A Subject-Delegated Decryption Scheme with "Tightly" Limited Authority

Lihua Wang[1], Takeshi Okamoto[1], Masahiro Mambo[2], and Eiji Okamoto[1]

Graduate School of Systems and Information Engineering,
University of Tsukuba, Tsukuba 305-8573, Japan
[1]{wlh,ken,okamoto}@risk.tsukuba.ac.jp,
[2]mambo@cs.tsukuba.ac.jp

Abstract. In this paper, we present a new proxy cryptosystem named subject-delegated decryption scheme, in which the original decryptor delegates decryption authority to multiple proxies according to different subjects. The advantage of our scheme is that the proxy authorities are tightly limited (*"Tightly" Limited Authority*). This means that the proxy authority can be temporarily aborted even if the validity period of the proxy key does not expire. Consequently, our protocol is more practical than the existential protocols because the secrecy of the original decryptor can be protected efficiently from his proxy, especially when the proxy becomes corrupted. Our scheme is efficient because the encryption method in our scheme is based on a hybrid of symmetric key and public key cryptographic techniques. We give the provable security using a variant decisional Bilinear Diffie-Hellman (BDH) assumption named *the Decisional β-BDH Assumption*.

Keywords: proxy cryptosystem, pairing, symmetric/public key cryptography.

1 Introduction

Proxy cryptosystems were invented by Mambo and Okamoto [8] for the delegation of the power to decrypt ciphertexts. In a proxy cryptosystem, there are three participants involved: encryptor Alice, original decryptor Bob and Bob's proxy Charlie.

1.1 Motivation

Let us consider the following scenario: A busy corporate manager, Bob, receives a large number of e-mail every day, which are encrypted using his public key. In order to release himself from his heavy office work, Bob partially delegates his decryption power to his secretaries, say Charlie, Clara etc., corresponding to various subjects. Here, the subjects might be established according to the duties corresponding to routine works, or the topics similar to the subject lines in the e-mail system (i.e., subjects may also be special projects such as the names

of trades). Suppose that Bob delegates the decryption power corresponding to some subject *sub* to Charlie, in order that Charlie can perform the decryption operation to the ciphertext under subject *sub*. From the convenience viewpoint, the encryptor Alice needs not know who Bob's proxies are. In fact, *sub* acts as proxy's "identity". So that Alice only needs to encrypt the plaintext with Bob's public key and *sub*.

In the real world, it is impossible for a person to trust others completely forever. When Charlie becomes corruptible and Bob dose not trust him anymore, it is desirable for Bob to abort Charlie's proxy authority even if the validity period of the proxy key does not expire. Meanwhile, the communication under the subject *sub* (the corresponding proxy authority was delegated to Charlie) can be continued securely. So the following properties should be strongly required:

(1) Bob can reveal any ciphertexts while Charlie can only reveal the ciphertexts under the subject he was delegated.
(2) The proxy authority should be limited to some validity period. In other words, the authority will be discontinued automatically after the validity period.
(3) The proxy authority can be temporarily aborted if it is necessary, even if it is during the valid proxy period distributed to Charlie.

The first two properties is called *authorization-limited* in [13] and [14]. We call a proxy decryption system with *"Tightly" Limited Authority* if the above three properties are satisfied.

1.2 Related Works and Our Contributions

In Mambo-Okamoto proxy cryptosystems, Alice encrypts a plaintext and transmits its ciphertext to Bob. In a conventional proxy cryptosystem, after a transformation of the ciphertext into another ciphertext, Bob forwards the new ciphertext to Charlie, his proxy decryptor, to recover the plaintext. That is to say Bob can not be really released from his heavy work because he has to transform the ciphertext into another ciphertext before Charlie works every time. So the efficiency of original Mambo-Okamoto proxy cryptosystems [8] is not good.

Series ciphertext transformation-free proxy cryptosystems, in which the proxy (or proxies) can do the decryption operation without ciphertexts transformation (e.g., [10], [12], [13] and [14]), have also been studied.

Referring to the above research works, three types of delegation formation are studied: one proxy is delegated to do all the decryption work for the original decryptor, such as Mambo-Okamoto schemes in [8]; a number of proxies are delegated to do the same decryption works for the original decryptor, such as MVN scheme in [10]; a number of proxies are delegated to do different decryption works in terms of subjects for the original decryptor, such as HEAD in [12] and AL-TFP systems in [13, 14], which is corresponding to the scenario described in Section 1.1. We name the last one a subject-delegated decryption scheme in this paper (see Section 1.1).

The HEAD proposed by Sarkar [12] is a hybrid encryption system, whose advantage is efficient, since in his scheme messages are encrypted by symmetric algorithms and signed by PKC. Due to the limited authorization, the AL-TFP systems proposed by Wang et al. [13, 14] have the advantage that the message is graded into two secrecy levels so that the important message between the encryptor and the original decryptor can be protected secretly by restricting the proxy decryptor's power.

Unfortunately, neither HEAD [12] nor AL-TFP systems [13, 14] can tightly limit the proxy authorities. That is to say, the proxy authority cannot be aborted during the distributed validity period. Consequently, if Charlie has been corrupted, the communication under the corresponding subject has to be stopped during Charlie's valid proxy period.

Our scheme is also hybrid encryption system as the HEAD [12]. As described in [12], DHIES [2] is a hybrid encryption scheme which is presented in draft standards of IEEE P1636a and ANSI X9.63 [3]. It combines an ElGamal type encryption with a symmetric encryption and a message authentication code (MAC) generation scheme. DHIES is an efficient encryption scheme and has been proved to be secure under the oracle Diffie-Hellman assumption; chosen plaintext security for symmetric encryption and unforgeable chosen message security for the MAC scheme. Kurosawa and Matsuo [7] showed that the MAC component can be removed from DHIES. Instead, they require the symmetric encryption scheme to satisfy chosen ciphertext security. Following [7], by doing away with the MAC component and using only a symmetric encryption scheme, Sarkar modified DHIES to achieve delegated decryption functionality. His main innovation is in the replacement of the public key part of DHIES.

In this paper, by reconstructing the ID-based encryption scheme introduced by Waters in [15], we modify the public key part of HEAD to construct a new subject-delegated decryption scheme, so that all of the conditions described in Section 1.1 are satisfied. In other words, our scheme is the first proxy decryption scheme with "Tightly" Limited Authority.

> "Tightly" Limited Authority: the proxy authority is possible to be aborted temporarily even if the validity period of the proxy key does not expire, at the same time that the communication under the subject *sub* (the corresponding proxy authority was delegated to Charlie) can be continued securely.

The rest of this paper is organized as follows. In Section 2, we introduce the properties of pairing and the complexity assumptions briefly. In Section 3, we describe the formal definition of subject-delegated decryption scheme and then propose our new protocol. In Section 6, we analyze the security of our scheme. Then, in order to overcome the lack of the proposed scheme, we devise a variant scheme in Section 4. Finally, a brief conclusion is drawn in Section 6.

2 Preliminaries

Our protocols are based on bilinear pairing which was first used to generate cryptosystems independently by Sakai et al. [11] in 2000 and Boneh et al. [5]

in 2001. In this section, we recall the basic properties related to bilinear maps between groups and introduce the corresponding complexity assumptions briefly.

2.1 Pairings

Let G_1, G_2 be two multiplicative groups with prime order p and g be a generator of G_1. G_1 has an admissible bilinear map, $\hat{e} : G_1 \times G_1 \longrightarrow G_2$, into G_2 if the following three conditions hold:

(1) *Bilinear.* $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$ for all $a, b \in Z_p$.
(2) *Nondegenerate.* $\hat{e}(g, g) \neq 1$.
(3) *Computable.* There is an efficient algorithm to compute $\hat{e}(A, B)$ for any $A, B \in G_1$.

2.2 Complexity Assumptions

(1) BDH Assumption: We assume that the BDH problem is hard in $\langle G_1, G_2, \hat{e}\rangle$, which means that there is no efficient algorithm to solve the BDH problem with non-negligible probability.

Let $Suc^{BDH}(g, g^a, g^b, g^c)$ denote the event that the BDH problem in $\langle G_1, G_2, \hat{e}\rangle$ is solved, that is, if $\langle g, g^a, g^b, g^c\rangle$ for some $a, b, c \in Z_q \setminus \{0\}$ are known where g is a generator of G_1, then $\hat{e}(g, g)^{abc} \in G_2$ can be computed. Therefore, according to the BDH assumption, the probability

$$Pr[Suc^{BDH}(g, g^a, g^b, g^c)]$$

is negligible in our schemes.

As usual, we say that a function $h : R \rightarrow R$ is a negligible if $h(k)$ is smaller than $1/f(k)$ for any polynomial f, where k is the security parameter.

(2) Decisional Bilinear Diffie-Hellman Assumption (Decisional BDH Assumption) [15]: The challenger chooses $a, b, c, z \in_R Z_p$ and then flips a fair binary coin γ. If $\gamma = 1$ it outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = \hat{e}(g, g)^{abc})$. Otherwise, if $\gamma = 0$, the challenger outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = \hat{e}(g, g)^z)$. The adversary must then output a guess γ' of γ. An adversary, $\mathcal{B}$, has at least an ϵ advantage in solving the decisional BDH problem if

$$|Pr[\mathcal{B}(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc}) = 1] - Pr[\mathcal{B}(g, g^a, g^b, g^c, \hat{e}(g, g)^z) = 1]| \geq \epsilon$$

where the probability is over the randomly chosen a, b, c, z and the random bits consumed by $\mathcal{B}$.

Definition 1 *The decisional (t, ϵ)-BDH assumption holds if no t-time adversary has at least ϵ advantage in solving the above game.*

Similarly, we define the β appended decisional BDH assumption (Decisional β-BDH Assumption) as follows:

(3) Decisional β-BDH Assumption: The challenger chooses $a, b, c, z \in_R Z_p$ and then flips a fair binary coin γ. If $\gamma = 1$ it outputs the tuple $(g, A = g^a, B = g^b, C = g^c, \beta, \beta^c, Z = \hat{e}(g,g)^{abc})$, where $\beta \in G_1$. Otherwise, if $\gamma = 0$, the challenger outputs the tuple $(g, A = g^a, B = g^b, C = g^c, \beta, \beta^c, Z = \hat{e}(g,g)^z)$. The adversary must then output a guess γ' of γ. An adversary, $\mathcal{B}$, has at least an ϵ advantage in solving the decisional β-BDH problem if

$$|Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, \hat{e}(g,g)^{abc}) = 1] - Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, \hat{e}(g,g)^z) = 1]| \geq \epsilon$$

where the probability is over the randomly chosen a, b, c, z and the random bits consumed by $\mathcal{B}$.

Definition 2 *The decisional (t, ϵ) β-BDH assumption holds if no t-time adversary has at least ϵ advantage in solving the above game.*

It is clear that the decisional (t, ϵ)-BDH assumption holds, if the decisional (t, ϵ) β-BDH assumption holds.

3 Our New Subject-DD Scheme with "Tightly" Limited Authority

In this section, we give the formal definitions of subject-delegated decryption (subject-DD) scheme and then propose a new subject-DD protocol.

3.1 Definitions

Definition 3 *A subject-delegated decryption (subject-DD) scheme ($\mathcal{G}$, $\mathcal{PA}$, $\mathcal{E}$, $\mathcal{D}$, $\mathtt{Random}$, $\mathtt{Mspace}$) is a 4-tuple of algorithms associated with two finite sets, $\mathtt{Random}(k)$, $\mathtt{Mspace}(k) \subseteq \{0,1\}^*$, for $k \in N$ under subject $sub \in \{sub\}$, where:*

- *$\mathcal{G}$, called the key generation algorithm, is a probabilistic algorithm which on input 1^k outputs a public/private key pair $(pk, sk) \leftarrow \mathcal{G}(1^k)$.*
- *$\mathcal{PA}$, called the proxy-authorization algorithm, is an algorithm which on input a private key sk and subject sub outputs a corresponding proxy key $sk^{sub} \leftarrow \mathcal{PA}(sk, sub)$.*
- *$\mathcal{E}$, called the encryption algorithm, is a probabilistic algorithm which on input a public key pk, a plaintext $x \leftarrow \mathtt{Mspace}(k)$, and a random number $r \leftarrow \mathtt{Random}(k)$ under subject line sub, outputs the ciphertexts $C_{sub} \leftarrow \mathcal{E}(pk; x; r; sub)$.*
- *$\mathcal{D} = (\mathcal{D}_{basic}, \mathcal{D}_{sub})$, called the decryption algorithm, is a deterministic algorithm which consists of an original decryption algorithm $\mathcal{D}_{basic}$ and a proxy decryption algorithm $\mathcal{D}_{sub}$, which on input a private key and a ciphertext C_{sub} outputs the corresponding plaintext x, where*

$$x = \mathcal{D}_{basic}(sk; C_{sub}); \;\; x = \mathcal{D}_{sub}(sk^{sub}; C_{sub}).$$

In a subject-DD scheme, the subject *sub* acts as the "identity" of the proxy. HEAD in [12] and AL-TFP systems in [13, 14] are subject-DD schemes.

Definition 4 *A symmetric encryption scheme* $(\mathcal{G}^{sym}, \mathcal{E}^{sym}, \mathcal{D}^{sym}, \mathtt{Mspace}(k_S))$ *is a 3-tuple of algorithms associated with a finite set,* $\mathtt{Mspace}(k_S) \subseteq \{0,1\}^*$*, for* $k_S \in N$*, where:*

- $\mathcal{G}^{sym}$*, called the key generation algorithm, is a probabilistic algorithm which on input* 1^{k_S} *outputs a symmetric key* $sk^{sym} \leftarrow \mathcal{G}(1^{k_S})$.
- $\mathcal{E}^{sym}$ *is the encryption algorithm which on input a key* sk^{sym}*, a plaintext* $M \leftarrow \mathtt{Mspace}(k_S)$ *and outputs the ciphertext* $C \leftarrow \mathcal{E}^{sym}(sk^{sym}; M)$.
- $\mathcal{D}^{sym}$ *is the decryption algorithm which on input a key* sk^{sym} *and a ciphertext* C *outputs the corresponding plaintext* $M \leftarrow \mathcal{D}^{sym}(sk^{sym}; C)$.

3.2 Our Protocol

In this subsection, by reconstructing the ID-based encryption scheme introduced by Waters in [15], we modify the public key part of HEAD to construct a new subject-delegated decryption scheme. We describe the four algorithms in our new subject-DD scheme as follows.

$\mathcal{G}$ **(Original Key Generation):** The public/private keys are generated as follows:

- Choose random values $u_i \in G_1$, $i = 0, 1, ..., n$, a random generator g of G_1, a secret integer $\alpha \in_R Z_p$, and two non-collision injection functions $H : G_2 \rightarrow \{0,1\}^{k_S}$ and $H' : \{0,1\}^* \rightarrow G_1$.
- Set the value $g_{pub} = g^{\alpha}$, choose g_U randomly in G_1 as Bob's public key and compute Bob's private key $sk_U = (g_U)^{\alpha}$. The parameters $(u_i (i = 0, 1, ..., n)$, g, g_{pub}, g_U, $H, H')$ are published while the private key sk_U is held secretly by Bob himself.

$\mathcal{PA}$ **(Proxy Key Generation):** Bob generates proxy keys for his l proxies as follows:

- Set l subjects $sub_j \in \{0,1\}^n$, $j = 1, ..., l$ for some $n \in N$.
- For a $sub \in \{sub_1, ..., sub_l\}$, compute the subject proxy key $sk_U^{sub} = (sk_1, sk_2)$ $= ((g_U)^{\alpha}\beta^d, g^d)$ for $d \in_R Z_p$, where $sub = (s_1 s_2 ... s_n)_2$ is the binary of sub, i.e.,
$$sub = s_1 \cdot 2^{n-1} + s_2 \cdot 2^{n-2} + ... + s_n,$$
$s_i \in \{0,1\}$ for $i = 1, ..., n$ and $\beta = u_0 \prod_{i=1}^{n} u_i^{s_i}$.
- sk_U^{sub} is distributed to the assistant in terms of the subject, sub, secretly by Bob.

$\mathcal{E}$ **(Encryption):** To encrypt message M under subject sub, where $sub = (s_1 s_2 ... s_n)_2$, $s_i \in \{0,1\}$, the encryptor, Alice,

- computes $\beta = u_0 \prod_{i=1}^{n} u_i^{s_i}$,

- selects $r \in_R Z_p$ and computes symmetric key $sk^{sym} = H(\hat{e}((g_{_U})^r, g_{pub}))$,
- computes the ciphertext $C = (C_1, C_2, C_3)$, where

$$C_1 = \mathcal{E}^{sym}(sk^{sym}; M), C_2 = g^r, C_3 = \beta^r.$$

- sends $\langle sub, C \rangle$ to Bob.

$\mathcal{D}$ (Basic/Proxy decryption):

- $\mathcal{D}_{basic}$: Bob can perform the decryption operation of the ciphertext under any subject, *sub*, by computing

$$sk^{sym} = H(\hat{e}(sk_{_U}, C_2)).$$

- $\mathcal{D}_{sub}$: Charlie, Bob's proxy under *sub*, computes

$$sk^{sym} = H(\frac{\hat{e}(C_2, sk_1)}{\hat{e}(C_3, sk_2)}).$$

Then the plaintext can be obtained as follows:

$$M = \mathcal{D}^{sym}(sk^{sym}, C_1).$$

3.3 The correctness

The correctness of this scheme can be proved as follows:

For $\mathcal{D}_{basic}$, we have

$$\hat{e}(C_2, (g_{_U})^\alpha) = \hat{e}(g^r, (g_{_U})^\alpha) = \hat{e}(g_{pub}, (g_{_U})^r).$$

For $\mathcal{D}_{sub}$, we have

$$\frac{\hat{e}(C_2, sk_1)}{\hat{e}(C_3, sk_2)} = \frac{\hat{e}(g^r, (g_{_U})^\alpha \beta^d)}{\hat{e}(\beta^r, g^d)} = \hat{e}(g^r, (g_{_U})^\alpha) = \hat{e}(g_{pub}, (g_{_U})^r).$$

3.4 Can the Proxy Authority be Limited Tightly?

Recall the description about "tightly" limited authority in Section 1.1. "Tightly" means that the proxy authority can be temporarily aborted if it is necessary, even if it is during the valid proxy period distributed to the proxy. Here, there are two cases should be considered.

(1) *Limitation from encryptor* : the encryptor Alice might send a ciphertext under *sub* which cannot be decrypted by the proxy Charlie.
(2) *Limitation from decryptor* : the original decryptor Bob might stop the proxy authority distributed to Charlie, even if it does not expire.

Note that, in our new subject-DD scheme provided in Section 3.2, Bob can decrypt the ciphertext under any subject only with his master private key. This implies case (1). In detail, when Alice wants to keep the information under subject, *sub*, secret from the assistant, Charlie, in terms of *sub*, she needs only to delete C_3 or let $C_3 = 0$.

However, when Charlie becomes corruptible, and Bob does not trust him any more, Bob cannot stop the proxy authority directly in the distributed proxy period, if Alice has sent the ciphertext according to rule. So in order to abort Charlie proxy authority, and in the same time let the communication under subject *sub* continue securely Bob needs to cope with this situation as follows:

- Stop to distribute the proxy keys to Charlie, which correspond to the following periods.
- Broadcast a *sub*-alert on period τ to the parties concerned.
- Run the basic decryption algorithm with the master private key by himself during the period remained.
- Delegate the proxy decryption power to other assistant from the next period.

As the correspondence for the above situation, the parties concerned deal with it as follows:

- Set $C_3 = RandomNumber$ instead of β^r when they send the messages under subject *sub* in period τ.
- Come back to the normal operation from the next period.

Note: The proposed scheme has the lack that Bob cannot stop the proxy authority directly in the distributed proxy period, if the encryptor has sent the ciphertext according to rule. Therefore he has to broadcast an *sub*-alert, whose validity can be verified, to the parties concerned. In the next section, we devise a variant scheme to solve this problem.

4 A Variant Scheme with Bulletin Board

In this variant scheme, we use a bulletin board for the proxies to limit the proxy authority tightly.

4.1 The Outline of This Variant Scheme

We show the outline of this variant scheme in Fig.1. In detail,

- Bob distributes proxy keys to his proxies in advance, e.g., sk_{sub} denotes the proxy key that Bob distributes to Charlie, who is delegated the proxy authority under subject, *sub*;
- Bob often (e.g., once a day) renews the bulletin board parameters with time stamp which are published on his private bulletin board for his proxies;

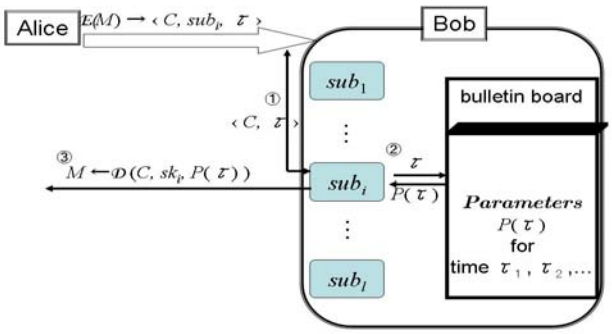

Fig. 1. The outline of the new scheme

- On receipt of the ciphertext corresponding to subject, *sub*, Charlie, the proxy under *sub*, fetches the parameter $P(\tau)$, then decrypts the ciphertext corresponding to subject, *sub*, using both sk_{sub}, the proxy key in terms of *sub*, and $P(\tau)$, the bulletin board parameter with time stamp.

Bob has the power to change the parameter any time so long as he needs. Consequently, this variant scheme is in fact a special subject-DD scheme, in which the proxy authority is limited tightly with the aid of the bulletin board.

4.2 Our protocol

$\mathcal{G}$ **(Original Key Generation):** It is the same to the original key generation algorithm $\mathcal{G}$ in the above proposed scheme.

$\mathcal{PA}$ **(Proxy Key Generation):** Bob generates proxy keys for his l proxies. For example, Charlie, whose authority is under subject *sub* $(= (s_1 s_2 ... s_n)_2)$, is one of them. Where $(s_1 s_2 ... s_n)_2$ is the binary of *sub*, i.e.,

$$sub = s_1 \cdot 2^{n-1} + s_2 \cdot 2^{n-2} + ... + s_n,$$

$s_i \in \{0, 1\}$ for $i = 1, ..., n$.

There are two steps have to be done.

- *Proxy key distribution*: Bob selects $b_0, b_1 \in Z_p \backslash \{0\}$ such that $b_0 + b_1 = 1 \bmod p$ for his proxies, and $d_{sub} \in_R Z_p \setminus \{0\}$ for Charlie whose proxy authority is under *sub*, then computes personal proxy key $sk_{sub} = (sk_1, sk_2)$ for Charlie as follows:

$$sk_1 = sk_U^{b_0} \beta^{d_{sub}} = g_U^{\alpha b_0} \beta^{d_{sub}}, sk_2 = g^{d_{sub}},$$

where $\beta = u_0 \prod_{i=1}^{n} u_i^{s_i}$.

- *Bulletin board parameters renovation*: Bob selects $d^{(\tau)} \in_R Z_p \backslash \{0\}$, computes $P(\tau) = (x(\tau), y(\tau))$ as follows:

$$x(\tau) = sk_{_U}^{b_1} H'(\tau)^{d^{(\tau)}} = g_{_U}^{\alpha b_1} H'(\tau)^{d^{(\tau)}},\ y(\tau) = g^{d^{(\tau)}}.$$

  Then he publishes the parameter $P(\tau)$ to his private bulletin board for his proxies.

In the above proxy key generation algorithm, the parameters b_0, b_1, $d^{(\tau)}$ ($\tau = \tau_1, \tau_2, ...,$) and d_{sub} ($sub \in \{sub_1, ..., sub_l\}$) must be held secretly by Bob himself.

$\mathcal{E}$ (Encryption): To encrypt message M under subject sub during period τ, the encryptor, Alice:

- selects $r \in_R Z_p$ and computes $sk^{sym} = H(\hat{e}((g_{_U})^r, g_{pub}))$;
- computes the ciphertext $C = (C_1, C_2, C_3, C_4)$, where

$$C_1 = \mathcal{E}^{sym}(sk^{sym}; M), C_2 = g^r, C_3 = \beta^r, C_4 = H'(\tau)^r;$$

- sends $\langle sub, \tau, C \rangle$ to Bob.

$\mathcal{D}$ (Basic/Proxy decryption):

- $\mathcal{D}_{basic}$: Bob can perform the decryption operation of the ciphertext under any subject, sub, by computing

$$sk^{sym} = H(\hat{e}(sk_{_U}, C_2)).$$

- $\mathcal{D}_{sub}$: Charlie, Bob's proxy under sub, computes

$$sk^{sym} = H(\frac{\hat{e}(sk_1 x(\tau), C_2)}{\hat{e}(C_3, sk_2)\hat{e}(C_4, y(\tau))}).$$

  Then the plaintext can be obtained as follows:

$$M = \mathcal{D}^{sym}(sk^{sym}, C_1).$$

4.3 The correctness

The correctness can be proved as follows:

For $\mathcal{D}_{basic}$, we have

$$\hat{e}(sk_{_U}, C_2) = \hat{e}(g_{_U}^{\alpha}, g^r) = \hat{e}((g_{_U})^r, g_{pub});$$

For $\mathcal{D}_{sub}$, we have

$$\begin{aligned}
&\hat{e}(sk_1 x(\tau), C_2) \\
&= \hat{e}(g_{_U}^{\alpha} \beta^{d_{sub}} H'(\tau)^{d^{(\tau)}}, g^r) \\
&= \hat{e}(g_{_U}^{\alpha}, g^r)\hat{e}(\beta^{d_{sub}}, g^r)\hat{e}(H'(\tau)^{d^{(\tau)}}, g^r) \\
&= \hat{e}(g_{_U}^r, g^{\alpha})\hat{e}(\beta^r, g^{d_{sub}})\hat{e}(H'(\tau)^r, g^{d^{(\tau)}}) \\
&= \hat{e}((g_{_U})^r, g_{pub})\hat{e}(C_3, sk_2)\hat{e}(C_4, y(\tau)).
\end{aligned}$$

4.4 How to Limit the Proxy Authority Tightly

Recall the description about "tightly" limited authority in Section 1.1. "Tightly" means that the proxy authority can be temporarily aborted if it is necessary, even if it is during the valid proxy period distributed to the proxy. Here, there are two cases should be considered.

(1) *Limitation from encryptor* : the encryptor Alice might send a ciphertext under *sub* which cannot be decrypted by the proxy Charlie.

Note that, in the special subject-DD scheme with bulletin board provided in Section 4.2, Bob can decrypt the ciphertext under any subject only with his master private key by performing the basic decryption operation $\mathcal{D}_{basic}$. This implies case (1). In detail, when Alice wants to keep the information under subject, *sub*, secret from the proxy, Charlie, in terms of *sub*, she needs only to delete C_3, C_4 or let $C_3 = C_4 = 0$.

(2) *Limitation from decryptor* : the original decryptor Bob might stop the proxy authority distributed to Charlie, even if it does not expire.

When Charlie who is under subject *sub* becomes corruptible, and the original decryptor Bob does not trust him any more. In order to maintain the secrecy and in the same time have the communication under subject *sub* continued, Bob can cope with this situation by changing original parameters b_0, b_1, d_{sub} into new parameters $b'_0, b'_1, d'_{sub} \in Z_p \setminus \{0\}$, where $b_i \neq b'_i$ for $i = 0, 1$, $d_{sub} \neq d'_{sub}$, and $b'_0 + b'_1 = 1 \bmod p$. In detail,

- Renew the data $P(\tau)$ on the private bulletin board corresponding to the new parameter b'_1.
- Renew the proxy keys for other proxies in terms of the new parameter b'_0.
- Perform the basic master decryption before the decryption authority is delegated to another assistant under the above new parameters b'_0, d'_{sub}.

4.5 Security Requirements

The general requirements of secure proxy cryptsystems include the following three aspects:

1. Outsiders can perform neither basic decryption nor proxy decryption operation;
2. Proxies cannot perform basic decryption operation;
3. Any proxy cannot perform the proxy decryption operation in terms of the subjects which are not delegated to him/her.

We say a subject-DD scheme secure if it satisfies the above three requirements.

Apart from the above three requirements for general secure subject-DD scheme, a secure subject-DD scheme with tightly limited authority also requires that the proxy whose authority is limited cannot decrypt any new ciphertexts under his/her subject

4. the proxy whose authority is limited cannot decrypt any new ciphertexts under his/her subject using proxy key only.
5. the proxy whose authority is limited cannot decrypt any new ciphertexts under his/her subject even if he/she can obtain the new parameter $P(\tau)$.
6. the proxy whose authority is limited cannot decrypt any new ciphertexts under his/her subject even if he/she can collude with other proxies under different subject.

Consider the attacks to the limitation from encryptor and the limitation from decryptor (see Section 4.4) respectively. We analyze the security of the new proposed scheme under the BDH assumption, if the algorithm $\mathcal{PA}$ is properly carried out.

(1) Adversary, Charlie who suffers the limitation from encryptor, tries to decrypt the ciphertext under subject *sub*.

Because the ciphertext only includes C_1, C_2, he has to perform basis decryption operation without private key. Our scheme is secure because it satisfies requirements 1. and 2. under BDH assumption.

(2) Adversary, Charlie who suffers the limitation from decryptor, tries to decrypt the ciphertext (C, sub, τ'), where $C = (C_1, C_2, C_3, C_4)$.

That is to say, Charlie tries to perform proxy decryption with proxy key only. Of course he has some old parameters for past periods τ_i ($\tau_i < \tau'$), and he might also collude with the proxies, whose proxy authorities are not limited, under other subjects. In the following context, we show that Charlie cannot decrypt the ciphertext using his proxy key and old parameters for past periods, even if he can obtain the new data, $P(\tau') = (x(\tau'), y(\tau'))$, on Bob's private bulletin board and collude with the proxies under other subjects.

From the parameter $P(\tau') = (x(\tau'), y(\tau'))$, $\hat{e}(g_U^{\alpha}, g^{r'})^{b_1'}$ can be cumputed. On the other hand, from Charlie's proxy key $sk_{sub} = (sk_1, sk_2)$, where $sk_1 = g_U^{\alpha b_0}\beta^{d_{sub}}$ and $sk_2 = g^{d_{sub}}$, $\hat{e}(g_U^{\alpha}, g^{r'})^{b_0}$ can be computed. The secret parameters b_0 and b_1' are independent each other, so the security is based on the hardness of the discrete logarithm problem in G_2. In order to obtain $\hat{e}(g_U^{\alpha}, g^{r'})^{b_0'}$ corresponding to the new parameter b_0' which satisfies $b_0' + b_1' = 1 \bmod p$, Charlie corrupts the proxy under subject sub_c, $sub_c \neq sub$, whose proxy key $sk^{(sub_c)} = (sk_1^{(sub_c)}, sk_2^{(sub_c)})$ has been renewed by Bob:

$$sk_1^{(sub_c)} = g_U^{\alpha b_0'}\beta_c^{d_{sub_c}},\ sk_2^{(sub_c)} = g^{d_{sub_c}}.$$

Accordingly,

$$\hat{e}(sk_1^{(sub_c)}, g^{r'}) = \hat{e}(g_U^{\alpha}, g^{r'})^{b_0'}\hat{e}(\beta_c^{d_{sub_c}}, g^{r'})$$

Therefore, obtaining $\hat{e}(g_U^{\alpha}, g^{r'})^{b_0'}$ is equivalent to computing $\hat{e}(\beta_c^{d_{sub_c}}, g^{r'}) = \hat{e}(\beta_c^{r'}, g^{d_{sub_c}})$ when $\beta_c, \beta, \beta^{r'}$ and $g, g^{r'}, sk_2^{(sub_c)} = g^{d_{sub_c}}$ are given.

First of all, neither $\beta_c^{d_{sub_c}}$ nor $\beta_c^{r'}$ can be computed using the above given parameters. If the adversary, Charlie, can produce $\beta_c^{d_{sub_c}}$ or $\beta_c^{r'}$, then from the

fact that there exists an $a \in Z_p \setminus \{0\}$ such that $\beta_c = g^a$, and there exists an $f \in Z_p \setminus \{0\}$ such that $\beta = g^f$, we will conclude that Charlie can solve the CDH problem in G_1: "Given $g, g^a, g^{r'}$, compute $g^{ar'}$", or "Given $g, g^a, g^{fd_{sub_c}}$, compute $g^{afd_{sub_c}}$", which contradicts with the BDH assumption because the BDH assumption implies that the CDH problem is hard.

Next, the pairing $\hat{e}(\beta_c^{r'}, g^{d_{sub_c}})$ cannot be computed using the above given parameters. If the adversary, Charlie, can produce $\hat{e}(\beta_c^{r'}, g^{d_{sub_c}})$, we will conclude that Charlie can solve the BDH problem in $\langle G_1, G_2, \hat{e}\rangle$: "Given g, g^a, $g^{r'}$, $g^{d_{sub_c}}$, compute $\hat{e}(g,g)^{ar'd_{sub_c}}$ ", which contradicts with the BDH assumption.

5 Discussion

5.1 Bulletin Board and Ciphertext Transformation

In Mambo-Okamoto scheme [8], Bob can limit the proxy authority "tightly", because his proxy cannot do decryption at all if he does not transfer the ciphertext for his proxy. In our scheme, Bob limit the proxy authority with the aid of bulletin board. Bob renews the parameter on the bulletin board once a day for his proxies. So the computation cost is nearly $\frac{1}{n(\tau)}$ of that in Mambo-Okamoto scheme, where $n(\tau)$ denotes the number of the ciphertexts that Bob received on the day, τ. So our variant scheme limits the proxy authority tightly and is more flexible and efficient than Mambo-Okamoto scheme.

5.2 Encryption under Multi-subject

Alice can send a ciphertext of a message m, which is in terms of several subjects, once if she likes. For example, she can send m to Bob under subjects sub_1 and sub_2, by computing $C_3 = \{\beta_1^r, \beta_2^r\}$ instead of the original one. In that case, it can be easily verified that either the proxy under sub_1 or the proxy under sub_2 can decrypt the ciphertext. Also in MVN scheme [10], a number of proxies are delegated to do the same decryption works for the original decryptor, but the encryptor cannot "select" the subjects/proxies. In a word, our scheme can fulfill almost all of the tasks that the previous related works do.

5.3 Application to the Key-insulated Scheme

The notion of key-insulated public key cryptosystem was first introduced by Dodis et. al [6] to minimize the damage of key exposures, in which a user begins by registering a single public key which remains for the lifetime of the scheme. Exposure of secret keys is perhaps the most devastating attack on a cryptosystem since it typically means that security is entirely lost. This problem is probably the greatest threat to cryptography in the real world: in practice, it is typically easier for an adversary to obtain a secret key from a naive user than to break the computational assumption on which the system is based. The threat is increasing

nowadays with users carrying mobile devices which allow remote access from public or foreign domains.

The secret key associated with a public key is here shared between the user and a physically-secure device. The master key is stored on a physically-secure device and a temporary secret key used to perform cryptographic operations is stored in an insecure device and updated regularly with the help of a physically-secure device that stores a master key.

Our varaint scheme can be used to construct an identity-based key-insulated scheme. We will describe it in the next version.

6 Conclusion

In this paper, we presented a new subject-delegated decryption protocol and its variant: a special subject-delegated decryption scheme with bulletin board. This variant scheme is a proxy decryption scheme with *tightly* limited authority. It means that the proxy authority can be temporarily aborted even if the validity period of the proxy key does not expire. Therefore our scheme is more practical than the existential protocols in protecting the privacy of the original decryptor from his proxies, because that even if during the validity period of the proxy key, the proxy authority can be aborted if it is necessary. We proved that the security of our scheme is based on a variant decisional BDH assumption and a secure symmetric encryption scheme.

References

1. M. Abdalla, M. Bellare and P. Rogaway. The oracle Diffie-Hellman assumption and an analysis of DHIES, *Topics in Cryptology - CT-RSA !@1*, LNCS 2020, pp. 143-158, Springer-Verlag, Berlin, 2001.
2. M. Abdalla, M. Bellare and P. Rogaway. DHIES : An encryption scheme based on the Diffie-Hellman problem, full version of [1] and [4], 2001.
3. American National Standards Institute (ANSI) X9.F1 sub-committee. ANSI X9.63 Public Key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Transport Schemes, working draft version 2.0. 1998.
4. M. Bellare and P. Rogaway. Minimizing the use of random oracles in authenticated encryption schemes, *Information and Communications Security*, LNCS 1334, pp. 1-16, Springer-Verlag, Berlin, 1997.
5. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing, *Advances in Cryptology - CRYPTO 2001*, LNCS 2139, pp. 213-229, Springer-Verlag, Berlin, 2001.
6. Y. Dodis, J. Katz, S. Xu and M. Yung. Key-Insulated Public Key Cryptosystems, *Advances in Cryptology - EUROCRYPT 2002*, LNCS 2332, pp. 65-82, Springer-Verlag, Berlin, 2002.
7. K. Kurosawa and T. Matsuo. How to remove MAC from DHIES, *Information Security and Privacy: 9th Australasian Conference (ACISP 2004)*, LNCS 3108, pp. 236-247, Springer-Verlag, Berlin, 2004.

8. M. Mambo and E. Okamoto. Proxy cryptosystem: Delegation of the power to decrypt ciphertexts, *IEICE Trans. Fundamentals*, Vol. E80–A, No.1, pp. 54-63, 1997.
9. M. Mambo, K. Usuda and E. Okamoto. Proxy signatures for delegating signing operation, *Proceedings of 3rd ACM Conference on Computer and Communications Security*, pp. 48-57, 1996.
10. Y. Mu, V. Varadharajan and K. Q. Nguyen. Delegation decryption, *IMA - Crypto & Coding'99*, LNCS 1746, pp. 258-269, Springer-Verlag, Berlin, 1999.
11. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing, *SCIS2000-C20*, 2000.
12. P. Sarkar. HEAD: hybrid encryption with delegated decryption capbility, *Advances in Cryptology - INDOCRYPTO 2004*, LNCS 3348, pp. 230-244, Springer-Verlag, Berlin, 2004.
13. L. Wang, Z. Cao, E. Okamoto, Y. Miao and T. Okamoto. Transformation-free proxy cryptosystems and their applications to electronic commerce, *Proceeding of International Conference on Information Security (InfoSecu'04)*, pp. 92-98, ACM Press, 2004.
14. L. Wang, Z. Cao, T. Okamoto, Y. Miao and E. Okamoto. Authorization-limited transformation-free proxy cryptosystems and their security analyses, *IEICE Trans. Fundamentals*, Vol. E89–A, No.1, pp. 106-114, 2006.
15. B. Waters. Efficient identity-based encryption without random oracles, *Advances in Cryptology - EUROCRYPT 2005*, LNCS 3494, pp. 114-127, Springer-Verlag, Berlin, 2005.

Appendix: Security Analysis

In this section, we describe the security notions in Section 6, and then show that the security of our scheme described in Section 3.2 is based on the decisional β-BDH assumption and the security of the symmetric encryption algorithm being used in our scheme.

A. Security Notions

We present the definition of semantic security against the chosen ciphertext attack (CCA) for a subject-DD scheme with l subjects being delegated, which was first described by Sarkar [12].

(1) Semantic Security against the CCA for a Subject-DD Scheme Consider the following game played by an adversary.
Setup. The challenger generates the public parameters and gives them to the adversary.
Phase 1. The adversary $\mathcal{A}$ has access to a decryption oracle, which is the decryption algorithm instantiated by a randomly chosen secret (i.e. unknown to the adversary) key. In other words, the adversary $\mathcal{A}$ can ask to the decryption oracle for ciphertexts under any subject $sub \in \{sub_1, ..., sub_l\}$ and receive the corresponding plaintexts.

Challenge. The adversary $\mathcal{A}$ submits two messages M_0, M_1. The challenger flips a fair binary coin, γ, encrypts M_γ using l subjects $sub_i, i = 1, 2, ..., l$, and then gives the l corresponding targets $T_1, T_2, ..., T_l$ to $\mathcal{A}$.
Phase 2. Phase 1 is repeated with the natural restriction that the adversary cannot query the decryption oracle on the targets $T_1, T_2, ..., T_l$.
Guess. The adversary $\mathcal{A}$ submits a guess γ' of γ.

The adversary $\mathcal{A}$'s advantage in breaking the system is defined to be

$$Adv_{\mathcal{A}}^{Sub-DD} = 2|Pr[\gamma' = \gamma] - 1/2|.$$

The system's advantage is defined as

$$Adv^{Sub-DD}(t, q) = \max_{\mathcal{A}} Adv_{\mathcal{A}}^{Sub-DD}$$

where the maximum is taken over all adversaries running in time at most t and making at most q queries to the decryption oracle.

(2) Semantic Security against the CCA for a Symmetric Encryption Scheme
The usual model of security is extended in the following manner. First l keys $K_1, ..., K_l \in \{0,1\}^{k_S}$ are chosen randomly. An adversary $\mathcal{A}$ for symmetric encryption scheme is given access to l decryption oracles $\mathcal{D}_{K_i}()$. Then the adversary $\mathcal{A}$ runs in two stages Phase 1 and Phase 2 as follows.
Setup. l keys $K_1, K_2, ..., K_l \in \{0,1\}^{k_S}$ are chosen randomly.
Phase 1. An adversary $\mathcal{A}^{sym}$ can make arbitrary queries to any of decryption oracle $\mathcal{D}_{K_i}()$ for q_1 queries.
Challenge. The adversary $\mathcal{A}^{sym}$ outputs two messages (x_0, x_1). The challenger flips a fair binary coin, γ, encrypts x_γ under $K_1, K_2, ..., K_l$. The corresponding l targets $y_1, ..., y_l$ are given to $\mathcal{A}^{sym}$.
Phase 2. Phase 1 is repeated with the natural restriction of the l targets, i.e., the ith decryption oracle, $\mathcal{D}_{K_i}()$, is not queried with the ith target, y_i.
Guess. $\mathcal{A}^{sym}$ outputs a bit γ'. Formally, the advantage that $\mathcal{A}^{sym}$ has in breaking the symmetric encryption scheme is defined as

$$Adv_{\mathcal{A}^{sym}}^{sym} = 2|Pr[\gamma' = \gamma] - \frac{1}{2}|.$$

The quantity $Adv^{sym}(t, q)$ is defined to be the maximum of $Adv_{\mathcal{A}^{sym}}^{sym}$, where the maximum is taken over all adversaries $\mathcal{A}^{sym}$ running in time at most t and making a total of at most q queries to its decryption oracles.

B. The Security against the CCA

Consider, first of all, the security against the CCA from outsiders. The adversarial behavior described in Appendix A is modelled by playing the following game. For $0 \leq i \leq l$, we define the game $\mathcal{G}_i$ as follows. The phase 1 and the phase 2 of Appendix A remain unchanged. In the challenge step, the adversary generates (M_0, M_1) as usual. The targets generation for game $\mathcal{G}_i$ is constructed as follows:

choose a random bit $\gamma \in \{0,1\}$.
for $j = 1$ to i do
randomly choose $r_j \in Z_p$ and form $X_j = g^{r_j}, Y_j = \beta^{r_j}$;
randomly choose $K_j \in \{0,1\}^{k_S}$;
set $y_j = \mathcal{E}^{sym}(K_j, M_\gamma)$
set target $T_j = (y_j, X_j, Y_j)$
end for;
for $j = i+1$ to l do
generate T_j from M_γ using the system's public key g_{pub}, user U's public key g_U and sub_j as defined by Subject-DD scheme.
end for;
output targets $(T_1, ..., T_l)$

According to the guess, γ', output by the adversary $\mathcal{A}$, define the output of $\mathcal{G}_i$ to be: $\mathcal{A}(\mathcal{G}_i) = 1$, if $\gamma' = \gamma$; otherwise $\mathcal{A}(\mathcal{G}_i) = 0$. Formally the notation

$$Adv_{\mathcal{A}}^{Sub-DD} = 2|Pr[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2|$$

denotes the advantage that an adversary $\mathcal{A}$ has in breaking subject-DD scheme. $Adv^{Sub-DD}(t,q)$ is defined to be maximum of $Adv_{\mathcal{A}}^{Sub-DD}$ where the maximum is taken over all adversaries running in time t and making at most q queries to its decryption oracle.

Lemma 1 *Let $\mathcal{A}$ be an adversary for Subject-DD scheme which runs in time t and makes q queries to its decryption oracle. Then for all $0 \leq i \leq l-1$,*

$$|Pr[\mathcal{A}(\mathcal{G}_i) = 1] - Pr[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \leq \frac{p}{p-q} Adv^{\beta-dbdh}(t,q).$$

Proof: Our proof is a reduction. We show that if $\mathcal{A}$ can distinguish between $\mathcal{G}_i$ and $\mathcal{G}_{i+1}$, then it is possible to construct an algorithm $\mathcal{B}$ to solve the decisional β-BDH problem.

In order to prove it, we set the instance of decisional β-BDH problem as the $(i+1)$-th target, T_{i+1}, in the game described in Section 6. Then, if the instance is a valid tuple then the game is in fact the game $\mathcal{G}_i$; otherwise the game is corresponding to the game $\mathcal{G}_{i+1}$.

$\mathcal{A}$: the adversary of subject-DD scheme;
$\mathcal{B}$: an algorithm to solve the decisional β-BDH problem;
$\mathcal{C}$: the BDH problem oracle.
$\mathcal{D}_K$: the decryption oracle for the symmetric encryption scheme.

Setup.

- Set $u_i \in G_1, i = 0, ..., n$ and let $(g, g^a, g^b, g^c, sub \Rightarrow \beta, \beta^c, Z)$ be the corresponding instance of the decisional β-BDH problem.
- Algorithm $\mathcal{B}$ constructs an instance of subject-DD scheme in the following manner. For $j = 1, ..., l$ with $j \neq i+1$, randomly choose $sub_j \in \{0,1\}^n$. Set $sub_{i+1} = sub$, $g_{pub} = g^a$ and $g_U = g^b$.

- choose a non-collision injection function $H : G_2 \rightarrow \{0,1\}^{k_S}$.
- The public information $(g_{pub}; g_U; sub_1, sub_2, ..., sub_l; u_0, u_1, ..., u_n; H)$ is provided to $\mathcal{A}$.

Phase 1.

The adversary $\mathcal{A}$ asks q_1 decryption oracle queries on $(sub'_i, y'_i, g^{r_i}, \beta'^{r_i})$, $i = 1, ..., q_1$. On receipt of the queries,

- $\mathcal{B}$ queries the BDH problem oracle $\mathcal{C}$ on (g_{pub}, g_U, g^{r_i}) and obtains the answer $w_i = \hat{e}(g,g)^{abr_i}$ from $\mathcal{C}$,
- $\mathcal{B}$ computes $K_i = H(w_i)$, $x_i = \mathcal{D}_{K_i}(y'_i)$ for $i = 1, ..., q_1$ and then returns the answer x_i to $\mathcal{A}$.

Note that if $sub'_i = sub$, then with certain probability $r = c$ and due to the oracle restriction, algorithm $\mathcal{B}$ cannot invoke $w = \hat{e}(g,g)^{abc}$ on (g^a, g^b, g^c) and thus fail to answer the query. In this case, algorithm $\mathcal{B}$ outputs a random bit and exits.

Challenge.

(1) The adversary $\mathcal{A}$ submits two messages M_0, M_1.

(2) $\mathcal{B}$ flips a fair binary coin, γ, and generates the corresponding l targets of M_γ as follows:

- For $1 \leq j \leq i$, randomly generate $r_j \in Z_p$ and form $X_j = g^{r_j}$, $Y_j = \beta^{r_j}$. Randomly choose $K_1, ..., K_i \in \{0,1\}^{k_S}$ to compute $y_j = \mathcal{E}^{sym}_{K_j}(M_\gamma)$, respectively. Set $T_j = (sub_j, y_j, X_j, Y_j)$, for $1 \leq j \leq i$.
- For $j = i + 1$, set $X_j = g^c$ and encrypt M_γ with $K_{i+1} = H(Z)$ to obtain y_{i+1}. Set $T_{i+1} = (sub, y_{i+1}, g^c, \beta^c)$.
- For $i + 2 \leq j \leq l$, encrypt M_γ using subject-DD scheme encryption algorithm under subjects $sub_{i+2}, ..., sub_l$ to obtain the corresponding targets $T_{i+2}, ..., T_l$.

(3) All the targets $T_1, ..., T_l$ are given to the adversary $\mathcal{A}$.

Phase 2.

$\mathcal{A}$ asks q_2 queries to the decryption oracle for the plaintext of $(sub', y', X' = g^{r'}, Y' = \beta^{r'})$. There are two cases should be considered:

Case 1. $(sub', X') = (sub_j, X_j)$, for some $j \in \{1, ..., l\}$. It implies that $Y' = Y_j$ and $y' \neq y_j$ because of the restriction, then the simulator sets

$K'_j = K_j$, if $1 \leq j \leq i$;
$K'_j = Z$, if $j = i + 1$;
$K'_j = H(\hat{e}(g_{pub}, g_U)^{r_j})$, if $i + 2 \leq j \leq l$.

Case 2. $(sub', X') \neq (sub_j, X_j)$, for any $j \in \{1, ..., l\}$, then the simulator asks the BDH oracle $\mathcal{C}$ on (g_{pub}, X', g_U). Using the answer, $w_j = \mathcal{C}(g_{pub}, X', g_U) = \hat{e}(g,g)^{abr'_j}$, from $\mathcal{C}$, the simulator obtains $K'_j = H(w_j)$.

The simulator computes $x'_j = D_{K'_j}(y'_j)$ and gives the answers to $\mathcal{A}$. This completes the description of the simulation of adversary $\mathcal{A}$ by algorithm $\mathcal{B}$.

Guess.

$\mathcal{A}$ outputs his guess γ' for γ. According to $\mathcal{A}$'s answer, $\mathcal{B}$ outputs his guess as follows:

$$\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = \begin{cases} 1, & \gamma' = \gamma, \\ 0, & \gamma' \neq \gamma. \end{cases}$$

Let event $E_1 = \{Z = \hat{e}(g,g)^{abc}\}$ and event $E_2 = \{Z = random\}$. Suppose that E_1 occurs, then there are two possibilities,

- Fail occurs: In this case, $\mathcal{B}$ outputs a random bit. The probability that $\mathcal{B}$ fails for any particular query is the probability that for that query $r = c$ and this probability is $\frac{1}{p}$. Since $\mathcal{A}$ makes a total of $q = q_1 + q_2$ queries, then $Pr[Fail] = \frac{q}{p}$;
- Fail does not occur: In this case, $\mathcal{B}$ in fact runs $\mathcal{A}$ on game $\mathcal{G}_i$, it is because that $\mathcal{B}$'s challenge is a decisional β-BDH valid tuple and is set as the $(i+1)$-th target.

Therefore,

$$Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_1] = \frac{1}{2} \times \frac{q}{p} + Pr[\mathcal{A}(G_i) = 1] \times (1 - \frac{q}{p}).$$

By a similar argument, if E_2 occurs, in the case that Fail does not occur, $\mathcal{B}$ in fact runs $\mathcal{A}$ on game $\mathcal{G}_{i+1}$. So that we have

$$Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_2] = \frac{1}{2} \times \frac{q}{p} + Pr[\mathcal{A}(G_{i+1}) = 1] \times (1 - \frac{q}{p}).$$

Combining the above two equations we have,

$$\begin{aligned} & |Pr[\mathcal{A}(G_i) = 1] - Pr[\mathcal{A}(G_{i+1}) = 1]| \\ = & \frac{p}{p-q} \times |Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_1] \\ & \qquad - Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_2]|. \end{aligned}$$

Since adversary $\mathcal{A}$ for the subject-DD scheme makes at most q decryption queries, algorithm $\mathcal{B}$ for decisional β-BDH problem also makes at most q BDH oracle queries. Further, since adversary $\mathcal{A}$ runs in time t, algorithm $\mathcal{B}$ also runs in time t. Thus we have,

$$\begin{aligned} & |Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_1] - Pr[\mathcal{B}(g, g^a, g^b, g^c, \beta, \beta^c, Z) = 1 | E_2]| \\ \leq & Adv^{\beta - dbdh}(t, q). \end{aligned}$$

Therefore,

$$|Pr[\mathcal{A}(G_i) = 1] - Pr[\mathcal{A}(G_{i+1}) = 1]| \leq \frac{p}{p-q} Adv^{\beta - dbdh}(t, q).$$

The proof of Lemma 1 is completed. □

Lemma 2 *Let $\mathcal{A}'$ be an adversary for $\mathcal{G}_l$ running in time t and making q queries to its decryption oracle. Then*

$$2|Pr[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2| \leq Adv^{sym}(t, q).$$

Proof: Our proof is a reduction. In order to prove it, we construct an adversary $\mathcal{A}^{sym}$ for breaking symmetric encryption scheme in the sense described in Appendix A. Then show that the advantage of the adversary $\mathcal{A}$ has in game $\mathcal{G}_l$ is bound by the advantage of the adversary $\mathcal{A}^{sym}$ has in breaking symmetric encryption scheme.

The adversary $\mathcal{A}^{sym}$ is given l decryption oracles $\mathcal{D}_{K_1}, ..., \mathcal{D}_{K_l}$ corresponding to randomly chosen keys $K_1, ..., K_l \in \{0,1\}^{k_S}$.

Setup.

- $\mathcal{A}^{sym}$ chooses a random $a \in Z_p$ and forms $g_{pub} = g^a$.
- $\mathcal{A}^{sym}$ randomly chooses $sub_j \in \{0,1\}^n$, $j = 1, ..., l$. Sets $u_i \in G_1, i = 0, ..., n$.
- The public information $(g_{pub}; g_{_U}; sub_1, sub_2, ..., sub_l;\ u_0, u_1, ..., u_n)$ is provided to $\mathcal{A}'$.

Phase 1.

$\mathcal{A}'$ asks the decryption oracle query on (sub, y, X, Y). To answer this query, $\mathcal{A}^{sym}$ forms

$$K = H(\hat{e}(X, g_{_U})^a).$$

It then uses K to decrypt y and sends the answer to $\mathcal{A}'$.

Challenge.

$\mathcal{A}'$ outputs two messages M_0, and M_1. $\mathcal{A}^{sym}$ also outputs these two messages as his challenge. $\mathcal{A}^{sym}$'s challenger flips a fair coins to get γ and encrypts M_γ using $K_1, ..., K_l$ to get $y_1, ..., y_l$, respectively. Then $\mathcal{A}^{sym}$ is given the l targets $y_1, ..., y_l$ of M_γ. Next, $\mathcal{A}^{sym}$ randomly chooses $r_1, ..., r_l \in Z_p$. It forms l targets $T_1, ..., T_l$ by setting $T_j = (sub_j, y_j, X_j, Y_j)$. These targets are given to $\mathcal{A}'$.

Phase 2.

Let (sub, y, X, Y) be a decryption query from $\mathcal{A}'$. There are two cases to consider:

Case 1. $(sub, X) = (sub_j, X_j)$, for some $j \in \{1, ..., l\}$. It implies that $Y = Y_j$ and $y \neq y_j$ as otherwise, $(sub, y, X, Y) = T_j$. In this case, $\mathcal{A}^{sym}$ queries its jth decryption oracle $\mathcal{D}_{K_j}()$ on y_j and returns the answer to $\mathcal{A}'$.

Case 2. $(sub, X) \neq (sub_j, X_j)$, for any $j \in \{1, ..., l\}$. In this case, the query is dealt with as in the Phase 1.

Guess.

$\mathcal{A}^{sym}$ outputs whatever is produced by $\mathcal{A}'$. The above ensures a correct simulation by $\mathcal{A}^{sym}$ of $\mathcal{A}'$ on $\mathcal{G}_l$. Also, the number of decryption queries made by $\mathcal{A}^{sym}$ to all its decryption oracles is at most equal to q. Since $\mathcal{A}'$ runs in time t, the advantage of $\mathcal{A}'$ while running on $\mathcal{G}_l$ is

$$2|Pe[\mathcal{A}'(\mathcal{G}_l) = 1] - \frac{1}{2}| \leq Adv^{sym}(t, q).$$

This competes the proof. □

Theorem 1 *The security of our scheme is based on the decisional β-BDH assumption and the security of symmetric encryption scheme. In detail,*

$$Adv^{Subject-DD}(t,q) \leq \frac{2lp}{p-q} Adv^{\beta-dbdh}(t,q) + Adv^{sym}(t,q).$$

Proof: Theorem 1 is an immediate consequence of Lemma 1 and Lemma 2. In detail, let $\mathcal{A}$ be any adversary for Subject-DD scheme which runs in time at most t and makes at most q queries to the decryption oracle. Then from the definition, we have

$$Adv_{\mathcal{A}}^{Subject-DD} = 2|Pr[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2|.$$

From Lemma 1, we have for each $i \in \{0, 1, ..., l-1\}$,

$$|Pr[\mathcal{A}(\mathcal{G}_i) = 1] - Pr[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \leq \frac{p}{p-q} Adv^{\beta-dbdh}(t,q).$$

Now consider

$$\begin{aligned} & |Pr[\mathcal{A}(\mathcal{G}_0) = 1] - Pr[\mathcal{A}(\mathcal{G}_l) = 1]| \\ = & |\Sigma_{i=0}^{l-1} Pr[\mathcal{A}(\mathcal{G}_i) = 1] - Pr[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \\ \leq & \Sigma_{i=0}^{l-1} |Pr[\mathcal{A}(\mathcal{G}_i) = 1] - Pr[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \\ \leq & l\frac{p}{p-q} Adv^{\beta-dbdh}(t,q). \end{aligned}$$

It implies

$$\begin{aligned} & 2|Pr[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2| - 2|Pr[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2| \\ \leq & 2|(Pr[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2) - (Pr[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2)| \\ \leq & 2|Pr[\mathcal{A}(\mathcal{G}_0) = 1] - Pr[\mathcal{A}(\mathcal{G}_l) = 1]| \\ \leq & 2l\frac{p}{p-q} Adv^{\beta-dbdh}(t,q). \end{aligned}$$

Further, from Lemma 2, we have $2|Pr[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2| \leq Adv^{sym}(t,q)$. Therefore,

$$\begin{aligned} & 2|Pr[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2| \\ \leq & 2l\frac{p}{p-q} Adv^{\beta-dbdh}(t,q) + 2|Pr[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2)| \\ \leq & 2l\frac{p}{p-q} Adv^{\beta-dbdh}(t,q) + Adv^{sym}(t,q). \end{aligned}$$

Since $\mathcal{A}$ was chosen to be arbitrary adversary running in time at most t and making at most q queries to the decryption oracle, the above inequality holds for all such adversaries $\mathcal{A}$ and hence for any adversary which maximizes the advantage. The proof is completed. □

The above security proof is corresponding to the attack from outsiders, which is not enough for a subject-DD scheme. We have to consider the attack from

insiders, say a collusion of the proxies. It means that the adversary can obtain the keys of some of the proxies. Consider this case into the attack game described in Appendix A as follows: permit the adversary $\mathcal{A}$ to ask q_k queries to proxy key oracle in phase 1, and then challenger generates $l - q_k$ targets corresponding to the left $l - q_k$ subjects. By extending the above proof, we can also prove the security of our scheme against the CCA from insiders.

Similar to the proposed scheme introduced in Section 3.2, the security of the proposed variant scheme is also based on a variant BDH assumption. We will give the security proof in a full version of this paper.