Timed-Release Hierarchical Identity-Based Encryption

Toru Oshikiri
Graduate School of Engineering
Tokyo Denki University
Tokyo, Japan

Taiichi Saito
Tokyo Denki University
Tokyo, Japan

***Abstract*—We propose a notion of hierarchical identity-based encryption (HIBE) scheme with timed-release encryption (TRE) mechanism, timed-release hierarchical identity-based encryption (TRHIBE), and define its security models. We also show a generic construction of TRHIBE from HIBE and one-time signature, and discuss the security of the constructed scheme.**

***Keywords*—timed-release encryption, hierarchical identity-based encryption, one-time signature**

I. Introduction

Timed-release encryption (TRE) [1] [2] [3] [4] [5] is an encryption mechanism that allows a receiver to decrypt a ciphertext only after the time that a sender designates.

Timed-release identity-based encryption (TRIBE) [6] is an extension of TRE having a function of identity-based encryption (IBE). In TRIBE, even a legitimate receiver cannot decrypt a ciphertext using secret key until the time designated by the sender. A TRIBE system consists of a key generation center (KGC), a time server (TS), senders and receivers. A sender encrypts a message using an identity of a receiver and a time after which the ciphertext could be decrypted. The KGC generates a secret key corresponding to an identity of a receiver. The TS periodically broadcasts a time signal corresponding to the current time. The receiver decrypts the ciphertext using the secret key and the time signal corresponding to the time designated by the sender. TRIBE systems use identity of user as his/her public key. TRIBE has an advantage that it does not require linking public keys to identity such as PKI.

Timed-release hierarchical identity-based encryption (TRHIBE) is another extension of TRE having a function of hierarchical identity-based encryption (HIBE). In TRHIBE, even a legitimate receiver cannot decrypt a ciphertext using secret key until a time designated by a sender. A TRHIBE system consists of senders, multiple KGCs, a single TS, and receivers. The KGCs and users have a hierarchical structure in which each KGC generates a secret key corresponding to an identity of a child KGC or a child user. Therefore, the load of derivation of users secret keys can be distributed to multiple KGCs. A sender encrypts a message using an identity of a receiver and a time. The TS periodically broadcasts a time signal corresponding to the current time. The receiver decrypts the ciphertext using the secret key and the time signal corresponding to the time designated by the sender.

II. Related Works

In TRIBE, a user can decrypt a ciphertext only when the user has the receiver's secret key and the time signal generated by TS. Then, if the receiver does not have the time signal or the TS does not have the secret key, they cannot decrypt the ciphertext. In [6], two security models of TRIBE are defined. One is security against malicious receiver, $\mathsf{IND\text{-}ID\text{-}CCA_{CR}}$ security. The other is security against malicious TS, $\mathsf{IND\text{-}ID\text{-}CCA_{TS}}$ security. A generic construction of TRIBE that achieves the security is also shown in [6]. It is a combination of two IBE schemes and a one-time signature scheme, based on "Parallel Encryption" by Dodis-Katz [7], and the security is proved in the standard model.

III. Contribution

In this paper, we introduce timed-release hierarchical identity-based encryption (TRHIBE) and define two security models. One is security against malicious receiver, $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ security. The other is security against malicious TS, $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ security. We also present a generic construction of TRHIBE. It is a combination of two HIBE schemes and a one-time signature scheme, also based on "Parallel Encryption". We see that if the primitive HIBE schemes are $\mathsf{IND\text{-}hID\text{-}CCA}$ secure and the primitive one-time signature scheme is $\mathsf{OT\text{-}sEUF\text{-}CMA}$ secure, then the constructed TRHIBE scheme is $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ secure and $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ secure in the standard model.

IV. Preliminaries

In this section, we review hierarchical identity-based encryption (HIBE) and one-time signature, which we use later.

A. Hierarchical Identity-Based Encryption

In an HIBE scheme, the single KGC functionality of generating secret keys is divided into partial ones and they are delegated to multiple KGCs. If a KGC is assigned an identity vector, $\mathsf{ID}^{(k-1)} = (I_1, ..., I_{k-1})$, and given a secret key, $d_{\mathsf{ID}^{(k-1)}}$, corresponding to the identity vector, then it can generate a secret key, $d_{\mathsf{ID}^{(k)}}$, corresponding to an identity vector, $\mathsf{ID}^{(k)} = (I_1, ..., I_{k-1}, I_k)$. We may denote an identity by ID if we need not to specify its hierarchy depth.

Let λ be a security parameter and ℓ be a maximum depth of hierarchy. An *hierarchical identity-based encryption scheme* $\mathcal{HIBE}$ consists of five probabilistic polynomial-time algorithms $\mathcal{HIBE}$ =(HIBE.Setup, HIBE.Ext, HIBE.Del, HIBE.Enc, HIBE.Dec). The setup algorithm HIBE.Setup takes λ and ℓ as input, and outputs a public parameter $params$ and a master secret key msk. The extract algorithm HIBE.Ext takes $params$, msk, and an identity $\mathsf{ID}^{(k)} = (\mathsf{I}_1, \ldots, \mathsf{I}_k)$ as inputs, and outputs a decryption key $d_{\mathsf{ID}^{(k)}}$. The delegate algorithm HIBE.Del takes $params, \mathsf{ID}^{(k)}, d_{\mathsf{ID}^{(k)}}$ and an identity $\mathsf{ID}^{(k+1)}$ as inputs, and outputs a decryption key $d_{\mathsf{ID}^{(k+1)}}$. The encryption algorithm HIBE.Enc takes $params, \mathsf{ID}$, a message m as inputs, and outputs a ciphertext c. The decryption algorithm HIBE.Dec takes $params$, a ciphertext c and a decryption key d_{ID} as inputs, and outputs the plaintext m' or $\perp$. These algorithms are assumed to satisfy that if $(params, msk) = \mathsf{HIBE.Setup}(\lambda)$ and $d_{\mathsf{ID}} = \mathsf{HIBE.Ext}(params, msk, \mathsf{ID})$ or $d_{\mathsf{ID}^{(k)}} = \mathsf{HIBE.Del}(params, \mathsf{ID}^{(k-1)}, d_{\mathsf{ID}^{(k-1)}}, \mathsf{ID}^{(k)})$ for $k \leq n$, then $\mathsf{HIBE.Dec}(params, d_{\mathsf{ID}}, \mathsf{HIBE.Enc}(params, \mathsf{ID}, m)) = m$ for any m.

1) IND-hID-CCA *Security:* We review a standard security notion for HIBE: indistinguishability against adaptive hierarchical identity and chosen ciphertext attacks (IND-hID-CCA) security [8] [9]. We here describe the IND-hID-CCA security for HIBE scheme $\mathcal{HIBE}$ based on the following IND-hID-CCA game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup

$\mathcal{C}$ runs $(params, msk) \leftarrow \mathsf{HIBE.Setup}(\lambda, \ell)$. $\mathcal{C}$ sends $params$ to $\mathcal{A}$ and keeps msk secret.

Phase1

$\mathcal{A}$ can adaptively issue *extraction queries* ID and *decryption queries* (ID, c). $\mathcal{C}$ responds to an extraction query ID by running $d_{\mathsf{ID}_j} = \mathsf{HIBE.Ext}(params, msk, \mathsf{ID})$ and returning d_{ID} to $\mathcal{A}$. $\mathcal{C}$ responds to a decryption query (ID, c) by running $d_{\mathsf{ID}} = \mathsf{HIBE.Ext}(params, msk, \mathsf{ID})$ and $m' = \mathsf{HIBE.Dec}(d_{\mathsf{ID}}, c)$, and returning m' to $\mathcal{A}$.

Challenge

$\mathcal{A}$ sends two messages m_0, m_1 such that $|m_0| = |m_1|$, and an identity to be challenged ID^* to $\mathcal{C}$. The challenge identity ID^* must differ from any ID issued as extraction query in **Phase1**, and any its prefixes. $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$ and sends a challenge ciphertext $c^* = \mathsf{HIBE.Enc}(params, \mathsf{ID}^*, m_b)$ to $\mathcal{A}$.

Phase2

$\mathcal{A}$ can adaptively issue extraction queries ID and decryption queries (ID, c) in the same way as in **Phase1** except that the extraction queries ID must differ from the challenge identity ID^* and its prefixes, and decryption queries (ID, c) must differ from the pair (ID^*, c^*).

Guess

$\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b = b'$.

We define an advantage of $\mathcal{A}$ in the IND-hID-CCA game as $Adv^{\mathsf{IND\text{-}hID\text{-}CCA}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = |2\Pr[b = b'] - 1|$, in which the probability is taken over the random coins used by $\mathcal{C}$ and $\mathcal{A}$. We say that the HIBE scheme $\mathcal{HIBE}$ is IND-hID-CCA *secure* if, for any probabilistic polynomial-time adversary $\mathcal{A}$, the function $Adv^{\mathsf{IND\text{-}hID\text{-}CCA}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ is negligible in λ.

B. Signature

Let λ be a security parameter. $\mathcal{A}$ *signature* scheme $\mathcal{SIG}$ consists of three probabilistic polynomial-time algorithms $\mathcal{SIG} = (\mathsf{SigGen}, \mathsf{Sign}, \mathsf{Verify})$. The key generation algorithm SigGen takes λ as input, and outputs a signing key sk and a verification key vk. The signing algorithm Sign takes sk and a message m as inputs , and outputs a signature σ. The verification algorithm Verify takes vk, a message m, and a signature σ as inputs, and outputs `accept` or `reject`. These algorithms are assumed to satisfy that if $(sk, vk) = \mathsf{SigGen}(\lambda)$ then $\mathsf{Verify}(vk, m, \mathsf{Sign}(sk, m)) = \mathtt{accept}$ for any m.

1) OT-sEUF-CMA *Security:* We review a security notion for one-time signature scheme: one-time strong existential unforgeability against chosen message attacks (OT-sEUF-CMA) security [10]. We here describe the OT-sEUF-CMA security for signature scheme $\mathcal{SIG}$ based on the following OT-sEUF-CMA game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup

$\mathcal{C}$ runs the (*sk*, *vk*) $\leftarrow \mathsf{SigGen}(\lambda)$. $\mathcal{C}$ sends vk to $\mathcal{A}$ and keeps sk secret.

Query

$\mathcal{A}$ can issue a signing query m to $\mathcal{C}$ only once. $\mathcal{C}$ responds to the singing query m by running $\sigma = \mathsf{Sign}(vk, m)$ and returning σ to $\mathcal{A}$.

Forge

$\mathcal{A}$ outputs a pair (m^*, σ^*).

We define the advantage of $\mathcal{A}$ in the OT-sEUF-CMA game as $Adv^{\mathsf{OT\text{-}sEUF\text{-}CMA}}_{\mathcal{SIG},\mathcal{A}}(\lambda) = \Pr[\mathsf{Verify}(vk,\ m^*,\ \sigma^*) = \mathtt{accept} \wedge (m, \sigma) \neq (m^*, \sigma^*)]$, in which the probability is taken over the random coins used by $\mathcal{C}$ and $\mathcal{A}$. We say that the signature scheme $\mathcal{SIG}$ is OT-sEUF-CMA *secure* if, for any probabilistic polynomial-time adversary $\mathcal{A}$, the function $Adv^{\mathsf{OT\text{-}sEUF\text{-}CMA}}_{\mathcal{SIG},\mathcal{A}}(\lambda)$ is negligible in λ.

V. Timed-Release Hierarchical Identity-Based Encryption(TRHIBE)

In this section, we introduce timed-release hierarchical identity-based encryption(TRHIBE) scheme and define its security models.

A TRHIBE system consists of a single TS, multiple KGCs and multiple users connected through a communication network. The time server periodically broadcasts a time signal corresponding to the current time, and all users can receive the time signal. The single KGC functionality of generating secret keys is divided into partial ones and they are delegated to multiple KGCs. If a KGC is assigned an identity vector, $\mathsf{ID}^{(k-1)} = (I_1, ..., I_{k-1})$, and given a secret key, $d_{\mathsf{ID}^{(k-1)}}$, corresponding to the identity vector, then it can generate a secret key, $d_{\mathsf{ID}^{(k)}}$, corresponding to an identity vector, $\mathsf{ID}^{(k)} = (I_1, ..., I_{k-1}, I_k)$. We may denote an identity by ID if we need not to specify its hierarchy depth. A user (sender) encrypts a plaintext, designating another user (receiver) who can decrypt the ciphertext and a time only after which the ciphertext can be decrypted. The receiver can decrypt the ciphertext with the secret key that he/she has and the time signal that the time server broadcasts at the designated time.

Let λ be a security parameter and ℓ be a maximum depth of system. An *timed-release hierarchical identity-based encryption scheme* $\mathcal{TRHIBE}$ consists of seven probabilistic polynomial-time algorithms $\mathcal{TRHIBE}$=(TS_Setup, KGC_Setup, Release, Extract, Delegate, Encrypt, Decrypt). The time server's setup algorithm TS_Setup takes λ as input, and outputs a public key tpk and the corresponding secret key tsk. The key generation center's setup algorithm KGC_Setup takes λ and the depth ℓ as input, and outputs a public parameter $params$ and a master secret key msk. The release algorithm Release takes tpk, tsk and a time period T as inputs, and outputs a time signal d_T. The extract algorithm Extract takes $params$, msk, and an identity $\mathsf{ID}^{(k)} = (\mathsf{I}_1, \ldots, \mathsf{I}_k)$ as inputs, and outputs a decryption key $d_{\mathsf{ID}^{(k)}}$. The delegate algorithm Delegate takes $params$, $\mathsf{ID}^{(k)}$, $d_{\mathsf{ID}^{(k)}}$ and an identity $\mathsf{ID}^{(k+1)}$ as inputs, and outputs a decryption key $d_{\mathsf{ID}^{(k+1)}}$. The encryption algorithm Encrypt takes $tpk, params$, T, and ID, and a message m as inputs, and outputs a ciphertext c. The decryption algorithm Decrypt takes as inputs $tpk, params$, a ciphertext c', d_T, a user's secret key d_{ID}, and outputs the plaintext m' or $\perp$. These algorithms are assumed to satisfy that Decrypt(tpk, $params$, d_T, d_{ID}, Encrypt(tpk, $params$, T, ID, m)) = m holds for any m, if (tpk, tsk) = TS_Setup(λ), ($params$, msk) = KGC_Setup(λ, ℓ), s_T = TR.Release(tpk, tsk, T), and d_{ID} = HIBE.Ext($params$, msk, ID) hold, and that $d_{\mathsf{ID}^{(n)}}$ = HIBE.Ext($params$, msk, $\mathsf{ID}^{(n)}$) and $d_{\mathsf{ID}^{(k)}}$ = HIBE.Del($params$, $\mathsf{ID}^{(k-1)}$, $d_{\mathsf{ID}^{(k-1)}}$, $\mathsf{ID}^{(k)}$) for $k \leq n$ hold.

A. Security

We can consider security against malicious TS and security against malicious receiver.

1) $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ *Security.:* We introduce a security notion for TRHIBE: *indistinguishability against adaptive hierarchical identity and chosen ciphertext attacks by time-servers (*$\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$*) security*. This security ensures that a malicious time server, who has a secret key tsk, cannot obtain any information of message from ciphertext without decryption key d_{ID}. We here describe the $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ security for a TRHIBE scheme $\mathcal{TRHIBE}$ based on the following $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ game between a challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

Setup
: $\mathcal{C}$ runs (tpk, tsk) $\leftarrow$ TS_Setup(λ) and ($params$, msk) $\leftarrow$ KGC_Setup(λ, ℓ). $\mathcal{C}$ sends tpk, tsk and $params$ to $\mathcal{A}$ and keeps msk secret.

Phase1
: $\mathcal{A}$ can adaptively issue extraction queries ID and decryption queries (T, ID, c). $\mathcal{C}$ responds to an extraction query ID by running d_{ID} = Extract($params$, msk, ID) and returning d_{ID} to $\mathcal{A}$. $\mathcal{C}$ responds to a decryption query (T, ID, c) by running d_T = Release(tpk, tsk, T), d_{ID} = Extract($params$, msk, ID) and c = Decrypt(d_T, d_{ID}, c) , and returning c to $\mathcal{A}$.

Challenge
: $\mathcal{A}$ sends two messages m_0, m_1 such that $|m_0| = |m_1|$, a time period T^* and an identity to be challenged ID^* to $\mathcal{C}$. The challenge identity ID^* must differ from any ID issued as extraction queries in **Phase1** and any its prefixes. $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$ and sends a challenge ciphertext $c^* = \mathsf{Encrypt}(tpk, params, T^*, \mathsf{ID}^*, m_b)$ to $\mathcal{A}$.

Phase2
: $\mathcal{A}$ can adaptively issue extraction queries ID and decryption queries (T, ID, c) in the same way as **Phase1** except that the extraction queries ID must differ from the challenge identity ID^* and its prefixes, and the decryption queries (T, ID, c) must differ from the tuple $(T^*, \mathsf{ID}^*, c^*)$.

Guess
: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b = b'$.

We define an advantage of $\mathcal{A}$ in the $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ game as $Adv^{\mathsf{IND\text{-}hID\text{-}CCA_{TS}}}_{\mathcal{TRHIBE},\mathcal{A}}(\lambda) = |2\Pr[b = b'] - 1|$, in which the probability is taken over the random coins used by $\mathcal{C}$ and $\mathcal{A}$. We say that the TRIBE scheme $\mathcal{TRHIBE}$ is $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ *secure* if, for any probabilistic polynomial-time adversary $\mathcal{A}$, the function $Adv^{\mathsf{IND\text{-}hID\text{-}CCA_{TS}}}_{\mathcal{TRHIBE},\mathcal{A}}(\lambda)$ is negligible in λ.

2) $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ *Security.:* We introduce another security notion for TRIBE: *indistinguishability against adaptive hierarchical identity and chosen ciphertext attacks by curious receiver (*$\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$*) security*. This security ensures that a receiver who has a decryption key d_{ID} cannot obtain any information of message from ciphertext without time signal d_T. We here describe the $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ security for a TRIBE scheme $\mathcal{TRHIBE}$ based on the following $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup
: $\mathcal{C}$ runs (tpk, tsk) $\leftarrow$ TS_Setup(λ) and ($params$, msk) $\leftarrow$ KGC_Setup(λ, ℓ). $\mathcal{C}$ sends $params$, msk and tpk to $\mathcal{A}$ and keeps tsk secret.

Phase1
: $\mathcal{A}$ can adaptively issue release queries T and decryption queries (T, ID, c). $\mathcal{C}$ responds to a release query T by running d_T = Release(tpk, tsk, T) and returning d_T to $\mathcal{A}$. $\mathcal{C}$ responds to a decryption query (T, ID, c) by running d_T = Release(tpk, tsk, T), $d_{\mathsf{ID}} = \mathsf{Extract}(params msk, \mathsf{ID})$ and $c = \mathsf{Decrypt}(d_T, d_{\mathsf{ID}}, c)$, and returning c to $\mathcal{A}$.

Challenge
: $\mathcal{A}$ sends two messages m_0, m_1 such that $|m_0| = |m_1|$, a time period T^* and an identity ID^* to be challenged to $\mathcal{C}$. The challenge time period T^* must differ from any T issued as release queries in Phase1. $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$ and sends a challenge ciphertext c^* = Encrypt(tpk, $params$, T^*, ID^*, m_b) to $\mathcal{A}$.

Phase2
: $\mathcal{A}$ can adaptively issue release queries T and decryption queries (T, ID, c) in the same way as **Phase1** except that the release query T must differ from the challenge time period T^*, and the decryption queries (T, ID, c) must differ from the tuple $(T^*, \mathsf{ID}^*, c^*)$.

Guess
: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b = b'$.

We define an advantage of $\mathcal{A}$ in the $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ game as $Adv^{\mathsf{IND\text{-}hID\text{-}CCA_{CR}}}_{\mathcal{TRHIBE},\mathcal{A}}(\lambda) = |2\Pr[b = b'] - 1|$, in which the probability is taken over the random coins used by $\mathcal{C}$ and $\mathcal{A}$. We say that the TRIBE scheme $\mathcal{TRHIBE}$ is $\mathsf{IND\text{-}hID\text{-}CCA_{CR}}$ *secure* if, for any probabilistic polynomial-time adversary $\mathcal{A}$, the function $Adv^{\mathsf{IND\text{-}hID\text{-}CCA_{CR}}}_{\mathcal{TRHIBE},\mathcal{A}}(\lambda)$ is negligible in λ.

VI. Construction of TRHIBE

Here we present a generic construction of TRHIBE scheme from two HIBE schemes, and a one-time signature scheme.

A. Construction

Let Π = (HIBE.Setup, HIBE.Ext, HIBE.Del, HIBE.Enc, HIBE.Dec) and Π' = (HIBE′.Setup, HIBE′.Ext, HIBE′.Del, HIBE′.Enc, HIBE′.Dec) be hierarchical identity-based encryption schemes, and Σ = (SigGen, Sign, Verify) be a one-time signature scheme.

A TRHIBE scheme Γ = (TS_Setup, KGC_Setup, Release, Extract, Encrypt, Decrypt) is constructed as follows.

Time server setup TS_Setup(λ)**:**
Step 1: Run HIBE.Setup(λ, 1) to generate $(params, msk)$.
Step 2: Set $tpk = params$ and $tsk = msk$.
Step 3: Return (tpk, tsk).

Key generation center setup KGC_Setup(λ, ℓ)**:**
Step 1: Run HIBE′.Setup(λ, ℓ) to generate $(params, msk)$.
Step 2: Return $(params, msk)$.

Release Release(tpk, tsk, T)**:**
Step 1: Run HIBE.Ext(tpk, tsk, T) to obtain d_T.
Step 2: Return d_T.

Extraction Extract($params, msk$, ID)**:**
Step 1: Run HIBE′.Ext($params, msk$, ID) to obtain d_{ID_j}.
Step 2: Return d_{ID_j}.

Delegate($params, \mathsf{ID}^{(k)}, d_{\mathsf{ID}^{(k)}}, \mathsf{ID}^{(k+1)}$)**:**
Step 1: Run HIBE′.Del($params, \mathsf{ID}^{(k)}, d_{\mathsf{ID}^{(k)}}, \mathsf{ID}^{(k+1)}$) to obtain $d_{\mathsf{ID}^{(k+1)}}$.
Step 2: Return $d_{\mathsf{ID}^{(k+1)}}$.

Encryption Encrypt($tpk, params, m, T$, ID)**:**
Step 1: Run SigGen(λ) to generate (sk, vk).
Step 2: Randomly choose $s_1 \in \{0,1\}^{|m|}$.
Step 3: Compute $s_2 = m \oplus s_1$.
Step 4: Compute c_1 = HIBE.Enc($tpk, s_1||vk, T$).
Step 5: Compute c_2 = HIBE′.Enc($params, s_2||vk$, ID).
Step 6: Compute σ = Sign($sk, c_1||c_2||T||$ID).
Step 7: Set $c = (c_1, c_2, T, \mathsf{ID}, vk, \sigma)$.
Step 8: Return c.

Decryption Decrypt($tpk, params, c, d_T, d_{\mathsf{ID}}$)**:**
Step 1: Parse c as $c = (c_1, c_2, T, \mathsf{ID}, vk, \sigma)$.
Step 2: If Verify($vk, c_1||c_2||T||$ID$, \sigma$)= `reject` then return $\perp$ and stop.
Step 3: Compute $s_1||vk'$ = HIBE.Dec(tpk, c_1, d_T).
Step 4: Compute $s_2||vk''$ = HIBE′.Dec($params, c_2, d_{\mathsf{ID}}$).
Step 5: If $vk = vk' = vk''$ then return $m = s_1 \oplus s_2$, else return $\perp$.

B. Security of TRHIBE.

1) $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ secure:

Theorem 1: If Π' is an IND-hID-CCA secure hierarchical identity-based encryption scheme and Σ is a OT-sEUF-CMA secure one-time signature scheme, then Γ is an $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ secure timed-release hierarchical identity-based encryption scheme.

Proof(Theorem 1) Suppose $\mathcal{A}$ is an adversary that breaks the $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ security of Γ. We construct a simulator $\mathcal{B}$ which breaks the IND-hID-CCA security of the HIBE scheme Π' using $\mathcal{A}$. Say a ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk, \sigma)$ is *valid* if Verify$(vk, c_1||c_2||T||\mathsf{ID}, \sigma)$ = `accept`. Let $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ be the challenge ciphertext. Let `Forge` denote the event that $\mathcal{A}$ submits a valid ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk^*, \sigma)$ as a decryption query to $\mathcal{C}$ in the **Phase2**, and `Succ` denote the event that $\mathcal{B}$ wins the IND-hID-CCA game. We prove the following claims.

Claim 1: $\Pr[\mathtt{Forge}]$ is negligible.

Claim 2: $\Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] = Adv^{\mathsf{IND\text{-}hID\text{-}CCA_{TS}}}_{\Gamma,\mathcal{A}} + \frac{1}{2}$

Proof(Claim 1) We assume `Forge` occurs. Then, we construct a forger $\mathcal{F}$ who breaks OT-sEUF-CMA security of the one-time signature scheme Σ, from $\mathcal{A}$. The description of $\mathcal{F}$ is as follows.

Setup
$\mathcal{F}$ receives vk^* from $\mathcal{C}$. Then $\mathcal{F}$ runs (tpk, tsk) $\leftarrow$ TS_Setup(λ) and ($params$, msk) $\leftarrow$ KGC_Setup(λ, ℓ). $\mathcal{F}$ sends tpk, tsk and $params$ to $\mathcal{A}$ and keeps msk.

Query
$\mathcal{F}$ can respond to extract queries and decryption queries of $\mathcal{A}$ since $\mathcal{F}$ has tsk and msk. If $\mathcal{A}$ happens to issue a valid ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk^*, \sigma)$ as decryption query to $\mathcal{F}$ before **Challenge** in the $\mathsf{IND\text{-}hID\text{-}CCA_{TS}}$ game, then $\mathcal{F}$ simply outputs $(c_1||c_2||T||\mathsf{ID}, \sigma)$ as forgery and stops.

Challenge
If $\mathcal{A}$ outputs $(m_0, m_1, T^*, \mathsf{ID}^*)$ as challenge, $\mathcal{F}$ randomly chooses $s_1 \in \{0,1\}^{|m|}$ and $b \in \{0,1\}$, and computes $s_2 = m_b \oplus s_1$. Then $\mathcal{F}$ computes c_1^* = HIBE.Enc($tpk, s_1||vk^*, T^*$) and c_2^* = HIBE′.Enc($params, s_2||vk^*$, ID*), then issues $m^* = (c_1||c_2||T^*||\mathsf{ID}^*)$ as signing query to $\mathcal{C}$ and obtains σ^*. Finally $\mathcal{F}$ returns $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ as the challenge ciphertext to $\mathcal{A}$.

Forge
If $\mathcal{A}$ issues a valid ciphertxt c = (c_1, c_2, T, ID, vk^*, σ) as decryption query, then $\mathcal{F}$ outputs $(c_1||c_2||T||\mathsf{ID}^*, \sigma)$ as forgery.

$\mathcal{F}$ can forge the signature if $\mathcal{A}$ issues a decryption query that causes the event `Forge`. It, however, contradicts that Σ is OT-sEUF-CMA secure. Thus, $\Pr[\mathtt{Forge}]$ is negligible. □

Proof(Claim 2) We construct an adversary $\mathcal{B}$ who breaks IND-hID-CCA security of the HIBE scheme Π' using $\mathcal{A}$. The description of $\mathcal{B}$ is as follows.

Setup

$\mathcal{B}$ receives $params$ from $\mathcal{C}$. Then $\mathcal{B}$ runs (*tpk*, *tsk*) $\leftarrow$ TS_Setup(λ) and sends tpk, tsk and $params$ to $\mathcal{A}$.

Phase1

$\mathcal{B}$ responds to $\mathcal{A}$'s extraction query ID by issuing ID as $\mathcal{B}$'s extraction query to $\mathcal{C}$ and obtaining d_{ID} from $\mathcal{C}$ and returning d_{ID} to $\mathcal{A}$. $\mathcal{B}$ responds to $\mathcal{A}$'s decryption query c as follows. If $\mathsf{Verify}(vk, c_1||c_2||T||\mathsf{ID}, \sigma) = \mathtt{reject}$, then $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$. Otherwise $\mathcal{B}$ runs $s_1||vk' \leftarrow$ HIBE.Dec(c_1, d_T) and issues decryption query (c_2, ID) to $\mathcal{C}$ and obtains $s_2||vk''$. $\mathcal{B}$ returns $m = s_1 \oplus s_2$ to $\mathcal{A}$ if $vk = vk' = vk''$, and otherwise $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$.

Challenge

If $\mathcal{A}$ outputs $(m_0,\ m_1,\ T^*, \mathsf{ID}^*)$ as challenge, $\mathcal{B}$ runs $(sk^*,\ vk^*) \leftarrow$ SigGen(λ) and randomly chooses $s_1 \in \{0,1\}^{|m|}$ and runs $c_1^* =$ HIBE.Enc$(tpk, r||vk^*, T^*)$. Then $\mathcal{B}$ computes $M_0 = (m_0 \oplus r||vk^*)$ and $M_1 = (m_1 \oplus r||vk^*)$, and issues $(M_0, M_1, \mathsf{ID}^*)$ as $\mathcal{B}$'s challenge to $\mathcal{C}$ and obtains cyphertext c_2^*. $\mathcal{B}$ runs $\sigma^* = \mathsf{Sign}(sk^*, c_1^*||c_2^*||T^*||\mathsf{ID}^*)$ and returns $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ as challenge ciphertext to $\mathcal{A}$.

Phase2

$\mathcal{B}$ responds to $\mathcal{A}$'s extraction query ID in the same way as in **Phase1**. $\mathcal{B}$ responds to $\mathcal{A}$'s decryption query as follows. The followings are done in a sequential way.

Step1

If $\mathsf{Verify}(vk, c_1||c_2||T||\mathsf{ID}||, \sigma) = \mathtt{reject}$, then $\mathcal{B}$ returns $\perp$ and skips **step2**$\sim$**4**.

Step2

If $vk = vk^*$, then $\mathcal{B}$ stops the simulation and outputs a random bit b'.

Step3

If $(c_2, \mathsf{ID}) = (c_2^*, \mathsf{ID}^*)$, then $\mathcal{B}$ returns $\perp$ and skips **step4**.

Step4

$\mathcal{B}$ responds in the same way as in **Phase1**.

Guess If $\mathcal{A}$ outputs a bit, then $\mathcal{B}$ outputs a same bit as its guess.

We examine the $\mathcal{B}$'s simulation of the response to decryption queries in **Phase2**. In the case of $\mathsf{Verify} = \mathtt{reject}$ in **Step1**, $\mathcal{B}$ returns $\perp$ in the same way as in our decryption algorithm, and then it perfectly simulates the challenger in IND-hID-CCA$_{\mathsf{TS}}$ game. In the case of $vk = vk^*$ in **Step2**, the event $\mathtt{Forge}$ occurs. In the case of $(c_2, \mathsf{ID}) = (c_2^*, \mathsf{ID}^*)$ in **Step3**, since c_2 equals to c_2^*, the decryption of c_2 is $M_0 = (m_0 \oplus r||vk^*)$ or $M_1 = (m_1 \oplus r||vk^*)$. However, since $vk \neq vk^*$, the decryption of c is $\perp$, and then $\mathcal{B}$ simulates perfectly. In the case of $(c_2, \mathsf{ID}) \neq (c_2^*, \mathsf{ID}^*)$, $\mathcal{B}$ can issue the valid decryption query (c_2, ID) to $\mathcal{C}$.

If the event $\mathtt{Forge}$ does not occurs, $\mathcal{B}$ perfectly simulates the challengers in the IND-hID-CCA$_{\mathsf{TS}}$ game and wins the IND-hID-CCA game with the same probability that $\mathcal{A}$ wins the IND-hID-CCA$_{\mathsf{TS}}$ game, i.e., $\Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] = Adv_{\Gamma,\mathcal{A}}^{\text{IND-hID-CCA}_{\text{TS}}} + \frac{1}{2}$. □

We see that

$$\begin{aligned}
\Pr[\mathtt{Succ}] &\geq \Pr[\mathtt{Succ} \wedge \overline{\mathtt{Forge}}] \\
&= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot \Pr[\overline{\mathtt{Forge}}] \\
&= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot (1 - \Pr[\mathtt{Forge}]) \\
&= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] - \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot \Pr[\mathtt{Forge}] \\
&\geq \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] - \Pr[\mathtt{Forge}],
\end{aligned}$$

then, from **Claim 2**, we have that

$$\Pr[\mathtt{Succ}] \geq Adv_{\Gamma,\mathcal{A}}^{\text{IND-hID-CCA}_{\text{TS}}} + \frac{1}{2} - \Pr[\mathtt{Forge}].$$

If $Adv_{\Gamma,\mathcal{A}}^{\text{IND-hID-CCA}_{\text{TS}}}$ is not negligible, $Adv_{\Pi,\mathcal{B}}^{\text{IND-hID-CCA}} = |\Pr[\mathtt{Succ}] - \frac{1}{2}|$ is not negligible from **Claim 1**, and it contradicts our assumption. This completes the proof of **Theorem 1**. □

2) IND-hID-CCA$_{\mathsf{CR}}$ secure:

Theorem 2: If Π is an IND-hID-CCA secure hierarchical identity-based encryption scheme and Σ is a OT-sEUF-CMA secure one-time signature scheme, then Γ is an IND-hID-CCA$_{\mathsf{CR}}$ secure timed-release hierarchical identity-based encryption scheme.

Proof(Theorem 2) Suppose $\mathcal{A}$ is an adversary that breaks the IND-hID-CCA$_{\mathsf{TS}}$ security of Γ. We construct a simulator $\mathcal{B}$ which breaks the IND-hID-CCA security of the HIBE scheme Π using $\mathcal{A}$. Say a ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk, \sigma)$ is $valid$ if $\mathsf{Verify}(vk, c_1||c_2||T||\mathsf{ID}, \sigma) = \mathtt{accept}$. Let $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ be the challenge ciphertext. Let $\mathtt{Forge}$ denote the event that $\mathcal{A}$ submits a valid ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk^*, \sigma)$ as a decryption query to $\mathcal{C}$ in the $\mathbf{Phase2}$, and $\mathtt{Succ}$ denote the event that $\mathcal{B}$ wins the IND-hID-CCA game. We prove the following claims.

Claim 3: $\Pr[\mathtt{Forge}]$ is negligible.

Claim 4: $\Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] = Adv_{\Gamma,\mathcal{A}}^{\text{IND-hID-CCA}_{\text{CR}}} + \frac{1}{2}$

Proof(Claim 3) We assume $\mathtt{Forge}$ occurs. Then, We construct a forger $\mathcal{F}$ who breaks OT-sEUF-CMA security of the one-time signature scheme Σ, from $\mathcal{A}$. The description of $\mathcal{F}$ is as follows.

Setup

$\mathcal{F}$ receives vk^* from $\mathcal{C}$. Then $\mathcal{F}$ runs (*tpk*, *tsk*) $\leftarrow$ TS_Setup(λ) and (*params*, *msk*) $\leftarrow$ KGC_Setup(λ, ℓ). $\mathcal{F}$ sends $params, msk$ and tpk to $\mathcal{A}$ and keeps tsk.

Query

$\mathcal{F}$ can respond to extract queries and decryption queries of $\mathcal{A}$ since $\mathcal{F}$ has tsk and msk. If $\mathcal{A}$ happens to issue a valid ciphertext $c = (c_1, c_2, T, \mathsf{ID}, vk^*, \sigma)$ as decryption query to $\mathcal{F}$ before **Challenge** in the IND-hID-CCA$_{\mathsf{TS}}$

game, then $\mathcal{F}$ simply outputs $(c_1||c_2||T||\mathsf{ID}, \sigma)$ as forgery and stops.

Challenge

If $\mathcal{A}$ outputs $(m_0, m_1, T^*, \mathsf{ID}^*)$ as challenge , $\mathcal{F}$ randomly chooses $s_1 \in \{0,1\}^{|m|}$ and $b \in \{0,1\}$, and computes $s_2 = m_b \oplus s_1$. Then $\mathcal{F}$ computes c_1^* = HIBE.Enc($tpk, s_1||vk^*, T^*$) and c_2^* = HIBE'.Enc($params, s_2||vk^*, \mathsf{ID}^*$), then issues $m^* = (c_1||c_2||T^*||\mathsf{ID}^*)$ as signing query to $\mathcal{C}$ and obtains σ^*. Finally $\mathcal{F}$ returns $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ as the challenge ciphertext to $\mathcal{A}$.

Forge

If $\mathcal{A}$ issues a valid ciphertxt $c = (c_1, c_2, T, \mathsf{ID}, vk^*, \sigma)$ as decryption query, then $\mathcal{F}$ outputs $(c_1||c_2||T||\mathsf{ID}^*, \sigma)$ as forgery.

$\mathcal{F}$ can forge the signature if $\mathcal{A}$ issues a decryption query that causes the event $\mathtt{Forge}$. It, however, contradicts that Σ is OT-sEUF-CMA secure. Thus, $\Pr[\mathtt{Forge}]$ is negligible. □

Proof(Claim 4) We construct an adversary $\mathcal{B}$ who breaks IND-hID-CCA security of the HIBE scheme Π using $\mathcal{A}$. The description of $\mathcal{B}$ is as follows.

Setup

$\mathcal{B}$ receives $params$ from $\mathcal{C}$. We call this $params$ tpk. Then $\mathcal{B}$ runs (*params*, *msk*) $\leftarrow$ KGC_Setup(λ, ℓ) and sends $params, msk$ and tpk to $\mathcal{A}$.

Phase1

$\mathcal{B}$ responds to $\mathcal{A}$'s release query T by issuing T as $\mathcal{B}$'s extraction query to $\mathcal{C}$ and obtaining d_T from $\mathcal{C}$ and returning d_T to $\mathcal{A}$. $\mathcal{B}$ responds to $\mathcal{A}$'s decryption query c as follows. If $\mathsf{Verify}(vk, c_1||c_2||T||\mathsf{ID}, \sigma) = \mathtt{reject}$, then $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$. Otherwise $\mathcal{B}$ runs $s_2||vk' \leftarrow$ HIBE.Dec(c_2, d_{ID}) and issues decryption query (c_1, T) to $\mathcal{C}$ and obtains $s_1||vk''$. $\mathcal{B}$ returns $m = s_1 \oplus s_2$ to $\mathcal{A}$ if $vk = vk' = vk''$, and otherwise $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$.

Challenge

If $\mathcal{A}$ outputs $(m_0, m_1, T^*, \mathsf{ID}^*)$ as challenge, $\mathcal{B}$ runs $(sk^*, vk^*) \leftarrow$ SigGen(λ) and randomly chooses $s_1 \in \{0,1\}^{|m|}$ and runs c_1^* = HIBE.Enc($params, r||vk^*, \mathsf{ID}^*$). Then $\mathcal{B}$ computes $M_0 = (m_0 \oplus r||vk^*)$ and $M_1 = (m_1 \oplus r||vk^*)$, and issues (M_0, M_1, T^*) as $\mathcal{B}$'s challenge to $\mathcal{C}$ and obtains cyphertext c_2^*. $\mathcal{B}$ runs $\sigma^* = \mathsf{Sign}(sk^*, c_1^*||c_2^*||T^*||\mathsf{ID}^*)$ and returns $c^* = (c_1^*, c_2^*, T^*, \mathsf{ID}^*, vk^*, \sigma^*)$ as challenge ciphertext to $\mathcal{A}$.

Phase2

$\mathcal{B}$ responds to $\mathcal{A}$'s extraction query T in the same way as in **Phase1**. $\mathcal{B}$ responds to $\mathcal{A}$'s decryption query as follows. The followings are done in a sequential way.

Step1

If $\mathsf{Verify}(vk, c_1||c_2||T||\mathsf{ID}||, \sigma) = \mathtt{reject}$, then $\mathcal{B}$ returns $\perp$ and skips **step2∼4**.

Step2

If $vk = vk^*$, then $\mathcal{B}$ stops the simulation and outputs a random bit b'.

Step3

If $(c_1, T) = (c_1^*, T^*)$, then $\mathcal{B}$ returns $\perp$ and skips **step4**.

Step4

$\mathcal{B}$ responds in the same way as in **Phase1**.

Guess

If $\mathcal{A}$ outputs a bit, then $\mathcal{B}$ outputs a same bit as its guess.

We examine the $\mathcal{B}$'s simulation of the response to decryption queries in **Phase2**. In the case of $\mathsf{Verify} = \mathtt{reject}$ in **Step1**, $\mathcal{B}$ returns $\perp$ in the same way as in our decryption algorithm, and then it perfectly simulates the challenger in IND-hID-CCA$_{\mathsf{TS}}$ game. In the case of $vk = vk^*$ in **Step2**, the event $\mathtt{Forge}$ occurs. In the case of $(c_1, T) = (c_1^*, T^*)$ in **Step3**, since c_1 equals to c_1^*, the decryption of c_1 is $M_0 = (m_0 \oplus r||vk^*)$ or $M_1 = (m_1 \oplus r||vk^*)$. However, since $vk \neq vk^*$, the decryption of c is $\perp$, and then $\mathcal{B}$ simulates perfectly. In the case of $(c_1, T) \neq (c_1^*, T^*)$, $\mathcal{B}$ can issue the valid decryption query (c_1, T) to $\mathcal{C}$.

If the event $\mathtt{Forge}$ does not occurs, $\mathcal{B}$ perfectly simulates the challengers in the IND-hID-CCA$_{\mathsf{CR}}$ game and wins the IND-hID-CCA game with the same probability that $\mathcal{A}$ wins the IND-hID-CCA$_{\mathsf{CR}}$ game, i.e., $\Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] = Adv_{\Gamma,\mathcal{A}}^{\mathsf{IND\text{-}hID\text{-}CCA_{CR}}} + \frac{1}{2}$. □

We see that

$$\begin{aligned}\Pr[\mathtt{Succ}] &\geq \Pr[\mathtt{Succ} \wedge \overline{\mathtt{Forge}}]\\ &= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot \Pr[\overline{\mathtt{Forge}}]\\ &= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot (1 - \Pr[\mathtt{Forge}])\\ &= \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] - \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] \cdot \Pr[\mathtt{Forge}]\\ &\geq \Pr[\mathtt{Succ}|\overline{\mathtt{Forge}}] - \Pr[\mathtt{Forge}],\end{aligned}$$

then, from **Claim 3**, we have that

$$\Pr[\mathtt{Succ}] \geq Adv_{\Gamma,\mathcal{A}}^{\mathsf{IND\text{-}hID\text{-}CCA_{CR}}} + \frac{1}{2} - \Pr[\mathtt{Forge}].$$

If $Adv_{\Gamma,\mathcal{A}}^{\mathsf{IND\text{-}hID\text{-}CCA_{CR}}}$ is not negligible, $Adv_{\Pi,\mathcal{B}}^{\mathsf{IND\text{-}hID\text{-}CCA}} = |\Pr[\mathtt{Succ}] - \frac{1}{2}|$ is not negligible from **Claim 4**, and it contradicts our assumption. This completes the proof of **Theorem 2**. □

VII. Conclusion

In this paper, we introduced a notion of TRHIBE and defined IND-hID-CCA$_{\mathsf{CR}}$ security and IND-hID-CCA$_{\mathsf{TS}}$ security. Moreover, we showed a generic construction of TRHIBE in which a constructed scheme achieves those security if the primitive HIBE schemes are IND-hID-CCA secure and the primitive one-time signature scheme is OT-sEUF-CMA secure.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 26330160.

REFERENCES

[1] T. May, “Timed-release crypto,” Manuscript, February 1993.

[2] A. C.-F. Chan and I. F. Blake, “Scalable, server-passive, user-anonymous timed release cryptography,” in *ICDCS 2005*. IEEE Computer Society, 2005, pp. 504–513.

[3] J. H. Cheon, N. Hopper, Y. Kim, and I. Osipkov, “Timed-release and key-insulated public key encryption,” in *FC 2006*, ser. Lecture Notes in Computer Science, G. Di Crescenzo and A. Rubin, Eds., vol. 4107. Springer-Verlag, 2006, pp. 191–205.

[4] ——, “Provably secure timed-release public key encryption,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 11, no. 2, p. Article 4, 2008.

[5] J. Cathalo, B. Libert, and J.-J. Quisquater, “Efficient and non-interactive timed-release encryption,” in *ICICS 2005*, ser. Lecture Notes in Computer Science, S. Qing, W. Mao, J. Lopez, and G. Wang, Eds., vol. 3783. Springer-Verlag, 2005, pp. 291–303.

[6] T. Oshikiri and T. Saito, “Timed-release identity-based encryption,” *IPSJ Journal*, vol. 55, no. 9, pp. 1964–1970, sep 2014.

[7] Y. Dodis and J. Katz, “Chosen-ciphertext security of multiple encryption,” in *TCC 2005*, ser. Lecture Notes in Computer Science, J. Kilian, Ed., vol. 3378. Springer-Verlag, 2005, pp. 188–209.

[8] D. Boneh, X. Boyen, and E.-J. Goh, “Hierarchical identity based encryption with constant size ciphertext,” in *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, ser. EUROCRYPT’05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 440–456.

[9] A. Lewko and B. Waters, “New techniques for dual system encryption and fully secure hibe with short ciphertexts,” in *Theory of Cryptography*, ser. Lecture Notes in Computer Science, D. Micciancio, Ed. Springer Berlin Heidelberg, 2010, vol. 5978, pp. 455–479.

[10] R. C. Merkle, “A digital signature based on a conventional encryption function,” in *A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, ser. CRYPTO ’87. London, UK, UK: Springer-Verlag, 1988, pp. 369–378. [Online]. Available: http://dl.acm.org/citation.cfm?id=646752.704751