A unidirectional conditional proxy re-encryption scheme based on non-monotonic access structure

Bin Wang

Information Engineering College of Yangzhou University

No.196 West HuaYang Road, Yangzhou City, Jiangsu Province, P.R.China

E-mail: jxbin76@yahoo.cn

Abstract: Recently, Fang et al. [6] introduced an interactive(bidirectional) conditional proxy re-encryption(C-PRE) scheme such that a proxy can only convert ciphertexts that satisfy access policy set by a delegator. Their scheme supports monotonic access policy expressed by “OR” and “AND” gates. In addition, their scheme is called interactive since generation of re-encryption keys requires interaction between the delegator and delegatee. In this paper, we study the problem of constructing a unidirectional(non-interactive) C-PRE scheme supporting non-monotonic access policy expressed by “NOT”, “OR” and “AND” gates. A security model for unidirectional C-PRE schemes is also proposed in this paper. To yield a unidirectional C-PRE scheme supporting non-monotonic access policy, we extend the unidirectional PRE scheme presented by Libert et al. [8] by using the ideas from the non-monotonic attributed based encryption (ABE) scheme presented by Ostrovsky et al. [9]. Furthermore, the security of our C-PRE scheme is proved under the modified 3-weak Decision Bilinear Diffie-Hellman Inversion assumption in the standard model.

Keywords: Unidirectional conditional proxy re-encryption, The standard model, Non-monotonic access structure, Chosen ciphertext security, Attributed based encryption

1. Introduction

Encryption is one of the most fundamental cryptographic primitives. The concept of proxy re-encryption (PRE) was introduced by Blaze et al. in 1998 [4]. A proxy in PRE systems can convert a ciphertext encrypted under Alice’s public key (delegator) into a ciphertext of the same message under Bob’s public key (delegatee). Proxy re-encryption has many applications such as email forwarding, distributed file system [2]. A bidirectional PRE scheme allows a proxy to convert ciphertexts encrypted under Alice into ciphertexts under

Bob via a re-encryption key and the same key can also be used to translate from Bob to Alice. On the other hand, if the re-encryption key only allows one-way conversion (e.g., from Alice to Bob), then the corresponding PRE scheme is called unidirectional.

The PRE scheme in [4] is bidirectional and CPA secure under DDH assumption. In 2005, Ateniese et al. [2] presented several CPA secure unidirectional PRE schemes based on bilinear pairing. Then Canetti and Hohenberger [5] presented an appropriate definition of chosen ciphertext security(CCA) for bidirectional PRE schemes and the first CCA secure bidirectional PRE scheme. The work in [5] left an open problem to come up with a CCA secure unidirectional PRE scheme. Libert and Vergnaud [8] presented a definition of chosen ciphertext security (CCA) for unidirectional PRE schemes and the first unidirectional PRE scheme with CCA security in the standard model.

Normal PRE schemes allow a semi-trusted proxy to translate ciphertexts from Alice to Bob unconditionally. It is desirable that a proxy can only convert ciphertexts under certain constraints set by the delegator. Shao et al. [12] designed a PRE scheme with keyword search property, which allows a proxy equipped with trapdoor information to test whether a ciphertext from Alice contains one specified keyword. However, it is pointed out [13] that the trapdoor still allows the proxy to convert ciphertexts from Alice without any restriction. On the other hand, Weng et al. [14, 15] introduced the notion of conditional proxy re-encryption (C-PRE) such that only ciphertexts whose keywords satisfy certain conditions set by Alice can be converted by a proxy. They also left it as an open problem to construct a C-PRE scheme supporting access policy consisting of “OR” and “AND” gates over keywords.

Wang et al. [13] presented a unidirectional PRE scheme supporting conjunctive keywords search and selective delegation such that a proxy can only re-encrypt ciphertexts that contain a set of keywords specified by the delegator. In other words, their construction supports access policy expressed by “AND” gates. By regarding keywords as attributes, Fang et al. [6] presented an interactive(bidirectional) single-hop C-PRE scheme based on access tree used in the attribute based encryption scheme [7], which supports access policy consisting of “OR” and ‘AND” gates. Their scheme is called interactive since generation of re-encryption keys requires interactions between the delegator and delegatee who take their secret keys as private input. Interactive generation of re-encryption keys is an essential feature of bidirectional C-PRE scheme defined in [5]. CCA security of their C-PRE scheme was proved under the random oracle model. They also left it as an open problem to construct a *non-interactive(unidirectional)* C-PRE scheme with security in *the standard model*.

Although Wang et al. [13] defined their CCA security model for unidirectional PRE schemes supporting conjunctive keywords search, their security model is coupled tightly with the notion of conjunctive keywords search. Hence the model in [13] is not suitable for C-PRE schemes supporting generic access structure. In addition, the work in [6] considered security model for interactive(bidirectional) C-PRE schemes and proved security of their construction under the random oracle model.

Sahai and Waters [11] introduced the concept of attribute based encryption (ABE), in which a ciphertext is associated with a set of attributes, and a user's private key will reflect an access policy over attributes that controls which ciphertexts a user is able to decrypt. The original construction of Sahai and Waters was limited to express threshold access structure. Goyal et al. [7] presented ABE schemes based on access tree in which the private key supports any monotonic access structure. To increase the expressibility of ABE schemes, Ostrovsky et al. [9] designed an ABE construction that supports non-monotonic access structure represented by "NOT", "OR" and "AND" gates over attributes.

Motivated by the above discussion, we aim to design a *unidirectional(non-interactive)* C-PRE scheme supporting *non-monotonic access structure* to enhance the expressibility of C-PRE schemes. The rest of paper is organized as follows. At first, we provide security definitions for unidirectional C-PRE schemes in which a ciphertext is associated with a set of keywords, and a re-encryption key will reflect an access policy that controls which ciphertexts a proxy is able to re-encrypt. Subsequently, we extend the unidirectional PRE scheme [8] to yield a unidirectional C-PRE scheme supporting non-monotonic access structures. Finally our construction is proved to be CCA secure under the standard model.

A challenge in our security proof lies in the fact that a corrupted user in our model is allowed to obtain re-encryption keys from the target user so long as the access structure associated with these re-encryption keys are not satisfied by the challenge set of attributes associated with the challenge ciphertext. On the other hand, in order to support negation by using the techniques in [9], we have to design two types of re-encryption keys, which also affects the structure of a user's secret key in our construction.

2. Preliminaries

2. 1 Bilinear pairing

Given a security parameter λ, an efficient algorithm $PG(1^\lambda)$ outputs (e, G, G_T, g, p), where G is a cyclic group of a prime order p generated by g, and $2^{\lambda-1} < p < 2^\lambda$. G_T is a cyclic group of the same order, and let $e: G \times G \to G_T$ be a efficiently computable bilinear function with the following properties:

1. Bilinear: $e(g^a, g^b) = e(g, g)^{ab}$, for all $a, b \in Z_p$.

2. Non-degenerate: $e(g, g) \neq 1_{G_T}$

2.2 Modified 3-wDBDHI assumption

Given (e, G, G_T, g, p) output by $PG(1^\lambda)$, we define two experiments in which an adversary A outputs 0 or 1.

Experiment 0: A is given $(g, g^{\frac{1}{a}}, g^a, g^{a^2}, g^b, e(g, g)^{\frac{b}{a^2}}), a, b \leftarrow_R Z_p^*$.

Experiment 1: A is given $(g, g^{\frac{1}{a}}, g^a, g^{a^2}, g^b, T)$, $a, b \leftarrow_R Z_p^*, T \leftarrow_R G_T$.

The modified 3-**weak Decision Bilinear Diffie-Hellman Inversion** assumption [8] claims for any polynomial time algorithm A, the probability $|\Pr[W_0] - \Pr[W_1]|$ is negligible, where W_i is the event that A outputs 1 in experiment i.

2.3 One-time signature

A digital signature scheme $Sig = (\text{Gen,S,V})$ consists of the following algorithms:

1. **Gen**(λ): Outputs a secret/public key pair (sk, pk).

2. **S** (sk, m): Given a secret key sk and a message m, then outputs a signature σ.

3. **V**(pk, m, σ): Takes as input a public key pk, a message m and a signature σ, then outputs either 1 or 0 to denote "accept" or "reject".

We review the definition of strong existential unforgeability for a signature scheme denoted by $Sig = (\mathrm{Gen,S,V})$ in experiment $Exp^{Sig}_{1^\lambda,SCMA}(A)$.

$Exp^{Sig}_{1^k,SCMA}(A)$

The challenger C runs $(pk, sk) \leftarrow \mathrm{Gen}(1^\lambda)$ and sets $S_\sigma \leftarrow \varnothing$.

$(m^*, \sigma^*) \leftarrow A^{\text{O-Sig}}(pk)$.

The adversary A wins if $(m^*, \sigma^*) \notin S_\sigma$ and $\mathbf{V}(pk, m^*, \sigma^*)$ =1.

Advantage of A in experiment $Exp^{Sig}_{1^\lambda,SCMA}(A)$ is defined to be the probability that A wins in the experiment.

The oracle O-Sig is defined as follows:

$\text{O-Sig}(m)$

Returns $\sigma = S(sk, m)$ and updates $S_\sigma = S_\sigma \bigcup \{(m, \sigma)\}$.

A strongly unforgeable one-time signature scheme Sig requires that for any PPT adversary A who can access the oracle O-Sig only once, its advantage Adv^{OTS} in experiment $Exp^{Sig}_{1^k,SCMA}(A)$ is negligible.

3. Security definitions and model

3.1 Syntax of unidirectional C-PRE schemes

A *unidirectional* single-hop C-PRE scheme consists of the following algorithms. A ciphertext is associated with a set of keywords, and a re-encryption key will reflect an access policy over keywords that controls which ciphertexts a proxy is able to re-encrypt.

$\mathrm{Setup}(\lambda)$: Given the security parameter λ, this algorithm produces a set par of global public parameters.

$\mathrm{Keygen}(par)$: Given par, this algorithm generates a secret/public key pair (sk, pk).

$\mathrm{ReKeygen}(par, sk_i, pk_j, \widetilde{A})$: Given par, the secret key sk_i of user i, the public key pk_j of user $j \neq i$ and an access structure $\widetilde{A}$, this algorithm generates a re-encryption key $R_{i \to j, \widetilde{A}}$. We use an algorithm rather than an interactive protocol to implicitly assume that the process of generating re-encryption keys is non-interactive.

$\mathrm{Enc}_1(par, pk_i, m)$: Given par, a public key pk_i and a message m, this algorithm outputs a first level ciphertext CT_1 that cannot be re-encrypted for another party.

$\mathrm{Enc}_2(par, pk_i, m, \gamma)$: Given par, a public key pk_i, a message m and a set γ of keywords(attributes), this algorithm outputs a second level ciphertext CT_2 that can be re-encrypted into a first level ciphertext.

$\mathrm{ReEnc}(par, CT_2, \gamma, pk_i, R_{i \to j, \widetilde{A}})$: Given par, a re-encryption key $R_{i \to j, \widetilde{A}}$ and a second level ciphertext CT_2 encrypted under pk_i and a set γ of keywords, this algorithm outputs a first level ciphertext CT_1 encrypted under pk_j when γ satisfies access structure $\widetilde{A}$;otherwise a message "invalid" is returned.

$\mathrm{Dec}_1(par, sk_i, CT_1)$: Given par, a secret key sk_i and a first level ciphertext CT_1, this algorithm outputs a message m or a message "invalid".

$\mathrm{Dec}_2(par, sk_i, CT_2)$: Given par, a secret key sk_i and a second level ciphertext CT_2, this algorithm outputs a message m or a message "invalid".

In the following, we will take par as implicit input for simplicity. For any message m, any couple of secret/public key pair (sk_i, pk_i), (sk_j, pk_j), the following conditions of correctness should be satisfied:

(1) $\mathrm{Dec}_1(sk_i, \mathrm{Enc}_1(pk_i, m)) = m$; $\mathrm{Dec}_2(sk_i, \mathrm{Enc}_2(pk_i, m, \gamma)) = m$;

(2) If γ satisfies the access structure $\widetilde{A}$, the following should hold:

$$CT_1 = \mathrm{ReEnc}(\mathrm{Enc}_2(pk_i, m, \gamma), \gamma, pk_i, \mathrm{ReKeygen}(sk_i, pk_j, \widetilde{A})),$$

$$\mathrm{Dec}_1(sk_j, CT_1) = m.$$

3.2 Security of second level ciphertexts

Init: As in [8], the adversary Ad determines the target user i^*, the corrupted users and declares a set γ^* of keywords that he wishes to be challenged upon at this stage.

Setup: The challenger C runs $\mathrm{Setup}(\lambda)$ to produce the global public parameters par and generates key pairs as follows:

$$\mathrm{KeyGen}(\cdot) \to (pk^*, sk^*),\ \mathrm{KeyGen}(\cdot) \to (pk_x, sk_x),\ \mathrm{KeyGen}(\cdot) \to (pk_h, sk_h).$$

(pk^*, sk^*) is the key pair for the honest target user i^*. Key pairs subscripted by h or h' represents honest parties and corrupted key pairs are subscripted by x or x'.

Phase 1: Ad takes pk^*, $\{pk_h\}, \{pk_x, sk_x\}$ as input and issue queries to oracles O_{rekey}, O_{renc} and O_{dec-1}.

Challenge: Ad outputs two equal-length messages (m_0, m_1). The challenger C flips a random bit b and returns $CT_2^* = \mathrm{Enc}_2(pk^*, m_b, \gamma^*)$.

Phase 2: Ad still issues queries to oracles O_{rekey}, O_{renc} and O_{dec-1}.

Guess: Ad outputs a bit b'.

The advantage of the adversary in this game is $\varepsilon = |\Pr[b' = b] - 0.5|$. A C-PRE scheme is CCA secure at level 2 if ε is negligible.

The re-encryption key oracle O_{rekey}

Given a tuple $(pk_i, pk_j, \widetilde{A})$, this oracle proceeds as follows:

(1) If both pk_i and pk_j are honest, returns $R_{i \to j, \widetilde{A}} \leftarrow \mathrm{ReKeygen}(sk_i, pk_j, \widetilde{A})$;

(2) If the honest $pk_i = pk^*$, pk_j is corrupted and γ^* does not satisfy $\widetilde{A}$, returns $R_{i^* \to j, \widetilde{A}} \leftarrow \mathrm{ReKeygen}(sk^*, pk_j, \widetilde{A})$;

The re-encryption oracle O_{renc}

Given a tuple $((CT_2, \gamma, pk_i), pk_j, \widetilde{A})$, where CT_2 is a second level ciphertext encrypted under $(pk_i,\ \gamma)$, and pk_i, pk_j are public keys produced by Keygen, this oracle proceeds as follows:

(1) If $pk_i = pk^*$, pk_j is corrupted and $\mathrm{Dec}_2(sk^*, CT_2) \in \{m_0, m_1\}$, returns a message "invalid" since re-encryption may leak information about the challenge bit b in this case.

(2) If γ satisfies the access structure $\widetilde{A}$, computes $R_{i \to j, \widetilde{A}} \leftarrow \mathrm{ReKeygen}(sk_i, pk_j, \widetilde{A})$ and returns the first level ciphertext $CT_1 \leftarrow \mathrm{ReEnc}((CT_2, pk_i, \gamma), R_{i \to j, \widetilde{A}})$. Otherwise, outputs a message "invalid".

First level decryption oracle O_{dec-1}

Given a tuple (pk_i, CT_1), where CT_1 is a first level ciphertext encrypted under the public key pk_i, this oracle proceeds as follows:

(1) If (pk_i, CT_1) is a **derivative** of the challenge pair (pk^*, CT_2^*), returns a message "invalid".

(2) Otherwise, returns $m \leftarrow \mathrm{Dec}_1(sk_i, CT_1)$.

A **Derivative** (pk_i, CT_1) of the challenge pair (pk^*, CT_2^*) in this game is defined as follows:

If CT_1 is a first level ciphertext and $pk_i = pk^*$, or pk_i belongs to a honest user, (pk_i, CT_1) is a **derivative** of the challenge pair if $\mathrm{Dec}_1(sk_i, CT_1) \in \{m_0, m_1\}$.

3.3 Security of first level ciphertexts

Init: The adversary Ad determines the target user i^* and the corrupted users at this stage.

Setup: The challenger C runs $\mathrm{Setup}(\lambda)$ to produce the global public parameters par and generates key pairs in the same way as described previously:

$$\mathrm{KeyGen}(\cdot) \to (pk^*, sk^*),\ \mathrm{KeyGen}(\cdot) \to (pk_x, sk_x),\ \mathrm{KeyGen}(\cdot) \to (pk_h, sk_h).$$

Phase 1: The adversary Ad who takes as input pk^*, $\{pk_h\}, \{pk_x, sk_x\}$ can issue queries to oracles O_{rekey}, O_{dec-1}.

Challenge: Ad outputs two equal-length message (m_0, m_1). The challenger C flips a random bit b and returns $CT_1^* = \mathrm{Enc}_1(pk^*, m_b)$.

Phase 2: Ad still issues queries to the oracle O_{dec-1}.

Guess: The adversary outputs a bit b'.

The advantage of the adversary in this game is $\varepsilon = |\Pr[b' = b] - 0.5|$. A C-PRE scheme is CCA secure at level 1 if ε is negligible.

The re-encryption key oracle O_{rekey}

Given a tuple $(pk_i, pk_j, \widetilde{A})$, this oracle returns $R_{i \to j, \widetilde{A}} \leftarrow \mathrm{ReKeygen}(sk_i, pk_j, \widetilde{A})$. This means that the adversary is allowed access to all re-encryption keys without any restriction.

First level decryption oracle O_{dec-1}

Given a tuple (pk_i, CT_1), where CT_1 is a first level ciphertext encrypted under the public key pk_i, this oracle proceeds as follows:

If (pk_i, CT_1) is a **derivative** of the challenge pair (pk^*, CT_1^*), returns a message "invalid". Otherwise, returns $m \leftarrow \mathrm{Dec}_1(sk_i, CT_1)$.

A **Derivative** (pk_i, CT_1) of the challenge pair (pk^*, CT_1^*) in this game is defined as follows:

If CT_1 is a first level ciphertext and $pk_i = pk^*$, (pk_i, CT_1) is a **derivative** of the challenge pair if $\mathrm{Dec}_1(sk_i, CT_1) \in \{m_0, m_1\}$.

Ateniese et al. [2] defined a security notion called *master secret security* for unidirectional PRE schemes. This notion requires that no coalition of dishonest delegatees be able to pool their re-encryption keys in order to expose the secret key of their common delegator. It is discussed in [6, 8] that CCA security at level 1 implies master secret security for single-hop PRE schemes.

4. Our C-PRE scheme

$\mathrm{Setup}(\lambda)$: Given (e, G, G_T, g, p) output by $PG(1^\lambda)$, picks generators $(g_1, u, v) \leftarrow G, g_2 = g^w, w \leftarrow Z_p^*$ and a strongly unforgeable one-time signature scheme $Sig = (Gen, S, V)$. Let parameter d specifies the exact number of keywords that every second level ciphertext has. We associate each keyword with a unique element in Z_p^*.

Then chooses two random polynomials $h(x)$ and $q(x)$ of degree d subject to the constraint $q(0) = w^{-1} \bmod p$. We also define two functions $T(x) = g_1^{x^d} \cdot g_2^{h(x)}$ and $V(x) = g_2^{q(x)}$ that are publicly computable by interpolation. The set par of public parameters is $(g, u, v, g_1, g_2, g_2^{q(0)} = g, \cdots, g_2^{q(d)}, g_2^{h(0)}, \cdots, g_2^{h(d)}, Sig)$.

Keygen : Picks $(x_{i1}, x_{i2}) \leftarrow Z_p^*$ and sets a secret/public key pair for user i as $sk_i = (x_{i1}, x_{i2}), pk_i = (X_{i1} = g^{x_{i1}}, X_{i2} = g^{x_{i2}})$.

$\mathrm{ReKeygen}(sk_i, pk_j, \widetilde{A})$: Given the secret key sk_i of user i, the public key pk_j of user j and a non-monotonic access structure $\widetilde{A}$, user i generates a re-encryption key $R_{i \to j, \widetilde{A}}$ as follows:

When dealing with a non-monotonic access structure $\widetilde{A}$ over a set of (unprimed)keywords $\widetilde{P}$, we proceed similarly as in [9]. For each unprimed keyword $p \in \widetilde{P}$, we define another primed keyword $p^{/}$. Let $P^{/}=\{p^{/} \mid p \in \widetilde{P}\}$. Then define a monotonic access structure A over $P=P^{/} \bigcup \widetilde{P}$ in such a way that $S \in \widetilde{A}$ if and only if $N(S) \in A$, where $N(\cdot)$ is an operator defined as $N(S)=S \bigcup \{p^{/} \in P^{/} \mid p \in P \setminus S\}$. That is, $N(S)$ consists of all the keywords in S plus the primed part of all the keywords that are not in S.

Let A be associated with a linear secret sharing mechanism $\prod$. Then user i applies $\prod$ over the set P to obtain shares $\{\lambda_k\}$ of the secret x_{i2}^{-1}. For each keyword $\widetilde{p_k} \in P$ (the underlying unprimed keyword is p_k), a random $r_k \leftarrow Z_p$ is chosen:

If $\widetilde{p_k} = p_k$ is unprimed, $D_k = (D_k^{(1)} = X_{j1}^{\lambda_k} \cdot T(p_k)^{r_k}, D_k^{(2)} = X_{i2}^{r_k})$.

If $\widetilde{p_k} = p_k^{/}$ is primed, $D_k = (D_k^{(3)} = X_{j1}^{\lambda_k} g^{r_k}, D_k^{(4)} = V(p_k)^{r_k}, D_k^{(5)} = X_{i2}^{r_k})$.

The re-encryption key $R_{i \to j, \widetilde{A}} = \{D_k\}_{\widetilde{p_k} \in P}$.

$\mathrm{Enc}_1(pk_i, m)$: Given a public key pk_i and a message m, this algorithm proceeds as follows:

(1) Chooses $r \leftarrow Z_p$ and generates a fresh one-time signature key pair $(ssk, svk) \leftarrow Gen(\lambda)$;

(2) $C_1 = svk, C_2^{/} = e(g, X_{i1})^r, C_3 = e(g,g)^r \cdot m$, $C_4 = (u^{svk} v)^r$;

(3) Generates a one-time signature $\sigma = S(ssk, m \| C_3 \| C_4)$;

The first level ciphertext $CT_1 = (C_1, C_2^{/}, C_3, C_4, \sigma)$.

$\mathrm{Enc}_2(pk_i, m, \gamma)$: Given the public key pk_i, a message m and a set γ of d keywords, outputs a second level ciphertext CT_2 that can be re-encrypted into a first level ciphertext as follows:

(1) Chooses $r \leftarrow Z_p$ and generates a fresh one-time signature key pair (ssk, svk);

(2) $C_1 = svk, C_2 = X_{i2}^{r}, C_3 = e(g,g)^r \cdot m, C_4 = (u^{svk} v)^r$

$C_5^p = \{T(p)^r\}_{p\in\gamma}, C_6^p = \{V(p)^r\}_{p\in\gamma}$;

(3) Generate a one-time signature $\sigma = S(ssk, m \| C_3 \| C_4)$.

The second level ciphertext $CT_2 = (C_1, C_2, C_3, C_4, C_5^p, C_6^p, \sigma)$.

$\mathrm{ReEnc}((CT_2, \gamma, pk_i), R_{i\to j,\widetilde{A}})$: Given a re-encryption key $R_{i\to j,\widetilde{A}}$ and a second level ciphertext CT_2 encrypted under (pk_i, γ), if γ satisfies access structure $\widetilde{A}$, this algorithm outputs a first level ciphertext CT_1 encrypted under pk_j as follows:

(1) Parses CT_2 as $C_1 = svk, C_2 = X_{i2}{}^r, C_3 = e(g,g)^r \cdot m, C_4 = (u^{svk}v)^r$

$$C_5^p = \{T(p)^r\}_{p\in\gamma}, C_6^p = \{V(p)^r\}_{p\in\gamma},\ \sigma$$

(2) Recall that $\widetilde{A}$ induces a monotonic access structure A. Denote $\gamma^{/} = N(\gamma)$. As γ satisfies access structure $\widetilde{A}$, $\gamma^{/}$ is authorized in A by previous definition of the operator $N(\cdot)$. Let $I = \{k : \widetilde{p_k} \in \gamma^{/}\}$. A set of coefficients $\{\omega_k\}_{k\in I}$ can be efficiently computed such that $\sum_{k\in I} \omega_k \lambda_k = x_{i2}{}^{-1}$ [3].

For every unprimed attribute $\widetilde{p_k} = p_k \in \gamma^{/}$, (so $p_k \in \gamma$ by definition of the operator $N(\cdot)$), we proceeds as follows:

(2.1) Extracts $D_k = (D_k^{(1)} = X_{j1}{}^{\lambda_k} \cdot T(p_k)^{r_k}, D_k^{(2)} = X_{i2}{}^{r_k})$ from the re-encryption key;

(2.2) Computes $Z_k = e(D_k^{(1)}, C_2) / e(D_k^{(2)}, C_5^{p_k})$

$$= e(g^{x_{j1}\lambda_k} \cdot T(p_k)^{r_k}, g^{x_{i2}r}) / e(g^{x_{i2}r_k}, T(p_k)^r) = e(g,g)^{x_{j1}x_{i2}\lambda_k r}$$

For every primed attributed $\widetilde{p_k} = p_k^{/} \in \gamma^{/}$ (so $p_k \notin \gamma$ by definition), let $\gamma_k = \gamma \bigcup \{p_k\}$. Note that $|\gamma_k| = d+1$ and recall that the degree of the polynomial $q(\cdot)$ is d. Using the keywords in γ_k as an interpolation set, we compute lagrangian coefficients $\{\sigma_p\}_{p\in\gamma_k}$ such that $\sum_{p\in\gamma_k} \sigma_p q(p) = q(0) = (\log_g g_2)^{-1}$. Then we proceeds as follows:

(2.3) Extracts D_k = $(D_k^{(3)} = X_{j1}{}^{\lambda_k} g^{r_k}, D_k^{(4)} = V(p_k)^{r_k}, D_k^{(5)} = X_{i2}{}^{r_k})$ from the re-encryption key and computes:

$$Z_k = \frac{e(D_k^{(3)}, C_2)}{e(D_k^{(5)}, \prod_{p\in\gamma}(C_6^p)^{\sigma_p})e(D_k^{(4)}, C_2)^{\sigma_{p_k}}} = \frac{e(g^{\lambda_k x_{j1}} g^{r_k}, g^{x_{i2}r})}{e(g^{x_{i2}r_k}, \prod_{p\in\gamma}(V(p)^r)^{\sigma_p})e(V(p_k)^{r_k}, g^{x_{i2}r})^{\sigma_{p_k}}}$$

$$= \frac{e(g^{x_{j1}\lambda_k}, g^{x_{i2}r})e(g^{r_k}, g^{x_{i2}r})}{e(g^{x_{i2}r_k}, \prod_{p\in\gamma}(g_2^{q(p)\cdot r})^{\sigma_p})e(g_2^{q(p_k)r_k}, g^{x_{i2}r})^{\sigma_{p_k}}}$$

$$= \frac{e(g^{x_{i2}x_{j1}\lambda_k}, g^r)e(g^{r_k}, g^{x_{i2}r})}{e(g, g_2)^{x_{i2}r_k r\sum_{p\in\gamma_k}\sigma_p q(p)}} = e(g,g)^{x_{j1}x_{i2}\lambda_k r}$$

Finally we have $\prod_{k\in I} Z_k^{\omega_k} = e(g,g)^{x_{j1}x_{i2}r\cdot(\sum_{k\in I}\omega_k\lambda_k)} = e(g,g)^{\frac{rx_{j1}x_{i2}}{x_{i2}}} = e(g,g)^{rx_{j1}}$.

The first level ciphertext $CT_1 = (C_1, C_2^{/} = e(g, X_{j1})^r, C_3, C_4, \sigma)$.

$\mathrm{Dec}_1(sk_i, CT_1)$: Given a secret key sk_i and a first level ciphertext CT_1, this algorithm proceeds as follows:

(1) Parses CT_1 as $(C_1, C_2^{/}, C_3, C_4, \sigma)$;

(2) Computes $C_2^{/\frac{1}{x_{i1}}} = e(g, g^{x_{i1}})^{\frac{r}{x_{i1}}} = e(g,g)^r$ and $m = C_3 / e(g,g)^r$;

(3) Tests $V(C_1, m \| C_3 \| C_4) = 1$ (V1)

If relation V1 does not hold, outputs a message “invalid”; otherwise outputs m .

$\mathrm{Dec}_2(sk_i, (CT_2, \gamma))$: Given a secret key sk_i and a second level ciphertext CT_2, this algorithm proceeds as follows:

(1) Parses CT_2 as $(C_1, C_2, C_3, C_4, C_5^p, C_6^p, \sigma)$;

(2) Tests $e(C_2, (u^{C_1}v)) = e(X_{i2}, C_4)$ (V2)

If relation V2 does not hold, outputs a message “invalid”.

(3) Otherwise, computes $m = \dfrac{C_3}{e(g, C_2)^{\frac{1}{x_{i2}}}} = \dfrac{e(g,g)^r \cdot m}{e(g, g^{x_{i2}\cdot r})^{\frac{1}{x_{i2}}}}$;

(4) If relation V1 does not hold, outputs a message “invalid” ; otherwise outputs m .

Remark: Although our construction requires that every second level ciphertext has exactly

d keywords, this restriction can be mitigated by using the method proposed in [9].

Theorem 1: Assume that the one-time signature scheme is strongly unforgeable. Our scheme is CCA secure at level 2 under the modified 3-wDBDHI assumption.

Proof: Let $(g, A_{-1} = g^{\frac{1}{a}}, A_1 = g^a, A_2 = g^{a^2}, B = g^b, T)$ be a modified 3-wDBDHI instance. We build an algorithm B_A deciding whether $T = e(g,g)^{\frac{b}{a^2}}$ from a successful CCA adversary Ad at level 2 with advantage ε.

Init: The adversary Ad determines the target user i^*, the corrupted users and declares a set γ^* of d keywords to be challenged upon.

Setup: B_A picks a one-time signature scheme $Sig = (Gen, S, V)$ such that the maximal probability δ that any public key can be selected should be less than $2^{-\lambda}$ as in [8]. B_A generates a fresh one-time signature key pair (ssk^*, svk^*) and sets $u = A_1^{\alpha_1}$, $v = A_1^{-\alpha_1 \cdot svk^*} \cdot A_2^{\alpha_2}$, $g_1 = (A_1)^{\mu}, g_2 = A_2, \alpha_1, \alpha_2, \mu \leftarrow Z_p^*$.

Having chosen a random degree d polynomial $f(x)$, two random degree d polynomials $u(x)$ and $h(x)$ are defined as follows:

Let $\gamma^* = \{p_1^*, \cdots, p_d^*\}$. B_A sets $u(x) = -x^d$ for all $x \in \gamma^*$ and $u(x) \neq -x^d$ for some(arbitrary) $x \notin \gamma^*$. This ensures that $u(x) = -x^d$ if and only if $x \in \gamma^*$. Let $h(x) = (a^{-1} \cdot \mu \cdot u(x) + f(x))$. Hence $T(x) = g_1^{x^d} \cdot g_2^{h(x)} = g_1^{x^d + u(x)} \cdot g_2^{f(x)}$ can be publicly computed for arbitrary x.

Then B_A picks $\{\theta_1, \cdots, \theta_d\} \leftarrow Z_p^*$ and implicitly defines a random degree d polynomial $q(x)$ such that $q(0) = (a^2)^{-1}$, $q(p_i^*) = \theta_i$, $1 \le i \le d$. We have $g_2^{q(0)} = g$ and $V(x) = g_2^{q(x)}$ can be computed for arbitrary x by interpolation. These parameters are distributed identically to that in the real scheme.

Key generation: Public key of an honest user $i \in HU \setminus \{i^*\}$ is defined as $X_{i1} = g^{ax_{i1}} = A_1^{x_{i1}}$, $X_{i2} = g^{x_{i2}}$ for randomly chosen $x_{i1}, x_{i2} \leftarrow Z_p^*$. The target user's public key is set as $X_{i^*1} = g^{a \cdot x_{i^*1}} = A_1^{x_{i^*1}}$, $X_{i^*2} = g^{a^2 \cdot x_{i^*2}} = A_2^{x_{i^*2}}$ for randomly chosen $x_{i^*1}, x_{i^*2} \leftarrow Z_p^*$. Key pair of a corrupted user j is set as $X_{j1} = g^{x_{j1}}$, $X_{j2} = g^{x_{j2}}$ for randomly chosen $x_{j1}, x_{j2} \leftarrow Z_p^*$.

Given a tuple $(pk_i, pk_j, \widetilde{A})$ chosen by Ad, the re-encryption key oracle O_{rekey} is simulated by B_A as follows:

Let $\prod$ be the linear secret sharing mechanism associated with the monotonic access structure A induced by $\widetilde{A}$ over a set P. Let M be the share-generating matrix for $\prod$ with l rows and n columns. Each row M_k of M is labeled by a keyword named $\widetilde{p_k} \in P$ and p_k is the unprimed keyword underlying $\widetilde{p_k}$. We list the following propositions:

Proposition 1 [3]**:** Assume Q is not an authorized set in the access structure A. $(\underbrace{1, 0, \cdots, 0}_{n})$ is linearly independent of the rows M_Q, where M_Q is the sub-matrix of M containing those rows labeled by keywords in Q.

Proposition 2 [1, 10]**:** A vector π is linearly independent of a matrix N if and only if there exist a vector θ which can be efficiently computed such that $N \cdot \theta = \vec{0}$ while $\pi \cdot \theta = 1$.

Then we consider the following cases:

(1) $i = i^*$, $j \in CU$ and γ^* does not satisfy $\widetilde{A}$:

When $\widetilde{A}$ is not satisfied by γ^*, $\gamma^{*/} = N(\gamma^*)$ is not an authorized set in A. According to **Proposition** 1 and 2, there exists a column vector $\vec{\theta} = (\theta_1, \cdots, \theta_n)^T$ such that $M_{\gamma^{*/}} \cdot \vec{\theta} = \vec{0}$ and $(\underbrace{1, 0, \cdots, 0}_{n}) \cdot \vec{\theta} = \theta_1 = 1$.

Given a row vector $R = (r_1, \cdots, r_n) \leftarrow Z_p$, let $S = R + (s - r_1) \cdot \vec{\theta}$, where

$s = (a^2 \cdot x_{i^*2})^{-1}$. Note that S is uniformly distributed subject to the constraint that the first component is s. Let $M \cdot S^T$ be the vector of l shares for the secret s. Let M_k be the row labeled by $\widetilde{p_k} \in \gamma^{*/}$, we have that $M_k \cdot \vec{\theta} = 0$ by **Proposition** 2. Hence the share $\lambda_k = M_k \cdot S^T = M_k \cdot R^T$, which has no dependency on s.

(1.1) For a unprimed keyword $\widetilde{p_k} = p_k \in \gamma^* \subseteq \gamma^{*/}$, B_A picks a random $r_k \leftarrow Z_p$ and outputs the following:

$$D_k = (D_k^{(1)} = X_{j1}^{\lambda_k} \cdot T(p_k)^{r_k}, D_k^{(2)} = X_{i^*2}^{r_k})$$

The above is computable since the share λ_k is independent of $(a^2 \cdot x_{i^*2})^{-1}$ by the above-mentioned discussion.

(1.2) For a unprimed keyword $\widetilde{p_k} = p_k \notin \gamma^*$, λ_k is linearly dependent on $(a^2 \cdot x_{i^*2})^{-1}$. B_A picks a random $r_k^{/} \leftarrow Z_p$, implicitly defines $r_k = r_k^{/} - \dfrac{(a\mu)^{-1} \cdot x_{j1} \cdot \lambda_k}{(p_k)^d + u(p_k)}$ and outputs $D_k = (D_k^{(1)}, D_k^{(2)})$ as follows:

$$D_k^{(1)} = X_{j1}^{\lambda_k} \cdot T(p_k)^{r_k} = (g^{x_{j1}})^{\lambda_k} \cdot (g_1^{(p_k)^d + u(p_k)} \cdot g_2^{f(p_k)})^{r_k}$$

$$= (g^{x_{j1}})^{\lambda_k} \cdot (g_1^{(p_k)^d + u(p_k)} \cdot g_2^{f(p_k)})^{r_k^{/} - \frac{(a\mu)^{-1} \cdot x_{j1} \cdot \lambda_k}{(p_k)^d + u(p_k)}}$$

$$= (g_1^{(p_k)^d + u(p_k)} \cdot g_2^{f(p_k)})^{r_k^{/}} \ g^{-\frac{f(p_k) \cdot x_{j1} \cdot a \cdot \mu^{-1} \cdot \lambda_k}{(p_k)^d + u(p_k)}}$$

$$= (g_1^{(p_k)^d + u(p_k)} \cdot g_2^{f(p_k)})^{r_k^{/}} (g^{a \cdot \lambda_k})^{-\frac{f(p_k) \cdot x_{j1} \cdot \mu^{-1}}{(p_k)^d + u(p_k)}}$$

$$D_k^{(2)} = X_{i^*2}^{r_k} = (A_2^{x_{i^*2}})^{r_k^{/} - \frac{(a\mu)^{-1} \cdot x_{j1} \cdot \lambda_k}{(p_k)^d + u(p_k)}} = (A_2^{x_{i^*2} r_k^{/}})(g^{a \cdot \lambda_k})^{-\frac{\mu^{-1} \cdot x_{j1} \cdot x_{i^*2}}{(p_k)^d + u(p_k)}}$$

As λ_k is linearly dependent on $(a^2 \cdot x_{i^*2})^{-1}$, $g^{a \cdot \lambda_k}$ can be computed via A_{-1} and A_1.

(1.3) For a primed keyword $\widetilde{p_k} = p_k^{/} \notin \gamma^{*/}$ (the underlying unprimed keyword $p_k \in \gamma^*$),

λ_k is linearly dependent on $(a^2 \cdot x_{i^*2})^{-1}$. B_A picks a random $r_k^/ \leftarrow Z_p$, and implicitly defines $r_k = r_k^/ - \lambda_k \cdot x_{j1}$. Then outputs $D_k = (D_k^{(3)}, D_k^{(4)}, D_k^{(5)})$ as follows:

$$D_k^{(3)} = X_{j1}{}^{\lambda_k} g^{r_k} = g^{r_k^/}$$

$$D_k^{(4)} = V(p_k)^{r_k} = (g_2)^{\theta_{p_k} \cdot (r_k^/ - \lambda_k \cdot x_{j1})} = (A_2)^{r_k^/ \cdot \theta_{p_k}} \cdot (A_2{}^{\lambda_k})^{-x_{j1} \cdot \theta_{p_k}}$$

$$X_{i*2}{}^{r_k} = (g^{a^2 \cdot x_{i^*2}})^{(r_k^/ - \lambda_k \cdot x_{j1})} = (A_2^{r_k^/ \cdot x_{i^*2}}) \cdot (g^{a^2 \cdot \lambda_k})^{-x_{j1} \cdot x_{i^*2}}$$

$g^{a^2 \cdot \lambda_k}$ can be computed via A_2 since λ_k is linearly dependent on $(a^2 \cdot x_{i^*2})^{-1}$.

(1.4) For a primed keyword $\widetilde{p_k} = p_k^/ \in \gamma^{*/}$ (the underlying unprimed keyword $p_k \notin \gamma^*$), λ_k is independent of $(a^2 \cdot x_{i^*2})^{-1}$. B_A picks a random $r_k \leftarrow Z_p$, Then outputs the following:

$$D_k = (D_k^{(3)} = X_{j1}{}^{\lambda_k} g^{r_k}, D_k^{(4)} = V(p_k)^{r_k}, D_k^{(5)} = X_{i*2}{}^{r_k})$$

(2) $i = i^*$ and $j \in HU \setminus \{i^*\}$:

(2.1) For a primed keyword $\widetilde{p_k} = p_k^/$, B_A picks a random $r_k \leftarrow Z_p$ and outputs $D_k = (D_k^{(3)}, D_k^{(4)}, D_k^{(5)})$ as follows:

$$D_k^{(3)} = X_{j1}{}^{\lambda_k} g^{r_k} = g^{a \cdot x_{j1} \lambda_k} g^{r_k} = (g^{a \cdot \lambda_k})^{x_{j1}} g^{r_k}$$

$$D_k^{(4)} = V(p_k)^{r_k}$$

$$D_k^{(5)} = X_{i^*2}{}^{r_k} = A_2^{r_k \cdot x_{i^*2}}$$

As λ_k may be linearly dependent on $(a^2 \cdot x_{i^*2})^{-1}$, $g^{a \cdot \lambda_k}$ can be computed via A_{-1} and A_1.

(2.2) For a unprimed keyword $\widetilde{p_k} = p_k$, B_A picks a random $r_k \leftarrow Z_p$ and outputs $D_k = (D_k^{(1)}, D_k^{(2)})$ as follows:

$$D_k^{(1)} = X_{j1}{}^{\lambda_k} \cdot T(p_k)^{r_k} = (g^{a \cdot \lambda_k})^{x_{j1}} \cdot T(p_k)^{r_k}$$

$$D_k^{(2)} = X_{i^*2}{}^{r_k} = A_2^{r_k \cdot x_{i^*2}}$$

Similarly, $g^{a\cdot\lambda_k}$ can be computed via A_{-1} and A_1.

(3) If $i \in HU \setminus \{i^*\}$: This can be handled easily since we know the shared secret x_{i2} in this case.

Finally, returns the re-encryption key $R_{i\to j,\widetilde{A}} = \{D_k\}$.

Let the challenge second level ciphertext $CT_2^* = (svk^*, C_2^*, C_3^*, C_4^*, C_5^{p*}, C_6^{p*}, \sigma^*)$ and m_b be the encrypted challenge message. Then we define an event F_{OTS} as follows and bound its probability to occur in a way similar to [8].

(1) The adversary issues a re-encryption query in which $CT_2 = (svk^*, C_2, C_3, C_4, C_5^p, C_6^p, \sigma)$ and m is the implicit message embedded in CT_2 such that $(m \| C_3 \| C_4, \sigma) \neq (m_b \| C_3^* \| C_4^*, \sigma^*)$ and $V(svk^*, m \| C_3 \| C_4, \sigma) = 1$.

(2) The adversary issues a first level decryption query in which $CT_1 = (svk^*, C_2^/, C_3, C_4, \sigma)$ and m is the implicit message embedded in CT_1 such that $(m \| C_3 \| C_4, \sigma) \neq (m_b \| C_3^* \| C_4^*, \sigma^*)$ and $V(svk^*, m \| C_3 \| C_4, \sigma) = 1$.

As the adversary has no information on svk^* before the challenge phase, the probability of occurrence of F_{OTS} in phase 1 can be bounded by $q_O \cdot \delta \le \frac{q_O}{2^\lambda}$, where q_O is the total number of oracle queries made by the adversary and δ is the maximal probability that any one-time public key can be selected.

During the guess phase, the event F_{OTS} could be used to construct an algorithm breaking strong unforgeability of the one-time signature scheme. Therefore $\Pr[F_{OTS}] \le \frac{q_O}{2^\lambda} + Adv^{OTS}$.

The re-encryption oracle O_{renc} is simulated as follows:

Given a tuple $((CT_2, \gamma), pk_i, pk_j, \widetilde{A})$ chosen by the adversary, B_A proceeds as follows:

(1) Parses CT_2 as $(C_1, C_2, C_3, C_4, C_5^p, C_6^p, \sigma)$;

(2) If $\widetilde{A}$ is not satisfied by γ or relation V2 does not hold, outputs a message "invalid";

(3) If $i \in HU \setminus \{i^*\}$ or $j \notin CU$, B_A makes a query to the oracle O_{rekey} and re-encrypts by using the returned re-encryption key.

(4) If $i = i^*$, $j \in CU$, we consider the following sub-cases:

(4.1) $C_1 = svk^*$: If $(m, C_3, C_4, \sigma) = (m_b, C_3^*, C_4^*, \sigma^*)$ in this situation, we have $m = m_b$. But we should return "invalid" by our convention for the re-encryption oracle presented in section 3.2. In addition, $C_1 = svk^* \wedge (m, C_3, C_4, \sigma) \neq (m_b, C_3^*, C_4^*, \sigma^*)$ implies an occurrence of F_{OTS}. Hence B_A halts when $C_1 = svk^*$ at step (4.1).

(4.2) $C_1 \neq svk^*$: Assuming $C_4 = (u^{svk} v)^r$, relation V2 guarantees that $C_2 = (X_{i^*2})^r$. So $C_2^{1/x_{i^*2}} = (A_2^{r \cdot x_{i^*2}})^{1/x_{i^*2}} = A_2^r$, $C_4 = (u^{svk} v)^r = (A_1^{\alpha_1 (svk - svk^*)} \cdot A_2^{\alpha_2})^r$, B_A computes $\left(\frac{C_4}{C_2^{\alpha_2/x_{i^*2}}}\right)^{\frac{1}{\alpha_1 (svk - svk^*)}} = A_1^r$, $e(A_1^r, A_{-1}^{x_{j1}}) = e(g, X_{j1})^r = C_2^{/}$. Let $CT_1 = (C_1, C_2^{/}, C_3, C_4, \sigma)$. If $\mathrm{Dec}_1(sk_j, CT_1) \in \{m_0, m_1\}$, returns a message "invalid". Otherwise, returns CT_1.

The first-level decryption oracle O_{dec-1} is simulated as follows:

Given a tuple (pk_i, CT_1), where CT_1 is a first level ciphertext encrypted under a public key pk_i, B_A proceeds as follows:

(1) Parses CT_1 as $(C_1, C_2^{/}, C_3, C_4, \sigma)$;

(2) If $i \in CU$, decrypts the ciphertext by the known secret key;

(3) $i \in HU$, $C_1 = svk^*$: Assuming $(m, C_3, C_4, \sigma) = (m_b, C_3^*, C_4^*, \sigma^*)$, we should output a message "invalid" to indicate that CT_1 is a Derivative of the challenge (pk^*, CT_2^*). Otherwise, $(m, C_3, C_4, \sigma) \neq (m_b, C_3^*, C_4^*, \sigma^*)$ means that we either face with an occurrence of the event F_{OTS} or B_A should output a message "invalid" to indicate that relation V1 does not hold. Hence B_A halts at step (3).

(4) $i \in HU$ and $C_1 \neq svk^*$: As $C_4 = (u^{svk}v)^r = (A_1^{\alpha_1(svk-svk^*)} \cdot A_2^{\alpha_2})^r$, B_A computes:

$$e(A_{-1}, C_4) = e(A_{-1}, A_1^{\alpha_1(svk-svk^*)r})e(A_{-1}, A_2^{\alpha_2 r})$$

$$(C_2^{/})^{\frac{\alpha_2}{x_i}} = e(g, X_{i1})^{r\cdot\frac{\alpha_2}{x_{i1}}} = e(g, g^{a\cdot x_{i1}})^{r\cdot\frac{\alpha_2}{x_{i1}}} = e(g, g^a)^{\alpha_2 r}$$

$$\left(\frac{e(A_{-1}, C_4)}{(C_2^{/})^{\frac{\alpha_2}{x_i}}}\right)^{\frac{1}{\alpha_1(svk-svk^*)}} = e(g,g)^r, \quad m = C_3 / e(g,g)^r$$

(5) If relation V1 does not hold or $m \in \{m_0, m_1\}$, outputs a message "invalid"; Otherwise outputs m.

Challenge: The adversary outputs two equal-length message (m_0, m_1). B_A flips a random bit b and outputs the challenge second level ciphertext CT_2^* as follows:

$$C_1^* = svk^*, C_2^* = B^{x_{i^*2}}, C_3^* = T \cdot m_b, C_4^* = B^{\alpha_2}, \quad C_5^{p*} = \{B^{f(p_i^*)}\}_{p_i^* \in \gamma^*} \quad C_6^{p*} = \{B^{\theta_i^*}\}_{p_i^* \in \gamma^*},$$

$$\sigma^* = S(ssk^*, m_b \,\|\, C_3^* \,\|\, C_4^*)$$

When Ad outputs $b^/ = b$, B_A outputs 1 to indicate that $T = e(g,g)^{\frac{b}{a^2}}$. Otherwise B_A outputs 0 to indicate that T is random.

If $T = e(g,g)^{\frac{b}{a^2}}$, we have $r^* = \frac{b}{a^2}$ and the following equations:

$$C_2^* = X_{i^*2}{}^{r^*} = (g^{a^2 \cdot x_{i^*2}})^{\frac{b}{a^2}} = B^{x_{i^*2}}$$

$$C_4^* = (u^{svk^*}v)^{r^*} = (A_2^{\alpha_2})^{r^*} = B^{\alpha_2}$$

$$C_5^{p*} = \{T(p_i^*)^{r^*}\}_{p_i^* \in \gamma^*} = \{A_2^{f(p_i^*)\cdot r^*}\}_{p_i^* \in \gamma^*} = \{B^{f(p_i^*)}\}_{p_i^* \in \gamma^*}$$

$$C_6^{p*} = \{V(p_i^*)^{r^*}\}_{p_i^* \in \gamma^*} = \{A_2^{q(p_i^*)\cdot r^*}\}_{p_i^* \in \gamma^*} = \{B^{\theta_i^*}\}_{p_i^* \in \gamma^*}$$

So CT_2^* is a valid encryption of m_b if $T = e(g,g)^{\frac{b}{a^2}}$. In contrast, if T is random, CT_2^* perfectly hides m_b and Ad guesses b with probability 0.5. Hence the overall advantage of B_A is $\varepsilon - \Pr[F_{OTS}]$.

Theorem 2: Assume that the one-time signature scheme is strongly unforgeable, our scheme is CCA secure at level 1 under the modified 3-wDBDHI assumption.

Proof: Let $(g, A_{-1} = g^{\frac{1}{a}}, A_1 = g^a, A_2 = g^{a^2}, B = g^b, T)$ be a modified 3-wDBDHI instance. We build an algorithm B_A deciding whether $T = e(g,g)^{\frac{b}{a^2}}$ from a successful CCA adversary Ad at level 1 with advantage ε.

Init: The adversary Ad determines the target user i^* and the corrupted users at this stage.

Setup: B_A picks a one-time signature scheme $Sig = (Gen, S, V)$ and generates a fresh one-time signature key pair (ssk^*, svk^*).

B_A sets $u = g^{\alpha_1}$, $v = g^{-\alpha_1 \cdot svk^*} \cdot X_{i^*1}^{\alpha_2}, \alpha_1, \alpha_2 \leftarrow Z_p^*$, $g_1 \leftarrow G, g_2 = g^w, w \leftarrow Z_p^*$. Then chooses two random polynomials $h(x)$ and $q(x)$ of degree d subject to the constraint $q(0) = w^{-1}$. Subsequently B_A defines two publicly computable functions $T(x) = g_1^{x^d} \cdot g_2^{h(x)}$ and $V(x) = g_2^{q(x)}$.

Key generation: Key pair of an honest users $i \in HU \setminus \{i^*\}$ is defined as $X_{i1} = g^{x_{i1}}$, $X_{i2} = g^{x_{i2}}$ for randomly chosen $x_{i1}, x_{i2} \leftarrow Z_p^*$. The target user's public key is set as $X_{i^*1} = g^{a^2 \cdot x_{i^*1}} = A_2^{x_{i^*1}}$, $X_{i^*2} = g^{a \cdot x_{i^*2}}$ for randomly chosen $x_{i^*1}, x_{i^*2} \leftarrow Z_p^*$. Key pair of a corrupted user j is set as $X_{j1} = g^{x_{j1}}$, $X_{j2} = g^{x_{j2}}$ for randomly chosen $x_{j1}, x_{j2} \leftarrow Z_p^*$.

Given a tuple $(pk_i, pk_j, \widetilde{A})$ chosen by the adversary, the re-encryption key oracle O_{rekey} is simulated by B_A as follows:

Let $\prod$ be the linear secret sharing mechanism associated with the monotonic access structure A induced by $\widetilde{A}$ over a set P. As the secret keys of corrupt users and honest users $i \neq i^*$ are known to B_A, we only consider how to handle the following case:

(1) $i = i^*, j \neq i^*$:

Let M be the share-generating matrix for $\prod$ with l rows and n columns. Each row M_k of M is labeled by a keyword named $\widetilde{p_k} \in P$ and p_k is the unprimed keyword underlying $\widetilde{p_k}$. Given a vector $R = ((a \cdot x_{i^*2})^{-1}, r_2, \cdots, r_n)$, where $(r_2, \cdots, r_n)$ are randomly chosen from Z_p, $M \cdot R^T$ is the vector of l shares for the secret $(a \cdot x_{i^*2})^{-1}$. For each keyword $\widetilde{p_k} \in P$ (the underlying unprimed keyword is p_k), a random $r_k \leftarrow Z_p$ is chosen by B_A.

If $\widetilde{p_k}$ is unprimed, $D_k = (D_k^{(1)} = X_{j1}^{\lambda_k} \cdot T(p_k)^{r_k}, D_k^{(2)} = X_{i2}^{r_k})$.

If $\widetilde{p_k}$ is primed, $D_k = (D_k^{(3)} = X_{j1}^{\lambda_k} g^{r_k}, D_k^{(4)} = V(p_k)^{r_k}, D_k^{(5)} = X_{i2}^{r_k})$

A key point in the above expressions is to compute $X_{j1}^{\lambda_k} = (g^{\lambda_k})^{x_{j1}}$. As $\lambda_k = M_k \cdot R^T$ may be linearly dependent on the secret $(a \cdot x_{i^*2})^{-1}$, g^{λ_k} can be computed via A_{-1}.

The first level decryption oracle O_{dec-1} is simulated as follows:

As the secret keys of corrupt users and honest users $i \neq i^*$ are known to B_A, given a first level ciphertext CT_1 and a public key pk_i, we only consider how to handle the case $i = i^*$:

(1) Parses CT_1 as $(C_1, C_2^/, C_3, C_4, \sigma)$;

(2) $C_1 = svk^*$: Assume $(m \| C_3 \| C_4, \sigma) = (m_b \| C_3^* \| C_4^*, \sigma^*)$, which implies the randomness $r = r^*$. So $C_2^/ = C_2^{/*}$ and we have $CT_1 = CT_1^*$. But the challenge ciphertext CT_1^* is not allowed to be decrypted by our security definition. On the other hand, $(m \| C_3 \| C_4, \sigma) \neq (m_b \| C_3^* \| C_4^*, \sigma^*)$ means that we either face with an occurrence of the event F_{OTS} or B_A should output a message “invalid” to indicate that relation V1 does not hold. Hence B_A halts at step (2).

(3) $C_1 \neq svk^*$: As $C_4 = (u^{svk}v)^r = (g^{\alpha_1(svk-svk^*)} \cdot X_{i^*1}{}^{\alpha_2})^r$, B_A computes:

$$e(g, C_4) = e(g, g^{\alpha_1(svk-svk^*)r})e(g, X_{i^*1}{}^{\alpha_2 \cdot r})$$

$$(C_2^{/})^{\alpha_2} = e(g, X_{i^*1})^{\alpha_2 \cdot r}$$

$$(\frac{e(g, C_4)}{(C_2^{/})^{\alpha_2}})^{\frac{1}{\alpha_1(svk-svk^*)}} = e(g, g)^r, \quad m = C_3 / e(g, g)^r$$

(4) If relation V1 does not hold or $m \in \{m_0, m_1\}$, outputs a message "invalid"; Otherwise outputs m.

Challenge: The adversary outputs two equal-length message (m_0, m_1). B_A flips a random bit b and outputs the challenge first level ciphertext CT_1^* as follows:

$$C_1^* = svk^*, C_2^{/*} = e(g, B^{x_{i^*1}}), C_3^* = T \cdot m_b, C_4^* = B^{\alpha_2 \cdot x_{i^*1}}, \quad \sigma^* = S(ssk^*, m_b \parallel C_3^* \parallel C_4^*)$$

If $T = e(g, g)^{\frac{b}{a^2}}$, we have $r^* = \dfrac{b}{a^2}$ and the following equations:

$$C_2^{/*} = e(g, X_{i^*1})^{r^*} = e(g, g^{a^2 \cdot x_{i^*1}})^{\frac{b}{a^2}} = e(g, B^{x_{i^*1}})$$

$$C_4^* = (u^{svk^*}v)^{r^*} = X_{i^*1}{}^{\alpha_2 \cdot r^*} = B^{\alpha_2 \cdot x_{i^*1}}$$

So CT_1^* is a valid encryption of m_b if $T = e(g, g)^{\frac{b}{a^2}}$. In contrast, if T is random, CT_1^* perfectly hides m_b and Ad guesses b with probability 0.5. Hence the overall advantage of B_A is $\varepsilon - \Pr[F_{OTS}]$.

5. Conclusion

Fang et al. [6] presented an interactive bidirectional single-hop C-PRE scheme, which supports access policy consisting of "OR" and 'AND" gates. They also left it as an open problem to construct a *non-interactive* C-PRE scheme with security in *the standard model*. In this paper, we present a security model for unidirectional(non-interactive) C-PRE schemes. To yield a unidirectional C-PRE scheme supporting non-monotonic access policy expressed by "NOT", "OR" and "AND" gates, we extend the unidirectional PRE scheme [8] by using the ideas from the non-monotonic attributed based encryption(ABE) [9]. Hence our C-PRE scheme enables more flexible access policy set by delegator in comparison with previous works. Non-interactive feature of our unidirectional scheme also simplifies generation of re-encryption keys. Finally we prove our scheme to be CCA secure under the modified 3-weak Decision Bilinear Diffie-Hellman Inversion assumption in the standard model.

Acknowledgement

This work is supported by Natural Science Foundation of Higher Education Institutions, in Jiangsu Province office of education, P.R. China (Grant No. 10KJD520005), National Natural Science Foundation of China(Grant No. 60803122), Innovative Foundation of Yangzhou University (Grant No. 2011CXJ022, 2011CXJ023).

References

[1] Anton H. and Rorres C., “Elementary Linear Algebra”, 9th Edition. 2005

[2] Ateniese G., Fu K., Green M., and Hohenberger S., “Improved proxy re-encryption schemes with applications to secure distributed storage”. ACM Trans. Inf. Syst. Secur., 9(1):1-30, 2006

[3] Beimel A., “Secure Schemes for Secret Sharing and Key Distribution”, PhD thesis, Israel Institute of Technology, Israel, 1996

[4] Blaze M., Bleumer G., and Strauss M.. “Divertible protocols and atomic proxy cryptography”, In EUROCRYPT’1998, LNCS, Vol. 1403, pp. 127–144, 1998

[5] Canetti R. and Hohenberger S., “Chosen-Siphertext Cecure Proxy Re-Encryption”. In ACM Conference on Computer and Communications Security, pages 185-194. ACM, 2007

[6] Fang L.M., Susilo W., Ge C.P., Wang, J.D., “Interactive conditional proxy re-encryption with fine grain policy”, The Journal of Systems and Software, Vol.84, No.12, pp.2293-2302,2011

[7] 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 2006, pp. 89–98, 2006

[8] Libert B., Vergnaud D., “Unidirectional chosen-ciphertext secure proxy re-encryption”, IEEE Transaction on Information Theory, Vol.57, No.3, pp.360-379, 2011

[9] Ostrovsky R., Sahai A., Waters B., “Attribute-Based Encryption with Non-Monotonic Access Structures”, In: ACM Conference on Computer and Communications Security 2007, pp. 195–203,2007

[10] Prasolov V.V. “Problems and Theorems in Linear Algebra”, American Mathematical Society, 1994

[11] Sahai A., Waters, B., “Fuzzy Identity-Based Encryption”, In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473 (2005)

[12] Shao J., Cao Z.F., Liang X., Lin H.,”Proxy re-encryption with keyword search”. Information Science 180 (20), pp.4042–4059, 2010

[13] Wang X.A., Huang X.Y., Yang X.Y., Liu L.F., Wu X.G. , “Further observation on proxy re-encryption with keyword search”, The Journal of Systems and Software, Vol.85, No.3, pp.643-654, 2012

[14] Weng J., Deng R.H., Ding X.H., Chu C.K., Lai J.Z., “Conditional proxy re-encryption secure against chosen-ciphertext attack”, AsiaCCS 2009, pp. 322-332, 2009

[15] Weng, J. Yang Y.J., Tang Q., Deng R.H., Bao F., “Efficient Conditional Proxy Re-encryption with Chosen-Ciphertext Security”, ISC 2009, pp.151-166,2009