2005－CSEC－29 (6)
2005／5／19

On Public-key Broadcast Encryption

ATSUKO MIYAJI†

Broadcast Encryption (BE) gives a way to distribute digital contents to subscribers by using an open broadcast channel, in which a set of privileged users may be changed by each digital content. We focus on the public-key BE for stateless recivers, in which each user is given a fixed set of keys beforehand and keeps using it to decrypt broadcasted contents through the lifetime of the system; any contents provider including the center can broadcast contents safely by using the same setting and instruments; and the security of system is no longer compromised by exposures of secret keys stored by providers. The stateless receivers are natural setting for application such as DVD decorder, etc. An efficient method, called subset difference broadcast encyrption(SDBE), for realizing this setting was proposed. The public-key setting is also convenient and flexible for an open network although many recent works of BE for stateless receivers, including SDBE, are based on a symmetric key encryption. Recently, the public-key SDBE is proposed by using a hierarchical ID-based encryption (HIBE). HIBE can translate the symmetric-key SDBE to the public-key version faithfully and, thus, the transmission rate or the user secret key size of the public-key is at most that of the symmetric-key SDBE.

In this paper, we introduce a feature of "designated ancestor " to HIBE and the simplified version, the binary tree encryption (BTE), and apply BTE with a designated ancestor for the public-key SDBE. As a result, BTE with a designated ancestor realizes the public-key SDBE more suitably than the previous approach.

Key words: hierarchical identity based encryption, public key broadcast encryption

1. Introduction

Broadcast Encryption (BE) gives a way to distribute digital contents to subscribers by using an open broadcast channel, in which a set of privileged users may be changed by each digital content. We focus on the BE for stateless recivers, in which each user is given a fixed set of keys beforehand and keeps using it to decrypt a broadcasted contents through the lifetime of the system. The stateless receivers are quite natural setting for application such as DVD decorder, etc. In[11], the subset-cover framework is formalized to realize the BE for stateless receivers, which consists of the following procedures for users $\mathcal{N}$ with $|\mathcal{N}| = N$ and a set of revoked users $\mathcal{R}$ with $|\mathcal{R}| = r$: define such a family of subsets of $\mathcal{N}$, $\mathcal{S} = \{S_i\}$, that can cover any set of previleged users $\mathcal{N} \setminus \mathcal{R}$ by disjoint subsets in $\mathcal{S}$ such as $\mathcal{N} \setminus \mathcal{R} = \cup S_{i_j}$; generate (computationally unrelated) each secret key for each set in $\mathcal{S}$; define a secret key K_u of a user u in such a way that all set keys that u belongs to can be generated by K_u; to send a message for a set of previleged users, cover users by disjoint subsets $\mathcal{N} \setminus \mathcal{R} = \cup S_{i_j}$ and encrypt a session key by each subset key; and for a previleged user u to decrypt a message, generate a subset key that u belongs to from K_u. Thus, the subset-cover framework is determined by the subset-cover family $\mathcal{S}$ and a key derivation algorithm that derives a corresponding subset key from a user key. The efficiency of BE depends on the transmission rate or the size of a user secret key which is denoted simply by $|K_u|$.

Two specific examples are proposed in[11], the complete subtree broadcast encryption (CSBE) and the subset difference broadcast encryption (SDBE), in which users are arranged to leaves of a binary tree $\mathcal{T}$ of hight t for $|\mathcal{N}| = N = 2^t$. CSBE is rather simple compared with SDBE, in which a set of privileged users is covered by a complete subtree; and the center has only to generate each secret key for each node of $\mathcal{T}$ and set a user secret key K_u as a set of node secret keys along the path ρ_u from the root to a user (leaf) u. CSBE does not need a key derivation algorithm since users have already all set keys that they belong to. In a sense, a user secret key is not compressed. The performance of the transmission rate or $|K_u|$ of CSBE is $r \log \frac{N}{r}$ or $O(\log N)$. On the other hand, SDBE combines two adjacent covers with a common ancestor in CSBE. So privileged users are in the difference set of two subsets $S_{v,w} = S_v \setminus S_w$, where $v, w \in \mathcal{T}$, v is an ancestor of w, and S_v or S_w is a complete subtree rooted at v or w, respectively. As a result, the number of subsets that a user belongs to becomes large although the transmission rate is reduced. In order to reduce the size of user secret key, a user secret key has to been compressed by a key derivation algorithm. The performance of the transmission rate or $|K_u|$ of SDBE is $(2r - 1)$ or $O(\log^2 N)$, respectively. Both the original CSBE and SDBE are realized by using a symmetric-key encryption[11].

Another aspect of BE is whether it is based on a symmetric key or public key encryption. In public-key BE, any contents provider including the center can broadcast their contents safely by using the same setting and instruments; and the security of system is no longer compromised by exposures of secret keys stored by providers because they don't have to store them. This is why the public-key BE for stateless recivers is desirable under the recent open network. To real-

† Japan Advanced Institute of Science and Technology

ize the public-key BE, there are mainly two problems. One is to reduce the size of public keys of all users. This means that it is not enough to give a pair of public and secret keys to a subset in $\mathcal{S}$. The other is to compress secret keys and derive a necesary secret key from the compression while the mathematical relation between secret key and public key holds.

Recently, a very powerful tool, called the bilinear Diffie-Hellman (BDH) problem as formalized in[3]. The BDH problem is believed to be difficult and can be constructed from Weil and Tate pairings associated with elliptic curves. Many new schemes have been realized by using the BDH problem. An ID-based encryption (IBE), in which a user's public key is given as a user ID such as an email address or name and the corresponding secret key is computed by a center, is realized by using BDH in[3]. A hierachical ID-based encryption (HIBE), a kind of IBE, is also realized by BDH in[9], in which any user is arraged to a node v of a tree except a root; the root v_0 of tree corresponds to the root center with a master secret key; a system public key is associated with the tree; a user public key v is an ID-tuple $v = v_1 \cdots v_l$ which represents all ancestors information; the corresponding user secret key $\mathtt{SK}_v$ is generated from its parent node; an encryption of a message to a node is done by using a system public key and the name of a node; and the cipher text can be decrypted by using the secret key of the target node. A binary tree encryption (BTE), introduced in[4], is a simple version of HIBE that allows only a binary tree, whose node has only a left or right child. BTE has the same properties as HIBE except using only a binary tree. A secure BTE scheme can be converted to a secure HIBE[4]. We sometimes restrict us to BTE instead of HIBE since it is rather easy to handle than HIBE. One of the remarkable applications for IBE and HIBE is public-key broadcast encryption[5]. HIBE or BTE possesses a key derivation algorithm for not only a root but also any node which computes its child's node secret key, which is the important difference for application to BE.

Let us go back to the public-key BE. By applying IBE or HIBE to CSBE or SDBE, respectively, those public-key versions can be realized in[5]. In the case of CSBE, IBE can be naturally applied to realize the public-key version by just generating each node secret key $\mathtt{sk}_v$ corresponding to each node name "v". Then the size of public keys, the transmission rate or $|K_u|$ is $O(1)$, $r \log \frac{N}{r}$ or $O(\log N)$ because a user public key is a node name from the feature of IBE. So we can enjoy the public key CSBE with no additional memory and computation, which will be shown in Section 2. On the other hand, in the case of SDBE, not IBE but HIBE is necessary to realize the public-key version because the key derivation algorithm on a node is necessary. The derivation way can be well explained by using a key derivation tree $\mathcal{T}_{\mathtt{KDT}}$. $\mathcal{T}_{\mathtt{KDT}}$ is an $(N-1)$-ary-tree defined by a set of diffrence subsets $\{S_{w,v}\}_{w,v}$ as follows: a subset $\{S_{w,w}\}_{w \in \mathcal{T} \setminus \{u\}}$ for initialization is associated to nodes of height 1, where $w \in \mathcal{T} \setminus \{u\}$ is a node in $\mathcal{T}$ except a leaf; difference subsets $\{S_{w,v}\}_v$ follows $S_{w,w}$ as a child node, where v is a child of w; difference subsets $\{S_{w,v'}\}_{v'}$ follows $S_{w,v}$ as a child node, where v' is a child of v; and the same procedures proceed until it reaches a leaf node. The key derivation tree, $\mathcal{T}_{\mathtt{KDT}}$, is naturally HIBE setting, on which HIBE is applied to realize the public-key SDBE in[5]: the root center generates a secret key for a node of height 1, $\{S_{w,w}\}_{w \in \mathcal{T} \setminus \{u\}}$, and then the node generates secret keys of its children's nodes $\{S_{w,v}\}_v$; and so forth. Thus, HIBE realizes the public-key SDBE as it is. Then the size of public keys, the transmission rate or $|K_u|$ is $O(1)$, $(2r-1)|C_{\mathtt{HIBE}}|$ or $O(|\mathtt{SK}_{\mathtt{HIBE}}| \log^2 N)$, where $|C_{\mathtt{HIBE}}|$ or $|\mathtt{SK}_{\mathtt{HIBE}}|$ represents the size of ciphertext or node secret key in HIBE, respectively. As a result, the public-key SDBE cannot be better than the symmetric-key (original) SDBE from either point of view of the transmission rate or the size of user secret key. We may note that the user (binary) tree cannot be applied to HIBE in this approach and thus BTE cannot be used because a key derivation tree $\mathcal{T}_{\mathtt{KDT}}$ is never a binary tree since N, the number of users, is usually larger than 4.

In this paper, we give another approach to realize the public-key SDBE by adding a feature to the original HIBE (BTE). In our approach, we use the user (binary) tree $\mathcal{T}$ itself instead of $\mathcal{T}_{\mathtt{KDT}}$ to derive a subset key by adding a feature to HIBE (BTE), and, thus, BTE is enough for our approach. For a sake of simplicity, we just focus on BTE since our notion holds into both BTE and HIBE. BTE itself has already the function of key derivation from an ancestor to a child, which can be considered as a compress function of secret keys because a compressed secret key, that is a secret key of an ancestor, can generate a secret key corresponding to a public key of any descendent. Apparently, IBE does not have such a feature. Therefore, BTE seems to be applied to the binary tree $\mathcal{T}$ to realize the public-key SDBE in such a way that IBE is applied to CSBE. However, it does not work straightforwardly by the following property of BTE: any ancestor as well as root can make a secret key of any descendant node and, thus, a cipher text to a node can be decrypted by any ancestor node even if the ancestor does not have the same secret key as that of the target node. We can say that BTE does not have ancestor-designated feature, that is a sender cannot control an ancestor who can decrypt a message to the target node. This feature realizes the hierarchical center structure, however, it is exactly a reason that we fail to apply BTE to the user tree $\mathcal{T}$ for the public-key SDBE. Because, in SDBE, two difference subsets $S_{w,v}$ and $S_{v',v}$ are defined as a different key-derivation group, where $w, v', v \in \mathcal{T}$; w is an ancestor of v' and v; and v' is an ancestor of v. Therefore, a member in $S_{w,v}$ must not be able to generate a secret key related to $S_{v',v}$ in the scenario of SDBE. However, a decendent v cannot restrict its ancestor of w or v' in the scenario of BTE. This is why the original BTE on the user tree $\mathcal{T}$ does not have enough

feature for SDBE. In order to overcome this problem, we introduce the feature of "a designated ancestor" to BTE, that is, a sender can control an ancestor who can decrypt a message to a target node. BTE with this feature, denoted by BTE-DA in this paper, can successfully realize the public-key SDBE. The performance of public-key SDBE based on BTE-DA is as follows: the size of public keys, the transmission rate or $|K_u|$ is $O(1)$, $(2r-1)|C_{\mathtt{BTE\text{-}DA}}|$ or $O(\log N|\mathtt{SK}_{\mathtt{BTE\text{-}DA}}|)$, where $|C_{\mathtt{BTE\text{-}DA}}|$ or $|\mathtt{SK}_{\mathtt{BTE\text{-}DA}}|$ represents the ciphertext size or the maximum size of node secret keys in BTE-DA, respectively. We also give a concrete example of BTE-DA. By applying our example, the performance of public-key SDBE is as follows: the transmission rate or $|K_u|$ is $(2r-1)|\log N|$ or $O(\log^2 N)$, which improves both performances of public-key SDBE[5]. The user secret key size of our scheme is still reduced even if compared with the combination of[5] and HIBE proposed newly in[1]. Furthermore, if an efficient BTE-DA with the constant ciphertext and key length should be proposed, $|K_u|$ is reduced to $O(\log N)$ and the transmission rate remains the same as $O(r)$, which means that it improves the performance of even the original SDBE[11].

This paper organizes as follow. Section 2 summarizes the basic notions of IBE, HIBE, BTE, and BE. Section 3 gives the functional definition of BTE with a designated ancestor and the security definition, and then presents a concrete example. Section 4 applies BTE with a designated ancestor to SDBE and presents the public-key SDBE.

2. Preliminary

This section summarizes the basic notions, ID-based encryption (IBE), hierarchical ID-based encryption (HIBE), binary tree encryption (BTE) and broadcast encryption (BE). Then we explain the notion of "designated ancestor" and discuss why we need the notion of "designated ancestor".

2.1 The Bilinear Diffie-Hellman Assumption

The security of schemes summarized here including our BTE with a designated ancestor is based on the difficulty of the bilinear Diffie-Hellman (BDH) problem as recently formalized in[3]. We review definitions relevant to this paper. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order q, where $\mathbb{G}_1$ is represented additively and $\mathbb{G}_2$ is represented multiplicatively. We use a non-degenerate bilinear map $\hat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ for which the followings hold:

(1) The map $\hat{e}$ is *bilinear*; that is, for all $P_0, P_1 \in \mathbb{G}_1$ and all $x, y \in \mathbb{Z}_q$ we have
$$\hat{e}(xP_0, yP_1) = \hat{e}(yP_0, xP_1) = \hat{e}(P_0, P_1)^{xy}.$$
(2) There is an efficient algorithm to compute $\hat{e}(P_0, P_1)$ for any $P_0, P_1 \in \mathbb{G}_1$.
(3) The map is non-degenerate, i.e. $\hat{e}(P, P) \neq 1$ for some $P \in \mathbb{G}_1$.

A *BDH parameter generator* $\mathcal{IG}$ is a randomized algorithm that takes a security parameter 1^k, runs in polynomial time, and outputs two groups $\mathbb{G}_1, \mathbb{G}_2$ with order q and a map $\hat{e}$ satisfying the above conditions. We define the *computational BDH problem with respect to* $\mathcal{IG}$ in the following: given $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$ output by $\mathcal{IG}$ along with random $P, aP, bP, cP \in \mathbb{G}_1$, compute $\hat{e}(P, P)^{abc}$. We say that $\mathcal{IG}$ *satisfies the BDH assumption* if the following is negligible (in k) for all PPT algorithms A:
$$\Pr[(\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}(1^k); P \leftarrow \mathbb{G}_1; a, b, c \leftarrow \mathbb{Z}_q : A(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, aP, bP, cP) = \hat{e}(P, P)^{abc}].$$
We note that BDH parameter generators for which the BDH assumption is believed to hold can be constructed from Weil and Tate pairings associated with supersingular elliptic curves or some ordinary elliptic curves[10].

2.2 Broadcast Encryption

Broadcast Encryption (BE) gives a way to distribute digital contents to subscribers by using an open broadcast channel, in which a set of privileged users may be changed by each digital content. Let $\mathcal{N}$ be the set of all users with $|\mathcal{N}| = N$ and $\mathcal{R}$ be the set of all revoked users with $|\mathcal{R}| = r$. Then privileged users are in $\mathcal{N} \setminus \mathcal{R}$. In[11], the subset-cover framework is formalized and two specific examples, the complete subtree broadcast encryption (CSBE) and the subset difference broadcast encryption (SDBE), are proposed.

Definition 1 (BE) BE consists of a 3-tuple of PPT algorithm (BE-Ini, BE-Enc, BE-Dec)*, where

- BE-Ini$(1^k, t, \mathcal{N})$, the initialization algorithm, for input of the security parameter 1^k and a set of users $\mathcal{N}$ with $|\mathcal{N}| = N = 2^t$, outputs a system public parameter PK which includes two encryption algorithms E_1 and E_2 for an encryption of session keys and messages, a family $\mathcal{S} = \{S_i\}$ of subsets of $\mathcal{N}$, the master's secret key SK which can compute all subset keys of $\mathcal{S}$, and a secret key K_u for a user $u \in \mathcal{N}$.
- BE-Enc$(PK, \mathcal{N} \setminus \mathcal{R}, \mathcal{S}, K, M)$, the encryption algorithm, for input of the public parameter PK, a set of privileged users $\mathcal{N} \setminus \mathcal{R}$, a family $\mathcal{S}$ of sets, a session key K, and an message M, covers $\mathcal{N} \setminus \mathcal{R} = \cup_j S_{i_j}$ by disjoint subsets $\{S_{i_j}\}_j$ and encrypts the session key K by using each subset key $K_{S_{i_j}}$ to $C = \langle \{S_{i_j}\}_j, \{E_1(K_{S_{i_j}}, K)\}_j, E_2(K, M) \rangle$, where $E_i(K, M)$ $(i = 1, 2)$ means an encryption of M by using a key K. Remark that the size of $\{E(K_{S_{i_j}}, K)\}_j$ is a dominant part of the transmission rate.
- BE-Dec(PK, K_u, C), for input of a secret key K_u of a user u, the public parameter PK, and a ciphertext C broadcasted by the center, finds a subset $S_{i_j} \ni u$, derives a subset key $K_{S_{i_j}}$ from K_u, decrypts $E_1(K_{S_{i_j}}, K)$ to K, and then decrypts $E_2(K, M)$ to M.

The efficiency of BE depends on the transmission rate

* BE-Ini can be separated into two algorithms BE-Ini and BE-Reg as in[5]: BE-Ini outputs a system parameter PK, a family $\mathcal{S} = \{S_i\}$, and the master's secret key SK; and BE-Reg outputs a secret key K_u for a user u.

$\sum_j |E_1(K_{S_{i_j}}, K)|$ and the size of a user secret key $|K_u|$. To realize the public-key BE, in which any contents provider including the center can broadcast a content safely by using a common setting of public-key BE, we also need to discuss the size of a family $|\mathcal{S}| = \#\{S_i\}$ or the number of subsets $|\mathcal{S}_u|$ that a user belongs to, which determines the size of public keys or user secret key, respectively.

Let us summarize concrete subset-cover methods, CSBE and SDBE. In both CSBE and SDBE, users are arranged to leaves of a binary tree $\mathcal{T}$, where we assume that $|\mathcal{N}| = N = 2^t$ for the sake of simplicity. We call $\mathcal{T}$ the user tree. In CSBE, the subset-cover family $\mathcal{S}$ consists of all complete subtrees $\{S_v\}$, where S_v means a complete subtree rooted at v and covers users in leaves. The center generates each secret key for each node of $\mathcal{T}$ (there are $2N-1$ nodes) and sets a user secret key K_u as a set of node secret keys along the path ρ_u from the root to a leaf u (there are $t+1$ node keys). As we see in[5], IBE can be naturally applied to CSBE to realize the public-key version: the center generates each node secret key $\mathtt{sk}_v$ associated with each node name "v" according to IBE; distributes each user key $K_u = \{\mathtt{sk}_v\}_{\rho_u \ni v}$ to each user u according to CSBE; to encrypt the session key K according to a CS cover set $\mathcal{N} \setminus \mathcal{R} = \cup S_v$, the sender executes $C_v = \mathtt{IBE\text{-}Enc}(PK, v, K)$ for each node v; to decrypt, each privileged user u finds $u \in S_v$ and executes $K = \mathtt{IBE\text{-}Dec}(PK, \mathtt{sk}_v, C_v)$ by using $\mathtt{sk}_v \in K_u$. Table 1 summarizes performance of the CSBE and the public-key CSBE.

On the other hand, SDBE combines two adjacent covers with a common ancestor in CSBE to reduce the transmission rate. So privileged users are in the difference set of two subsets $S_{v|_h,v} = S_{v|_h} \setminus S_v$, where $v = v_1 \cdots v_h \cdots v_l$, its ancestor $v|_h = v_1 \cdots v_h$, and $S_{v|_h}$ or S_v is a complete subtree rooted at $v|_h$ or v, respectively. For a set $S_{v|_h,v}$, we call $v|_h$ or v primary root or secondary root, respectively, as in[5]. We denote the path from $v|_h$ or the root of $\mathcal{T}$ to a user (leaf) u by $\rho_{v|_h,u}$ or simply ρ_u, respectively, and a set of nodes that just hang off $\rho_{v|_h,u}$ by $\mathcal{V}_{hang.\rho_{v|_h,u}}$. We note that for a node $v = v_1 \cdots v_h \cdots v_l$ and a user u

$$S_{v|_h,v} \ni u \Leftrightarrow \begin{cases} \rho_u \ni v|_h, \\ \mathcal{V}_{hang.\rho_{v|_h,u}} \ni v|_j \text{ for } h < \exists j \le l \end{cases} \quad (1)$$

The performance of $\mathcal{S}$ is not so good as CSBE since $|\mathcal{S}| = \mathcal{N} \log N$ and $|\mathcal{S}_u| = O(N)$.

In order to reduce the size of user secret key $|K_u|$ relative to $|\mathcal{S}_u|$, SDBE sets a secret key $\mathtt{SK}_{w,w}$ to the primary root w and derives a secret key of its descendant v and a secret key according to a set $S_{w,v}$, denoted by $\mathtt{SK}_{w,v}$ and $K_{S_{w,v}}$ respectively*, by using the following PPT key derivation algorithm BE-KDer, where $v = v_1 \cdots v_h \cdots v_l$ and $w = v|_h$.

- $\mathtt{BE\text{-}KDer}(v|_h, v, \mathtt{SK}_{v|_h,v}, PK) \to (\mathtt{SK}_{v|_h,v1}, \mathtt{SK}_{v|_h,v0}, K_{S_{v|_h,v}})$
  For input of a primary root $v|_h$, a descendant node v, a secret key $\mathtt{SK}_{v|_h,v}$ of v to the primary root, and public parameter PK, output secret keys $\mathtt{SK}_{v|_h,v0}$ and $\mathtt{SK}_{v|_h,v1}$ of v's children nodes $v0$ and $v1$, and a secret key $K_{S_{v|_h,v}}$ of a set $S_{v|_h,v}$.

We note that the center executes $\mathtt{BE\text{-}KDer}(v, v, \mathtt{SK}_{v,v}, PK)$ to generate $\mathtt{SK}_{v,v0}$ and $\mathtt{SK}_{v,v1}$, but any user except center does not have to or cannot execute $\mathtt{BE\text{-}KDer}(v, v, \mathtt{SK}_{v,v}, PK)$ since she/he does not have $\mathtt{SK}_{v,v}$. $\mathtt{BE\text{-}KDer}(v|_h, v, \mathtt{SK}_{v|_h,v}, PK)$ has to satisfy the following features for 3-tuple nodes $(v|_h, v|_j, v)$ of $v = v_1 \cdots v_h \cdots v_j \cdots v_l$:

- *One-way feature for a common primary root*: Given $\mathtt{SK}_{v|_h,v|_j}$ it is easy to compute $\mathtt{SK}_{v|_h,v}$, but given $\mathtt{SK}_{v|_h,v}$ it is difficult to compute $\mathtt{SK}_{v|_h,v|_j}$.
- *CCA1-security*: It is difficult to compute $\mathtt{SK}_{v|_h,v}$ without knowledge of any secret key $\mathtt{SK}_{v|_h,v|_j}$ of an ancestor $v|_j$ of v.
- *Ancestor-designated feature***: Given $\mathtt{SK}_{v|_h,v}$, it is difficult to compute $\mathtt{SK}_{v|_j,v}$, where both $v|_h$ and $v|_j$ are ancestors of v but $v|_j$ is lower (younger) than $v|_h$.

Then a user secret key is given as

$$K_u = \{\mathtt{SK}_{v,d^v}\}_{(v,d^v) \in \rho_u \times \mathcal{V}_{hang.\rho_{u,v}}}.$$

From (1) and the feature of BE-KDer, the user secret key K_u enables a user u to compute any subset key that the user belongs to. Such a derivation way can be well explained by using a key derivation tree $\mathcal{T}_{\mathtt{KDT}}$. $\mathcal{T}_{\mathtt{KDT}}$ is an $(N-1)$-ary-tree defined by a set of difference subsets $\{S_{w,v}\}_{w,v}$: a subset $\{S_{w,w}\}_{w \in \mathcal{T} \setminus \{u\}}$ for initialization is associated to nodes of height 1 and difference subsets $S_{w,v}$ with the primary node w and its descendent node v follows. Here, $w \in \mathcal{T} \setminus \{u\}$ is a node in $\mathcal{T}$ except a leaf. The SD-BE key derivation can be rehashed by using $\mathcal{T}_{\mathtt{KDT}}$: the root center corresponds to the root of $\mathcal{T}_{\mathtt{KDT}}$; each node has each node secret key; and a node secret key can be generated by its parent's node secret key.

$\mathcal{T}_{\mathtt{KDT}}$ is naturally HIBE setting[5] and, thus, HIBE can realize the public-key SDBE as it is. This is why the transmission rate or the size of a user secret key, $|K_u|$, is $(2r-1)|C_{\mathtt{HIBE}}|$ or $O(|\mathtt{SK}_{\mathtt{HIBE}}| \log^2 N)$, respectively, as in Table 1. Compared with the performance of the original SDBE, the public-key SDBE cannot be better from either point of view of the transmission rate or $|K_u|$. If we use a scheme in[9] as HIBE described in[5], then both $|\mathtt{SK}_{\mathtt{HIBE}}|$ and $|C_{\mathtt{HIBE}}|$ is $O(\log N)$ and so the transmission rate or $|K_u|$ becomes $O(r \log N)$ or $O(\log^3 N)$, respectively. It is no better than public-key CSBE. Especially, $|K_u|$ is rather large. Recently, more efficient HIBE is reported in[1], in which $|\mathtt{SK}_{\mathtt{HIBE}}|$ or $|C_{\mathtt{HIBE}}|$ is $O(1)$ or $O(\log N)$, respectively. Then a combination of[1] and[5] can work in the transmission

* The secret key node $\mathtt{SK}_{v|_h,v}$ in this paper is denoted by proto-key in[5] and LABEL in[11].

** The notion of ancestor-designated feature is included in the notion of CCA1-security, but that exactly shows the reason why HIBE cannot be simply applied to a user-tree $\mathcal{T}$ which will be discussed below.

rate or $|K_u|$ of $O(r)$ or $O(\log^3 N)$, respectively. So the transmission rate is improved but the size of user secret key remains large. We note that BTE cannot be used in this approach because a key derivation tree $\mathcal{T}_{\mathtt{KDT}}$ is never a binary tree for $N > 4$.

Let us leave $\mathcal{T}_{\mathtt{KDT}}$ and focus on the binary tree $\mathcal{T}$ of users to clarify the necessary feature of HIBE for a realization of the public-key SDBE. In this situation, HIBE (more strictly BTE is enough) would be applied to SDBE: the center generates the root secret key $\mathtt{SK}_\varepsilon$ and generates each node secret key associated with each node name "$v = v_1 \cdots v_h \cdots v_j \cdots v_l$" according to HIBE; distributes a user key $K_u = \{\mathtt{SK}_v\}_{v \in \mathcal{V}_{hang,\rho_u}}$ to a user u; to encrypt the session key K to a user in $S_{v|_j,v}$, the sender executes $C = \mathtt{HIBE\text{-}Enc}(PK, v, K)$ by using a node name $v = v_1 \cdots v_h \cdots v_j \cdots v_l$. However, it does not work since HIBE does not satisfy the ancestor-designated feature and, thus, any ancestor node $v|_h$ of $v|_j$ can generate any its descendant node secret key such as $\mathtt{SK}_v$. This means that a user in $S_{v|_h,v}$ can decrypt C which is a ciphertext to a user in $S_{v|_j,v}$.

3. Binary Tree Encryption with a Designated Ancestor

This section gives the notion of binary tree encryption (BTE) with a designated ancestor. BTE is a simple version of hierarchical identity-based encryption (HIBE)[9]. For a sake of simplicity, we just focus on BTE since our notion of a designated ancestor can be applied into both BTE and HIBE. Our notion of HIBE with a designated ancestor will be described in the final paper. One of interesting properties of BTE is: any ancestor as well as root can make a secret key of any descendant node and, thus, a cipher text to a node can be decrypted by any ancestor node even if the ancestor does not have the same secret key as that of a target node. We can say that BTE does not have ancestor-designated feature, that is a sender cannot control an ancestor who can decrypt a message to a targeted node. This is why neither BTE cannot be applied on the user tree $\mathcal{T}$ to realize the public-key SDBE, seen in Section 2. In this section, we give the notion of binary tree encryption with a designated ancestor and a concrete example.

3.1 Functional and Security Definition

This section gives the functional definition of binary tree encryption with a designated ancestor and then the security definition. Let t denote the height of binary tree $\mathcal{T}$.

Definition 2 (BTE-DA) BTE-DA consists of a 5-tuple of PPT algorithms ($\mathtt{KGen}$, $\mathtt{KDer}_r$, $\mathtt{KDer}_p$, $\mathtt{Enc}$, $\mathtt{Dec}$), where

- $\mathtt{KGen}(1^k, t)$, the root center key-generation algorithm, for input of security parameter k and the height t of binary tree, outputs a system public key PK that includes system parameter and the root center's secret key $\mathtt{SK}_\varepsilon$.
- $\mathtt{KDer}_r(PK, v, \mathtt{SK}_\varepsilon)$, the root center key-derivation algorithm, for input of the public key PK, a node v whose height is l, and the root secret key $\mathtt{SK}_\varepsilon$, outputs the v's secret key with height h, $\mathtt{sk}_{v,l}$. A node secret key of v with height l, $\mathtt{sk}_{v,l}$, means the beginning of secret-key sequence and, thus, no ancestor of v except the root cannot generate the secret key.
- $\mathtt{KDer}_p(PK, v, \mathtt{SK}_v)$, the key derivation algorithm, for input of the public key PK, a node v with height l, and the secret key $\mathtt{SK}_v = \{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,l}\}$, output the node secret keys of children $v0$ and $v1$ whose height is from 1 to l. Note that although height of children is $l+1$, their secret keys with the height $l+1$ are not generated by their parent node but root.
- $\mathtt{Enc}(PK, v, h, M)$, the encryption algorithm, for input of the public key PK, user ID $v = v_1 \cdots v_h \cdots v_l$, height h of a designated ancestor, and a message M, computes a ciphertext C.
- $\mathtt{Dec}(\mathtt{SK}_v, C, v, h) = M$, the decryption algorithm, for input of a user secret key $\mathtt{SK}_v$, a ciphertext C, and height h of a designated ancestor, decrypts C to M.

The BTE-DA is a special case of the BTE and, thus, the security definition follows mostly that of BTE[4] or HIBE[9], which has the decryption and the key derivation oracles. The important difference lies in the key derivation oracles. In the case of BTE or HIBE, an adversary is not allowed to ask a secret key of any node in the path ρ_v from the root to a target node v. But, in our BTE-DA, an adversary is allowed to ask a secret key of a node w in the path ρ_v until the height of w is lower than the target height. We give the security definition of BTE-DA as follows.

Definition 3 We say that a BTE-DA scheme is IND-BTEDA-CCA secure against adaptive chosen ciphertext and node adversary if the advantage of any PPT adversary A against the challenger in the following experiment is negligible.

Set up The challenger takes a security parameter k and the height t of binary tree and executes $\mathtt{KGen}(1^k, t)$. Then it gives A the public parameter PK and keeps the root secret key $\mathtt{SK}_\varepsilon$.

Phase 1 A issues a number of queries $q_1, \cdots, q_m$, where query q_i is one of the following:

- Node-secret-key query: On the query of a node v, output the corresponding node key $\mathtt{SK}_v$.
- Decryption query: On the query of a node $v = v_1 \cdots v_h \cdots v_l$, the target height h, and a ciphertext C, output the recovering message M.

Challenge A outputs two equal length messages $M_0, M_1 \in \{0,1\}^*$, a node v^*, and a height h^*. The only constraint is that A did not previously issue a node-secret-key query on $v^*|_i$ with $i \geq h$ for the target node v^*. Then the challenger picks a random bit $b \in \{0,1\}$, sets $C^* = \mathtt{Enc}(PK, v^*, h^*, M_b)$, and sends C^* to A as a challenge.

Phase 2 A continues a number of queries $q_{m+1}, \cdots, q_n$, where a query q_i is one of the fol-

Table 1 Comparison of CS and SD methods

	CSBE [11]	SDBE [11]	PK-CSBE [5]	PK-SDBE [5]	
				general	[5]+[9]
$\|S\|$	$2N-1$	$N \log N$	$2N-1$	$N \log N$	$N \log N$
$\|S_u\|$	$\log N$	$O(2N)$	$\log N$	$O(2N)$	$O(2N)$
public-key size	—	—	$O(1)$	$O(1)$	$O(1)$
transmission rate	$r \log \frac{N}{r}$	$2r-1$	$r \log \frac{N}{r}$	$(2r-1)\|C_{\mathtt{HIBE}}\|^{\dagger}$	$(2r-1)\log N$
$\|K_u\|$	$O(\log N)$	$O(\frac{1}{2}\log^2 N)$	$O(\log N)$	$O(\|\mathtt{SK}_{\mathtt{HIBE}}\| \log^2 N)^{\dagger}$	$O(\log^3 N)$

†: $|C_{\mathtt{HIBE}}|$ or $|SK_{\mathtt{HIBE}}|$ represents the size of ciphertext or node secret key in HIBE, respectively.

lowings in the same way as Phase 1:

- Node-secret-key query: On the query of v under the constraint in Challenge, output the corresponding node key $\mathtt{SK}_v$.
- Decryption query: On the query of $(v, h, C) \neq (v^*, h^*, C^*)$, output the recovering message M.

Guess A outputs a guess $b' \in \{0,1\}$. The adversary wins the game if $b = b'$. The advantage of A attacking the scheme is defined as $|\Pr[b = b'] - 1/2|$. We can also define the chosen plaintext security for a BTE-DA scheme as in[1),3),9)], in which A is not allowed to issue any decryption query. But A still issues adaptive node-secret-key queries. This adversary notion is termed as BTEDA-CPA.

Definition 4 We say that a BTE-DA scheme is IND-BTEDA-CPA-secure against adaptive chosen node attacks if the advantage of any PPT adversary A against the challenger in the experiment defined in Definition 3 without decryption oracle is negligible.

3.2 A BTE with a Designated Ancestor Based on the BDH Assumption

Let us denote $v_{[i,j]} = v_i \cdots v_j$ for a node $v = v_1 \cdots v_i \cdots v_j \cdots v_l$ in addition to the notation of prefix $v|_i = v_1 \cdots v_i$. Here we only describe a basic scheme, but it is also extended to be secure against chosen-ciphertext security by using the CCA2-transformation[8),12)] as in[9)].

$\mathtt{KGen}(1^k, t)$ executes the followings:

(1) Run $\mathcal{IG}(k)$ to generate groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order q and bilinear map $\hat{e}$.
(2) Choose a random generator $P \in \mathbb{G}_1$ and a random secret $\alpha \in \mathbb{Z}_q$ and compute $Q = \alpha P$.
(3) Compute the root secret key $\mathtt{SK}_{\varepsilon,i} = \alpha H(\underbrace{\varepsilon || \cdots || \varepsilon}_{i})$ for $1 \leq i \leq t$.
(4) Choose two cryptographic hash functions, $H_1 : \{0,1\}^* \to \mathbb{G}_1$ and $H_2 : \mathbb{G}_2 \to \{0,1\}^n$, where the message space is $\{0,1\}^n$.
(5) The public key $PK = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, H_1, H_2, P, Q, t\}$ and the root secret key $\mathtt{SK}_{\varepsilon} = \{\mathtt{SK}_{\varepsilon,1}, \cdots, \mathtt{SK}_{\varepsilon,t}\}$.

Let a node $v = v_1 \cdots v_l$ in $\mathcal{T}$. Then a node secret key of v will consist of $(3l-1)$ group elements, denoted by $\mathtt{SK}_v = \{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,l}\}$, where $\mathtt{sk}_{v,1} = \{R_{v|_1,1}, \cdots, R_{v,1}, S_{v,1}\}$ and $\mathtt{sk}_{v,i} = \{R_{v|_i,i}, S_{v,i}\}$ for $2 \leq i \leq l$. Those group elemets $\mathtt{sk}_{v,l}$ or $\{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,l-1}\}$ are given as outputs of $\mathtt{KDer}_r$ or $\mathtt{KDer}_p$ executed by the root or the parent node $v|_{l-1}$, respectively. The size of node secret key is $O(l)$, which is the same as that of node secret key in BTE[4)].

$\mathtt{KDer}_r(PK, v, \mathtt{SK}_{\varepsilon,l})$ executes the followings:

(1) Let $v = v_1 \cdots v_l$.
(2) Choose a random secret $\alpha_{v,l} \in \mathbb{Z}_q$ and compute $R_{v,l} = \alpha_{v,l} P$ and $S_{v,l} = \mathtt{SK}_{\varepsilon,l} + \alpha_{v,l} H_1(v)$.
(3) Output $\mathtt{sk}_{v,l} = \{R_{v,l}, S_{v,l}\}$.

$\mathtt{KDer}_p(PK, v, \mathtt{SK}_v)$ executes the followings:

(1) Let $v = v_1 \cdots v_l$ and its children nodes vv_{l+1} with $v_{l+1} = 0$ or 1.
(2) Parse $\mathtt{SK}_v = \{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,l}\}$, where $\mathtt{sk}_{v,1} = \{R_{v|_1,1}, \cdots, R_{v,1}, S_{v,1}\}$ and $\mathtt{sk}_{v,i} = \{R_{v|_i,i}, S_{v,i}\}$ for $2 \leq i \leq l$.
(3) Choose a random secret $\alpha_{vv_{l+1},1} \in \mathbb{Z}_q$.
(4) Compute $R_{vv_{l+1},1} = \alpha_{vv_{l+1},1} P$ and $S_{vv_{l+1},1} = S_{v,1} + \alpha_{vv_{l+1},1} H_1(vv_{l+1})$.
(5) Set $\mathtt{sk}_{vv_{l+1},1} = \{R_{v|_1,1}, \cdots, R_{v,1}, R_{vv_{l+1},1}, S_{vv_{l+1},1}\}$.
(6) Compute $S_{vv_{l+1},i} = S_{v,i} + \alpha_{vv_{l+1},1} H_1(v_{[i,l]} v_{l+1})$ for $2 \leq i \leq l$.
(7) Set $\mathtt{sk}_{vv_{l+1},i} = \{R_{v|_i,i}, S_{vv_{l+1},i}\}$ for $2 \leq i \leq l$.
(8) Output $\{\mathtt{sk}_{vv_{l+1},1}, \cdots, \mathtt{sk}_{vv_{l+1},l}\}$.

A node secret key $\mathtt{SK}_v$ of $v = v_1 \cdots v_l$ consists of $\mathtt{SK}_v = \{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,l-1}, \mathtt{sk}_{v,l}\}$ after receiving each output of $\mathtt{KDer}_p(PK, v|_{l-1}, \mathtt{SK}_{v|_{l-1}})$ and $\mathtt{KDer}_r(PK, v, \mathtt{SK}_{\varepsilon,l})$. A secret key of a node $v = v_1$ is generated by the root's execution of $\mathtt{KDer}_r(PK, v, \mathtt{SK}_{\varepsilon,1})$.

$\mathtt{Enc}(PK, v, h, M)$ executes the followings:

(1) Let $v = v_1 \cdots v_h \cdots v_l$.
(2) Choose a random $\gamma \in \mathbb{Z}_q$.
(3) Compute $C = (\gamma P, \gamma H_1(v|_h), \gamma H_1(v_{[h,h+1]}), \cdots, \gamma H_1(v_{[h,l]}), M \oplus H_2(d))$, where $d = \hat{e}(Q, H_1(\underbrace{\varepsilon || \cdots || \varepsilon}_{h}))^{\gamma}$.
(4) Output (C, v, h).

$\mathtt{Dec}(\mathtt{SK}_v, C, h)$ executes the followings:

(1) Let $v = v_1 \cdots v_h \cdots v_l$ and $C = (U_0, U_h, \cdots, U_l, V)$.
(2) Parse $\mathtt{SK}_v = \{\mathtt{sk}_{v,1}, \cdots, \mathtt{sk}_{v,h}, \cdots \mathtt{sk}_{v,l}\}$, $\mathtt{sk}_{v,1} = \{R_{v|_1,1}, \cdots, R_{v|_{h+1},1}, \cdots, R_{v,1}, S_{v,1}\}$ and $\mathtt{sk}_{v,h} = \{R_{v|_h,h}, S_{v,h}\}$.
(3) Compute $m = V \oplus H_2(d)$, where

$$d = \frac{\hat{e}(U_0, S_{v,h})}{\prod_{i=h+1}^{l} \hat{e}(R_{v|_i,1}, U_i) \cdot \hat{e}(R_{v|_h,h}, U_h)}.$$

The decryption succeeds as follows,

The knowledge of $S_{v,h}$ in $\mathtt{sk}_{v,h}$ is necessray to de-

$$
\begin{aligned}
d &= \frac{\hat{e}(U_0, S_{v,h})}{\prod_{i=h+1}^{l} \hat{e}(R_{v|_i,1}, U_i) \cdot \hat{e}(R_{v|_h,h}, U_h)} \\
&= \frac{\hat{e}(\gamma P, \alpha H_1(\varepsilon||\cdots||\varepsilon) + \alpha_{v|_h,h} H_1(v|_h) + \sum_{i=h+1}^{l} \alpha_{v|_i,1} H_1(v_{[h,i]}))}{\prod_{i=h+1}^{l} \hat{e}(\alpha_{v|_i,1} P, \gamma H_1(v_{[h,i]})) \cdot \hat{e}(R_{v|_h,h}, \gamma H_1(v|_h))} \\
&= \frac{\hat{e}(Q, H_1(\varepsilon||\cdots||\varepsilon))^{\gamma} \cdot \hat{e}(\alpha_{v|_h,h} P, \gamma H_1(v|_h)) \cdot \prod_{i=h+1}^{l} \hat{e}(\alpha_{v|_i,1} P, \gamma H_1(v_{[h,i]}))}{\prod_{i=h+1}^{l} \hat{e}(\alpha_{v|_i,1} P, \gamma H_1 v_{[h,i]}) \cdot \hat{e}(R_{v|_h,h}, \gamma H_1(v|_h))} \\
&= \hat{e}(Q, \underbrace{H_1(\varepsilon||\cdots||\varepsilon)}_{h})^{\gamma}.
\end{aligned}
$$

crypt a ciphertext C given by $\mathtt{Enc}(PK, v, h, M)$ with $v = v_1 \cdots v_h \cdots v_l$. From the feature of $\mathtt{SK}_v$, no ancestor of v with the height $< h$ has any information of $S_{v,h}$. Therefore only ancestors of v with the height $\geq h$ can decrypt the C. This is why our scheme realizes the feature of a designated ancestor as well as binary tree encryption.

3.3 Efficiency and Security

Table 2 summarizes the efficiency of our scheme compared with that of BTE, both of which assume a random oracle model. Our scheme realizes the feature of a designated ancestor as well as binary tree encryption with slightly additional computation and memory cost of only a key derivation to BTE. We remark that the encryption/decryption time and ciphertext length is even reduced by designating an ancestor with the height > 1. We give the following theorem on the security. The proof is done in the same way as[9], which will be described in the final paper because of the lack of space.

Theorem 1 If $\mathcal{IG}$ satisfies the computational BDH assumption and H_2 is a random oracle model, then our scheme described above is IND-BTEDA-CPA secure.

4. Public-key Broadcast Encryption

We apply BTE-DA to realize the public-key SDBE. In our scheme, users $u \in \mathcal{N}$ are arranged to leaves of a binary tree $\mathcal{T}$ with the height t as in Section 2, where we assume that $|\mathcal{N}| = N = 2^t$ for the sake of simplicity. The public-key SDBE (BE-Ini, BE-Enc, BE-Dec) can be realized by using BTE-DA ($\mathtt{KGen}$, $\mathtt{KDer}_r$, $\mathtt{KDer}_p$, Enc, Dec) as follows.

$\mathtt{BE\text{-}Ini}(1^k, t, \mathcal{N})$ executes the followings:

(1) Run $\mathtt{KGen}(1^k, t)$ and get outputs of PK and the root secret key $\mathtt{SK}_\varepsilon$.

(2) Run $\mathtt{KDer}_r(PK, v, \mathtt{SK}_\varepsilon)$ and $\mathtt{KDer}_p(PK, v|_{l-1}, \mathtt{SK}_{v|_{l-1}})$ for $v = v_1 \cdots v_l$ from $l = 1$ to t one by one and generate node secret keys $\{\mathtt{SK}_v\}_{v \in \mathcal{T}}$.

(3) Set a user secret key K_u as a set of secret keys of nodes just hanging off the path ρ_u, $\mathcal{V}_{hang.\rho_u}$, (there are $t+1$ nodes). Then $K_u = \{\mathtt{SK}_v\}_{v \in \mathcal{V}_{hang.\rho_u}}$.

(4) Output a system parameter PK, the SD-family $\mathcal{S} = \{S_i\}$, the master's secret key $\mathtt{SK}_\varepsilon$, and a secret key K_u for a user u together with an encryption algorithm E_2 for contents.

$\mathtt{BE\text{-}Enc}(PK, \mathcal{N} \setminus \mathcal{R}, \mathcal{S}, K, M)$ executes the followings:

(1) Cover $\mathcal{N} \setminus \mathcal{R} = \cup_v S_{v|_h,v}$ by disjoint subsets according to SDBE, where $v = v_1 \cdots v_h \cdots v_l$.

(2) For each subset $S_{v|_h,v}$, encrypt K to $C_{v,h}$ by $\mathtt{Enc}(PK, v, h, M)$, where $C_{v,h} = (\gamma P, \gamma H_1(v|_h), \gamma H_1(v_{[h,h+1]}), \cdots, \gamma H_1(v_{[h,l]}), M \oplus H_2(d))$ for $d = \hat{e}(Q, \underbrace{H_1(\varepsilon||\cdots||\varepsilon)}_{h})^{\gamma}$.

(3) Output $C = \langle \{S_{v|_h,v}\}_v, \{C_{v,h}\}_v, E_2(K, M) \rangle$.

$\mathtt{BE\text{-}Dec}(PK, K_u, C)$ executes the followings:

(1) Find a subset $S_{v|_h,v} \ni u$ according to SDBE.

(2) Take a node secret key $\mathtt{SK}_{v|_j} \in K_u$ with $v|_j \in \mathcal{V}_{hang.\rho_{v|_h,u}} \cap \rho_{v|_h,v}$ for $h \leq \exists j \leq l$. Such $v|_j$ exactly exists from Equation (1).

(3) Execute $\mathtt{KDer}_p(PK, v|_j, \mathtt{SK}_{v|_j})$ one by one to derivate $\mathtt{SK}_v$ for $v = v_1 \cdots v_h \cdots v_j \cdots v_l$.

(4) Execute $\mathtt{Dec}(\mathtt{SK}_v, C, h)$ to get K, decrypt $E_2(K, M)$ to M, and output M.

Table 3 shows the performance of our scheme from the viewpoint of general and concrete BTE-DA in Section 3. Compared with the combination of[5] +[9] or[5] +[1] in Table 1, $|K_u|$ is reduced with the same size of transmission rate as that of[5] +[9]. We note that if an efficient BTE-DA with the constant ciphertext and key length should be proposed, the size of transmission rate or $|K_u|$ is reduced to $O(r)$ or $O(\log N)$, respectively. This means that BTE-DA with the constant ciphertext and key length improves the size of user secret key of the original SDBE[11] with the same size of transmission rate. Such an improvement can never accomplish by the previous approach of using HIBE[5].

Acknowledgment

This study is partly supported by "Research on key intrusion resilient encryption" (16016242), Grant-in-Aid Scientific Research on Priority Area "Informatics" (Area #004).

References

1) D. Boneh, X Boyen, and E. Goh, "Hierachical identity based encryption with constant size ciphertext", preprint (www.iacr.org).

2) R. Anderson. "Two remarks on public-key cryptology." Invited Lecture, *ACM-CCCS'97*. Available at `http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf`.

3) D. Boneh and M. Franklin. "Identity based encryption from the Weil pairing." *Advances in Cryptology — Crypto 2001*, LNCS vol. 2139, Springer-Verlag, 2001.

Table 2 Efficiency on a binary tree $\mathcal{T}$ with the height t

	BTE-DA	BTE[4)]
Key generation time	$O(t)$	$O(1)$
Public key size	$O(1)$	$O(1)$
Secret key size (at a node $v^{\dagger}$)	$O(l)$	$O(l)$
Key derivation time (at a node $v^{\dagger}$)	$O(l)$	$O(1)$
Encryption/Decryption time (to a node v with a designated ancestor ‡)	$O(l-h)$	$O(l)$
Ciphertext length †	$O(l-h)$	$O(l)$

†: the height of v is $l \leq t$.
†: the height of a designated ancestor of v is $h \leq l$.

Table 3 Performance of SDBE using BTE-DA

	BTE-DA-based SDBE	
	general	our scheme in Section 3
$\lvert\mathcal{S}\rvert$	$N \log N$	$N \log N$
$\lvert\mathcal{S}_u\rvert$	$O(2N)$	$O(2N)$
public-key size	$O(1)$	$O(1)$
transmission rate	$(2r-1)\lvert C_{\mathtt{BTE\text{-}DA}}\rvert^{\dagger}$	$(2r-1)\log N$
$\lvert K_u\rvert^{\ddagger}$	$O(\log N \lvert \mathtt{SK}_{\mathtt{BTE\text{-}DA}}\rvert)^{\dagger}$	$O(\log^2 N)$

†: $\lvert C_{\mathtt{BTE\text{-}DA}}\rvert$, $\lvert SK_{\mathtt{BTE\text{-}DA},i}\rvert$, or $\lvert SK_{\mathtt{BTE\text{-}DA}}\rvert$ represents the size of ciphertext or the maximum node secret key in BTE-DA, respectively.
‡: $\lvert K_u\rvert$ is evaluated more precisely to $O(\sum_{i=1}^{\log N} \lvert \mathtt{SK}_{\mathtt{BTE\text{-}DA},i}\rvert)$, where $\lvert SK_{\mathtt{BTE\text{-}DA},i}\rvert$ represents node secret key with height i in BTE-DA.

Full version to appear in *SIAM J. Computing* and available at `http://eprint.iacr.org/2001/090`.

4) R. Canetti, S. Halevi, and J. Katz. "A forward-secure public-key encryption scheme." *Advances in Cryptology — Eurocrypt 2003*, LNCS vol. **2656**, Springer-Verlag, 2003.
5) Y. Dodis and N. Fazio, "Public Key Broadcast Encryption for Stateless Receivers", *proceeding of ACM DRM '02*, LNCS **2696**(2002), Springer-Verlag, 61–80.
6) Y. Dodis, M. Franklin, J. Katz, A. Miyaji and M. Yung. "Intrusion-resilient public-key encryption." *RSA — Cryptographers' Track 2003*, LNCS 2612, Springer-Verlag, 2003.
7) Y. Dodis, M. Franklin, J. Katz, A. Miyaji and M. Yung. "Generic construction for Intrusion-resilient public-key encryption." *RSA — Cryptographers' Track 2004*, LNCS **2964**(2004), Springer-Verlag, 81-98.
8) E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. Crypto '99.
9) C. Gentry and A. Silverberg, "Hierarchical ID-Based Cryptography," *Asiacrypt 2002*, LNCS vol.2501(2002), Springer-Verlag, 548–566.
10) A. Miyaji, M. Nakabayashi and S. Takano, "New explicit conditions of Elliptic Curve Traces under FR-reduction", *IEICE Trans.*, Fundamentals. vol. E84-A, No.5(2001), 1234-1243.
11) D. Naor, M. Naor and J. Lotspiech, "Revocation and Tracing Scheme for Stateless Receivers", *Advances in Cryptology-CRYPTO 2001*, LNCS **2139**(2001), Springer-Verlag, 41–62.
12) T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-Security Asymmetric Cryptosystem Transform. CT-RSA 2001.