Flexible Attribute-Based Encryption

Seiko Arita[*]

Graduate School of Information Security,
Institute of Information Security, Japan
arita@iisec.ac.jp

Abstract. In this paper, we propose a notion of *flexible attribute-based encryption*. Flexible attribute-based encryption is a variant of ciphertext-policy ABE, which allows one to *loosen* a decryption policy underlying a given ciphertext, if one knows some system-wide trapdoor information, without knowing its underlying plaintext message. We give a concrete construction of the flexible attribute-based encryption that satisfies indistinguishability under the loosening operation, based on the construction of ciphertext-policy ABE given by Bethencourt, Sahai and Waters.

Keywords: Attribute-based encryption, Ciphertext-policy, Loosening operation.

1 Introduction

A notion of attribute-based encryption (ABE) was first proposed by Sahai and Waters [13], in which, a message m is encrypted to a ciphertext c under some predicate f, and a user with credential X can decrypt the ciphertext c if and only if the predicate f is satisfied by the user's credential X: $f(X) = 1$. The concept of ABE was further clarified by Goyal, Pandey, Sahai, and Waters [6]. They proposed two complementary forms of ABE: Key-Policy ABE and Ciphertext-Policy ABE. In this paper, we focus on Ciphertext-Policy ABE, in which attributes are used to describe users' credentials and formulas over these attributes are attached to the ciphertext by the encrypting party.

The first construction of Ciphertext-Policy ABE was given by Bethencourt, Sahai, and Waters [4]. Its security is proved under the generic bilinear group with random oracle model. (We call the model which uses both the generic bilinear group and random oracle the generic bilinear group with random oracle model.) Waters [15] gives a construction of ABE which can be proved under the standard model in a selective manner. Lewko, Okamoto, Sahai, Takashima, and Waters [10] and Okamoto and Takashima [12] give fully secure constructions of ABE in the standard model.

[*] Supported by MEXT-Supported Program for the Strategic Research Foundation at Private Universities, 2011-2013.

On a while, ABE has been applied in building a variety of secure systems [14,5,2]. One of major problems in these applications is that ABE-based systems tend to lack flexibility. A ciphertext once produced under decryption policy f never can be decrypted under a more loosened policy $\mathbf{or}(f, \Delta f)$ (if f is not satisfied) by the definition of security of ABE (of course). However, in reality, the degree of privacy of information is never fixed: yesterday's secret is not necessary secret of today. Even if some information is very restrictive (described as policy f) to be accessed at this moment of time, the same information gradually can be made more and more accessible (described as policy $\mathbf{or}(f, \Delta f)$) as time goes by.

Our contribution. In this paper, we propose a notion of *flexible attribute-based encryption.* Flexible attribute-based encryption is a variant of ciphertext-policy ABE, which allows one to *loosen* a decryption policy underlying a given ciphertext, if one knows some system-wide trapdoor information, without knowing its underlying plaintext message. More precisely, suppose a given ciphertext c was generated by encrypting a plaintext m under a decryption policy f. The flexible attribute-based encryption enables a "loosening operation" that, given Δf and some system-wide trapdoor information γ, converts the ciphertext c into a more nonrestrictive version of ciphertext c' which encrypts the same plaintext m under the loosened policy $\mathbf{or}(f, \Delta f)$, without knowing the message m itself. Users having attributes that satisfy (only) the appended policy Δf now can decrypt the ciphertext c' to know the message m. Here we note that the trapdoor information γ is independent of individual policies or ciphertexts.

As one of applications of such flexible attribute-based encryption, we can consider an integration of cloud storage services. Suppose two storage services A and B are going to integrate into one storage service. Suppose, by policy mapping, that encrypted files C_{f_A} under policy f_A in service A now should be decrypted also by entities satisfying policy f_B in service B. The authenticated operator in service A with trapdoor γ can use the loosening operation against those C_{f_A} to get new encrypted files $C_{\mathbf{or}(f_A, f_B)}$ that can be decrypted also by entities satisfying policy f_B in service B.

We will see that there is a subtlety over security concerning such loosening operations and then we will define two notions of security of flexible attribute-based encryption, *indistinguishability under loosening operation* and *indistinguishability under loosening key.*

We also give a concrete construction of the flexible attribute-based encryption that satisfies the indistinguishability under loosening operation and the indistinguishability under loosening key, based on the construction of ciphertext-policy attribute-based encryption given by [4]. Its security proof is given in the generic bilinear group with random oracle model.

Related works. The concept of our flexible attribute-based encryption is similar to the attribute-based proxy re-encryption [7,9,8].

In the attribute-base proxy re-encryption, one can generate re-encryption key $rk_{f_1 \to f_2}$, and by using the key $rk_{f_1 \to f_2}$, a ciphertext c_{f_1} for policy f_1 can be re-encrypted into a ciphertext c_{f_2} for policy f_2. To generate such re-encryption key $rk_{f_1 \to f_2}$, the secret key sk_{f_1} for policy f_1 is required. On a while, in our flexible ABE, all ciphertexts can be "loosened" using the single (system-wide) trapdoor information γ (which is independent of individual policies).

2 A Notion of Flexible Attribute-Based Encryption

A flexible attribute-based encryption scheme is a tuple of five PPT algorithms Setup, Enc, Ext, Dec and Loosen.

Algorithm Setup generates a public parameter par, a master secret mk and a trapdoor information lk for loosening, given a security parameter 1^k: $(par, mk, lk) \leftarrow \mathsf{Setup}(1^k)$. Algorithm Enc encrypts a given message m to a ciphertext c under a given decryption policy represented as a Boolean formula f: $(f, c) \leftarrow \mathsf{Enc}(par, f, m)$. Algorithm Ext generates a secret key d for a given attribute set as, using the master secret mk: $(as, d) \leftarrow \mathsf{Ext}(par, mk, as)$. Algorithm Dec decrypts a ciphertext (f, c) by using a secret key d for an attribute set as to obtain a resulting plaintext m. The plaintext m may be a special symbol $\perp$ indicating a decryption error if something is wrong: $m/\perp \leftarrow \mathsf{Dec}(par, (f, c), (as, d))$. By using the dedicated trapdoor information lk, algorithm Loosen loosens a decryption policy of a given ciphertext (f, c) so that more entities, that satisfy some added policy Δf, can also decrypt the ciphertext c, resulting a new ciphertext $(\mathrm{or}(f, \Delta f), c')$: $(\mathrm{or}(f, \Delta f), c') \leftarrow \mathsf{Loosen}(par, lk, (f, c), \Delta f)$.

Correctness requirement. Under any valid setup information (par, mk, lk) $(\leftarrow \mathsf{Setup}(1^k))$, if one encrypts any message $m \in \mathrm{Message}(k)$ under any decryption policy $f \in \mathrm{Policy}(k)$ to a ciphertext (f, c), then it must be decrypted to the original plaintext m as $\mathsf{Dec}(par, (f, c), (as, d)) = m$, if the secret key (as, d) $(\leftarrow \mathsf{Ext}(par, mk, as))$ is generated for some attribute set as that satisfies the decryption policy f.

If the ciphertext (f, c) is loosened by a policy Δf to a new ciphertext $(\mathrm{or}(f, \Delta f), c')$ as $(\mathrm{or}(f, \Delta f), c') \leftarrow \mathsf{Loosen}(par, lk, (f, c), \Delta f)$, then the resulting ciphertext c' must be decrypted to the original plaintext m as $\mathsf{Dec}(par, (\mathrm{or}(f, \Delta f), c'), (as', d')) = m$, even if the attribute set as' satisfies the appended policy Δf (or f).

Regarding security under loosening operations. Before defining security in a formal way, here we consider some aspects regarding security of such attribute-based encryption that gives loosening operations to users.

First of all, the loosening operation should be performed by some entity with possession of the trapdoor information lk without knowing the underlying message. This will be captured in the security condition named 'indistinguishability under loosening key'.

Another point is a more subtle one. Suppose an adversary A obtains a ciphertext c^* of a plaintext m under a policy $f = A$ or B or C. It is plausible that A

manages to construct a ciphertext c' of the same plaintext m (without knowing m itself) under a more restricted policy $f' = A$ or B, based on the ciphertext c^*. Then, A can use the loosening operation on c' to get another ciphertext c'' also of the same plaintext m but under a loosened policy $f'' = A$ or B or D and then A could know the underlying plaintext m of the original ciphertext c^* by using a corrupt key d_D of the added attribute D against the ciphertext c''.

That scenario means that a victim's ciphertext c^* can be corrupted even if c^* itself has never been processed under loosening operations. (Off course, if the attribute-based encryption has CCA-security, that type of attack based on malleability can be avoided. However, at the same time we lose the loosening operations, too.)

We will require that loosening operations for c' different from c^* should never affect the security of c^*, in the security condition named 'indistinguishability under loosening operation'.

3 Security of Flexible Attribute-Based Encryption

To define security of a flexible attribute encryption scheme, we describe two games using the framework of code-based games [3]. In the framework, a game $\mathbf{Game_A}$ is executed with an adversary A as follows. First, **Initialize** executes, and its outputs are the inputs to A. Then A executes, its oracle queries being answered by the corresponding procedures of $\mathbf{Game_A}$. When A terminates, its output becomes the input to the **Finalize** procedure. The output of the latter is called the output of the game.

3.1 Indistinguishability under Loosening Operation

Let $\mathsf{FABE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Ext}, \mathsf{Dec}, \mathsf{Loosen})$ be a flexible attribute encryption scheme. Let A be an arbitrary PPT adversary against FABE. Our game $\mathbf{Game}^{ind-lso}_{A,\mathsf{FABE}}(k)$ uses the following **Initialize** and **Finalize** procedures:

```
procedure Initialize:
    $b \xleftarrow{\$} \{0,1\}$
    $(par, mk, lk) \leftarrow \mathsf{Setup}(1^k)$
    return $par$.

procedure Finalize $(b')$:
    return $b' \overset{?}{=} b$.
```

The game uses procedures **Extract**, **LR** and **Loosen** to answer oracle queries from A:

```
procedure Extract $(as)$:
    assert$(f^*(as) = \mathsf{false})$
    $(as, d) \leftarrow \mathsf{Ext}(par, mk, as)$
    return $(as, d)$.

procedure Loosen $((f, c), \Delta f)$:
    assert$((f, c) != (f^*, c^*))$
    $(f', c') \leftarrow \mathsf{Loosen}(par, lk, (f, c), \Delta f)$
    return $(f', c')$.

procedure LR $(f^*, m_0, m_1)$:
    assert$(f^*(as) = \mathsf{false})$ for $as$'s submitted to Extract
    $(f^*, c^*) \leftarrow \mathsf{Enc}(par, f^*, m_b)$
    return $(f^*, c^*)$.
```

In the above, "assert($f^*(as) = \mathsf{false}$)" means that one must check whether the condition $f^*(as) = \mathsf{false}$ holds or not if f^* already defined, and abort if it does not hold, or else continue. Similar for "assert($(f, c) != (f^*, c^*)$)".

Definition 1. *A flexible attribute encryption scheme* FABE *is said to be* indistinguishable under loosening operation (IND-LSO) *if for an arbitrary PPT adversary A its advantage* $\mathbf{Adv}^{ind-lso}_{A,\mathsf{FABE}}(k) := |\Pr[\mathrm{Game}^{ind-lso}_{A,\mathsf{FABE}}(k) = 1] - 1/2|$ *is a negligible function in k.*

3.2 Indistinguishability under Loosening Key

Our game $\mathbf{Game}^{ind-lsk}_{A,\mathsf{FABE}}(k)$ uses the following **Initialize** and **Finalize** procedures:

```
procedure Initialize:
    $b \xleftarrow{\$} \{0,1\}$
    $(par, mk, lk) \leftarrow \mathsf{Setup}(1^k)$
    return $(par, lk)$.

procedure Finalize ($b'$):
    return $b' \overset{?}{=} b$.
```

Note that **Initialize** returns a trapdoor information lk for loosening operation as well as parameter par (and adversaries A will know lk as well as par). The game uses procedure **LR** to answer oracle queries from A:

```
procedure LR ($f^*, m_0, m_1$):
    $(f^*, c^*) \leftarrow \mathsf{Enc}(par, f^*, m_b)$
    return $(f^*, c^*)$.
```

Definition 2. *A flexible attribute encryption scheme* FABE *is said to be* indistinguishable under loosening key (IND-LSK) *if for an arbitrary PPT adversary A its advantage* $\mathbf{Adv}^{ind-lsk}_{A,\mathsf{FABE}}(k) := |\Pr[\mathrm{Game}^{ind-lsk}_{A,\mathsf{FABE}}(k) = 1] - 1/2|$ *is a negligible function in k.*

Note that since A has now loosening key lk, A can trivially decrypt the challenge ciphertext if A had access to **Extract**-oracle.

4 Concrete FABE Scheme

Definition 3. *A function* $F : \{0,1\}^* \to \{0,1\}^{2N}$ *is said to be* N-linear-dependency resistant *if for any* $n \leq N$ *any PPT algorithm A is not able to generate any n distinct strings* $x_1, \ldots, x_n$ *with function values* $F(x_1), \ldots, F(x_n)$ *that are linearly dependent (as vectors over* Z_2*) except with a negligible probability.*

A hash function $F : \{0,1\}^* \to \{0,1\}^{2N}$ is N-linear-dependency resistant in the random oracle model with respect to F.

We construct a concrete flexible attribute encryption scheme based on the attribute encryption scheme of [4]. In the followings, $\sharp\mathrm{Leaf}(f)$ denotes a number of leaf nodes of a given binary formula f. $(\rho, M) \leftarrow \mathrm{LSS}(p, f)$ denotes a transformation to convert a Boolean formula f into a linear secret sharing scheme defined by a share-generating matrix M over prime p (with corresponding secret-restoring coefficients $(\omega_i)_i$) with an assignment function ρ from the rows of matrix M to the universe of attributes. For its details we refer to [11]. Predicate $\mathrm{IsDH}(g, g_1, g_2, g_3)$ means the tuple (g, g_1, g_2, g_3) is a Diffie-Hellman tuple, i.e., $g_3 = g_2^a$ for a satisfying $g_1 = g^a$. For vectors $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$, their inner product is written as $a \cdot b = \sum_{i=1,\ldots,n} a_i b_i$.

```
Setup $(1^k, N(k))$:
  $(g, p, e) \leftarrow \mathsf{GenGrp}(1^k)$
  Select $F\ (= F_1 \cdots F_{2N}) : \{0,1\}^* \to \{0,1\}^{2N}$
  $\alpha, \beta, \gamma_1, \ldots, \gamma_{2N} \xleftarrow{\$} Z_p$
  $h = g^\beta, w = e(g,g)^\alpha, u_1 = g^{\gamma_1}, \ldots, u_{2N} = g^{\gamma_{2N}}$
  Return $par = (g, h, w, F, u_1, \ldots, u_{2N})$, $mk = (\beta, g^\alpha)$ and $lk = \gamma := (\gamma_1, \ldots, \gamma_{2N})$.
  /* $(u_1, \ldots, u_{2N})$ defines a hash function $\mathcal{H}(S) = u_1^{F_1(S)} \cdots u_{2N}^{F_{2N}(S)}$ */
```

```
Enc $(par, f, m)$:
 Assert $n := \sharp\mathrm{Leaf}(f) < N$
 $(\rho, M) \leftarrow \mathrm{LSS}(p, f)$ /* Let dimension of $M$ be $n \times l$ */
 $s, r_2, \ldots, r_l \xleftarrow{\$} Z_p$, $s_i = M_i \cdot (s, r_2, \ldots, r_l)$  $(i \in [1..n])$
 $c_0 = mw^s$, $c_1 = g^s$, $c_2 = h^s$, $c_3 = (g^{s_i})_{i \in [1..n]}$, $c_4 = (\mathcal{H}(\rho(i))^{s_i})_{i \in [1..n]}$, $c_5 = \mathcal{H}(f, c_0, \ldots, c_4)^s$
   Return $(f, c = (c_0, \ldots, c_5))$.
```

```
Ext $(f, mk, as)$:
  $r \xleftarrow{\$} Z_p$, $r_a \xleftarrow{\$} Z_p$ $(a \in as)$
  $d_1 = g^{(\alpha + r)/\beta}$
  $d_2 = (g^r \mathcal{H}(a)^{r_a})_{a \in as}$, $d_3 = (g^{r_a})_{a \in as}$
  Return $d = (as, d_1, d_2, d_3)$.
```

```
Dec $(par, (f, c), (as, d))$:
  $(\rho, M) \leftarrow \mathrm{LSS}(p, f)$
  $I = \rho^{-1}(as)$ and compute the constants $\{\omega_i\}_{i \in I}$
  $\kappa = \prod_{i \in I} \{e(c_{3,i}, d_{2,\rho(i)}) / e(c_{4,i}, d_{3,\rho(i)})\}^{\omega_i}$
  Return $\kappa c_0 / e(d_1, c_2)$.
```

```
Loosen $(par, lk, (f, c), \Delta f)$:
  Loosen the policy $f$ to $f' = \mathrm{or}(f, \Delta f)$
  Assert $n := \sharp\mathrm{Leaf}(f') < N$ and $\mathrm{IsDH}(g, \mathcal{H}(f, c_0, \cdots, c_4), c_1, c_5)$
  $(\rho, M) \leftarrow \mathrm{LSS}(p, f')$ /* Let dimension of $M$ be $n \times l$ */
  Let $g^s = c_1$ and $r_2, \ldots, r_l \xleftarrow{\$} Z_p$ /* We don't know the value of $s$ */
  Compute $g^{s_i} = g^{M_i \cdot (s, r_2, \ldots, r_l)}$ for $i \in [1..n]$ and set $c_3' = (g^{s_i})_{i \in [1..n]}$
    /* The knowledge $g^s$ is enough to compute $g^{s_i}$ */
  $c_4' = (g^{s_i(\gamma \cdot F(\rho(i)))})_{i \in [1..n]}$, $c_5' = (c_1)^{\gamma \cdot F(f', c_0, c_1, c_2, c_3', c_4')}$
  Return $c' = (f', c_0, c_1, c_2, c_3', c_4', c_5')$.
```

We can prove the following theorems regarding security of the FABE scheme (the proofs are in the full version [1]).

Theorem 1. *The FABE scheme with parameter $N = N(k)$ is indistinguishable under loosening operation in the generic bilinear group model, under the assumption that the function F is $(N+1)$-linear-dependency resistant.*

Theorem 2. *The FABE scheme is indistinguishable under loosening key in the generic bilinear group model.*

5 Conclusion

We proposed a notion of flexible attribute-based encryption, that allows one to loosen a decryption policy underlying a given ciphertext. We gave a concrete construction of such flexible attribute-based encryption that is provably secure in the generic bilinear group model.

References

1. Arita, S.: Flexible Attribute-Based Encryption (2012), `http://lab.iisec.ac.jp/~arita/pdf/fabe.pdf`
2. Badenand, R., Benderand, A., Springand, N., Bhattacharjee, B., Starin, D.: Persona: An online social network with user defined privacy. In: ACM SIGCOMM (2009)
3. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006)
4. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-Policy Attribute-Based Encryption. In: SP 2007, pp. 321–334. IEEE Computer Society (2007)
5. Bobba, R., Fatemieh, O., Khan, F., Gunter, A.K.C.A., Khurana, H., Prabhakaran, M.: Attribute-based messaging: Access control and confidentiality. ACM Transactions on Information and System Security (TISSEC) 13(4)
6. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
7. Guo, S., Zeng, Y., Wei, J., Xu, Q.: Attribute-based re-encryption scheme in the standard model. Wuhan University Journal of Natural Sciences 13(5), 621–625 (2008)
8. Ibraimi, L., Asim, M., Petkovic, M.: An Encryption Scheme for a Secure Policy Updating. In: SECRYPT 2010, pp. 399–408 (2010)
9. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute Based Proxy Re-encryption with Delegating Capabilities. In: ASIACCS 2009, pp. 276–286 (2009)
10. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010)
11. Lewko, A., Waters, B.: Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011)

12. Okamoto, T., Takashima, K.: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010)
13. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
14. Traynor, P., Butler, K.R.B., Enck, W., McDaniel, P.: Realizing massive-scale conditional access systems through attribute-based cryptosystems. In: NDSS (2008)
15. Waters, B.: Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011)