, s_i, u_{id}, \{u_j\}_{a_j \in \omega}, \{t_j\}_{j=1}^n\}$$

where each variable is an element picked at random in the scheme.
We now give the simulation of the security game. Following the proof from [2], in the simulation we modify slightly the security game given in section 3.2, and simulate a game in which the c_1 component of the challenge phase is either $\gamma_1(\alpha s)$ or $\gamma_1(\theta)$, where $\theta \in_R \mathbb{Z}_p$, and the adversary has to decide whether $c_1 = \gamma_1(\alpha s)$ or $c_1 = \gamma_1(\theta)$. It can be easily shown that if there is no adversary who has non-negligible advantage in a modified game, then there is no adversary who has non-negligible advantage in the security game given in section 3.2. The simulator maintains a table L_1 to store information about values generated from the interaction of the adversary with the Keygen^1 and Keygen^2 oracles. The security game is simulated as follows:

– Setup. The simulator chooses a group $\mathbb{G}_0$ of prime order p with a generator g and a bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$, a random value $\alpha \in \mathbb{Z}_p$, and for each attribute $a_j \in \Omega$, chooses random values $t_j \in \mathbb{Z}_p$. In addition to that, the simulator chooses two encoding functions γ_0 and γ_1, and two oracles for computing group operations in $\mathbb{G}_0$ and $\mathbb{G}_1$, and one oracle for computing pairing $\hat{e}$. The following encodings are sent to the adversary:

1. $\gamma_0(1)$ representing the generator g.
2. $\gamma_1(1)$ representing the generator $\hat{e}(g,g)$.
3. $\gamma_1(\alpha)$ representing $\hat{e}(g,g)^{\alpha}$.
4. $\{\gamma_0(t_j)\}_{j=1}^{n}$ representing $\{T_j = g^{t_j}\}_{j=1}^{n}$.

– Phase1. $\mathcal{A}$ performs a polynomially bounded number of queries:
-Keygen$^1(\omega, I_u)$. $\mathcal{A}$ makes request for the first share (mediator share) of the secret key for an attribute set ω and an identifier I_u. The simulator checks whether L_1 already contains a record for the attribute set ω and the identifier I_u. If such record exists, the simulator fetches $d_{j,1}$ from the record and sends the encoding of $sk_{\omega I_u,1} = (\{d_{j,1}\}_{a_j \in \omega})$ to the adversary $\mathcal{A}$. If such record does not exist, the simulator chooses a random value $u_{id} \in \mathbb{Z}_p$, and for each $a_j \in \omega$ chooses a random value $u_j \in \mathbb{Z}_p$. The simulator generates the following encodings:
   1. $\gamma_0(\alpha - u_{id})$ representing $d_0 = g^{\alpha - u_{id}}$.
   2. $\{\gamma_0(\frac{u_j}{t_j})\}_{a_j \in \omega}$ representing $\{d_{j,1} = g^{\frac{u_j}{t_j}}\}_{a_j \in \omega}$.
   3. $\{\gamma_0(\frac{u_{id}-u_j}{t_j})\}_{a_j \in \omega}$ representing $\{d_{j,2} = g^{\frac{u_{id}-u_j}{t_j}}\}_{a_j \in \omega}$.

   The simulator sends the encoding of $sk_{\omega I_u,1} = (\{d_{j,1}\}_{a_j \in \omega})$ to $\mathcal{A}$, and inserts a new record with the encoding of $sk_{\omega I_u} = (d_0, \{d_{j,1}, d_{j,2}\}_{a_j \in \omega})$ into L_1.
   - Keygen$^2(\omega, I_u)$. $\mathcal{A}$ makes request for the second share (user share) of the secret key for an attribute set ω and an identifier I_u. The simulator checks whether L_1 already contains a record for the attribute set ω and the identifier I_u. If such entry exists, the simulator sends the encoding of $sk_{I_u\omega,2} = (d_0, \{d_{j,2}\}_{a_j \in \omega})$ to the adversary $\mathcal{A}$. If such entry does not exist, the simulator calculates the encodings of d_0, $d_{j,1,}$ and $d_{j,2}$ as explained under Keygen$^1(I_u, \omega)$ and updates the table L_1 with the new encoding of $sk_{\omega I_u} = (d_0, \{d_{j,1}, d_{j,2}\}_{a_j \in \omega})$.

– Challenge. The adversary submits two messages m_0, $m_1 \in \mathbb{G}_1$ and the challenge access policy τ^*. The adversary is not allowed to ask for a challenge access policy τ^* such that one of the full secret keys issued in Phase1 satisfies τ^*.
   The simulation chooses a random $s \in \mathbb{Z}_p$, and for each $a_{j,i} \in \tau^*$ it constructs a value s_i as explained in section 4.2. The following encodings are sent to the adversary:
   1. $\gamma_0(s)$ representing $c_0 = g^s$.
   2. $\gamma_1(\theta)$ representing $c_1 = \hat{e}(g,g)^{\theta}$.
   3. $\{\gamma_0(t_j s_i)\}_{a_{j,i} \in \tau^*}$ representing $\{c_{j,i} = g^{t_j s_i}\}_{a_{j,i} \in \tau^*}$.

– Phase2. $\mathcal{A}$ can continue querying with the restriction that non of the full secret keys generated from the interaction with Keygen^1 and Keygen^2 oracles satisfies τ^*.

The adversary uses the group elements received from the interaction with the simulator to perform generic group operations and equality tests. The simulator provides the adversary with two oracles to compute group operation in $\mathbb{G}_0$, and $\mathbb{G}_1$ and one oracle to compute pairing operations $\hat{e}$. The adversary can make queries to perform group operations as follows:

– Queries to the oracles for group operation in $\mathbb{G}_0$ and $\mathbb{G}_1$: The adversary asks for multiplying or dividing group elements represented with their random encodings, and associated with a rational function. The oracle returns $\gamma_0(a+b)$ or $\gamma_1(a+b)$ when the adversary asks for multiplying $\gamma_0(a)$ and $\gamma_0(b)$, respectively $\gamma_1(a)$ and $\gamma_1(b)$, and returns $\gamma_0(a-b)$ or $\gamma_1(a-b)$ when the adversary asks for dividing $\gamma_0(a)$ and $\gamma_0(b)$, respectively $\gamma_1(a)$ and $\gamma_1(b)$.
– Queries to the oracle for computing pairing operation $\hat{e}$. The adversary asks for pairing of group elements represented with their random encoding, and associated with a rational function. The oracle returns $\gamma_1(ab)$ when the adversary asks for pairing $\gamma_1(a)$ and $\gamma_1(b)$.

We show that the adversary can distinguish with probability $O(\frac{q^2}{p})$ the simulation of the game where the challenge ciphertext is set $c_1 = \gamma_1(\theta)$, with the simulation of the game where the challenge ciphertext would have been set $c_1 = \gamma_1(\alpha s)$.
Firstly, we show what the adversaries view is when the challenge ciphertext is $\gamma_1(\theta)$. The adversaries view can change when an unexpected collision happen due to the random choice of the formal variables $\Upsilon = \{\alpha, s, s_i, u_{id}, \{u_j\}_{a_j \in \omega}, \{t_j\}_{j=1}^{n}\}$ chosen uniformly from $\mathbb{Z}_p$. A collision happen when two queries corresponding to two different rational functions map to a same string representation. Following the security proof from [2] it can be calculated that for any two distinct queries the probability of such collision happen is at most $O(q^2/p)$, where q is the total number of queries done by the adversary. We ignore this situation, since the probability of such collision is negligible.
Secondly, we show what the adversaries view would have been if the challenge ciphertext had been set $\gamma_1(\theta)$, when $\theta = \alpha s$. The adversary view can change when a collision happen, such that the values of two different encodings coincide. Note that the adversary cannot pair $\gamma_1(\theta)$ with other elements (since γ_1 is the encoding for $\mathbb{G}_1$) and the most the adversary can do is to make oracle queries to perform group operation in $\mathbb{G}_1$. Therefore the adversary can ask to multiply θ for δ times, and obtain $\delta\theta = \delta\alpha s$. Let $v_1 = \delta_1\theta$ and $v_2 = \delta_2\theta$. If we subtract v_2 from v_1 we have the following equation:

$$v_1 - v_2 = (\delta_1 - \delta_2)\theta = \delta'\theta = \delta'\alpha s$$

Therefore we say that the adversary can make a query $\delta'\alpha s$. But, we will show that if the adversary does not have sufficient set of attribute to satisfy the challenge access policy τ^*, the adversary cannot make a polynomial query which would be equal to $\delta'\alpha s$ (thus the collision cannot happen), and thus we prove the theorem through a contradiction.
In table 1 we list possible values that the adversary can get using group elements received from interaction with the simulator in the security game. The adversary can get these values by querying the oracle for computing pairing operation $\hat{e}$.
First we observe that the adversary can get $\alpha s - s u_{id}$ by pairing $\alpha - u_{id}$ and s. Thus, the adversary can make a query to the oracle which perform group operation in $\mathbb{G}_1$ to get $\delta'\alpha s - \delta' s u_{id}$, for some δ'. To get only $\delta'\alpha s$, the adversary

s	α	t_j	$\alpha - u_{id}$	$\frac{u_j s}{t_j}$
$\frac{u_j}{t_j}$	$\frac{u_{id}-u_j}{t_j}$	$t_j s$	u_j	$u_j s_i$
$u_{id} - u_j$	$t_j^2 s_i$	$t_j\alpha - t_j u_{id}$	$\frac{\alpha u_j - u_{id}u_j}{t_j}$	$\frac{su_{id}-su_j}{t_j}$
$\frac{\alpha u_{id}-\alpha u_j-u_{id}^2+u_{id}u_j}{t_j}$	$\alpha s - su_{id}$	$\alpha t_j s_i - u_{id}t_j s_i$	$\frac{u_j u_{id}-u_j^2}{t_j^2}$	$s_i u_{id} - s_i u_j$
$st_j s_i$				

Table 1. Possible pairing operations

has to combine group elements received from the interaction with the simulator and from the generic group oracles in order to cancel $\delta' s u_{id}$. From the table 1 we can see that the adversary can construct a query polynomial of the form:

$$\underbrace{\delta'\alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{C} + \underbrace{\delta' \sum_{a_j \in \omega} (u_{id} - u_j) s_i}_{D}$$

for some δ'.

The term B ($\delta' s u_{id}$) can be cancelled only if the sum of the term C and the term D is equal to $\delta' s u_{id}$. Therefore, the adversary must have all necessary secret key components $\frac{(u_{id}-u_j)}{t_j}$ to pair them with $t_j s_i$, and all secret key component $\frac{u_j}{t_j}$ to pair them with $t_j s_i$ and later obtain $s u_{id}$, or $\delta' s u_{id}$ for some constant δ'. However, this is not possible since in the Phase1 and Phase2 the adversary is not allowed to make queries to Keygen^1 and Keygen^2 oracles such that the full secret key generated sk_{ω,I_u} does satisfy the challenge access policy τ^* (there must be at least one $s_i u_{id}$ which the adversary cannot compute). Thus, the sum of terms C and D cannot be used to construct $\delta' s u_{id}$. Therefore the adversary cannot cancel term B, and as a result of this the adversary cannot construct a query of the form $\delta'\alpha s$.

We conclude the proof by making the following analyzes:

- Firstly, we analyze the case when the adversary has a share of a user secret key which satisfies the access policy and his attribute is in the revoked list (the adversary does not receive a decryption token $\sum_{a_j \in \omega} u_j s_i$ from the mediator). We make the following observation:

  - Since in Phase1 and Phase2 the adversary is allowed to make a user share secret key queries which satisfy the challenge access policy (note that the adversary is not allowed to make a query for a full user secret key which satisfies the challenge access policy), the adversary can make a polynomial query which has the form:

$$\underbrace{\delta'\alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} (u_{id} - u_j) s_i}_{D}$$

$$= \underbrace{\delta'\alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' s u_{id}}_{D_1} - \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{D_2}$$

From this we can see that the adversary can cancel terms B and D_1 . However the adversary cannot cancel term D_2, since the adversary needs to have the decryption token $\sum_{a_j \in \omega} u_j s_i$. As a result, the adversary cannot make a polynomial query which has the form $\delta'\alpha s$.

– Secondly, we analyze the case when the adversary corrupts the mediator and obtains the mediator share of the user secret key (the adversary can compute the decryption token) while the adversary does not have the user share of the secret key. We make the following observation:
  - Since in Phase1 and Phase2 the adversary is allowed to make a mediator share secret key queries which satisfy the challenge access policy (note that the adversary is not allowed to make a query for a mediator share of a secret key query if it has made a request for a user share of a secret key), the adversary can make a polynomial query which has the form:

$$\underbrace{\delta'\alpha s}_{A} - \underbrace{\delta' s u_{id}}_{B} + \underbrace{\delta' \sum_{a_j \in \omega} u_j s_i}_{C}$$

As we can see the adversary cannot cancel terms B and C, since the user share of the secret key $\sum_{a_j \in \omega} (u_{id} - u_j) s_i$ is missing. As a result, we conclude that the adversary cannot make a polynomial query which has the form $\delta'\alpha s$. □