# Fully Secure Codes Based Tracing and Revoking Scheme with Constant Ciphertext

Xingwen Zhao, Hui Li

*State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071*
*School of Telecommunications Engineering, Xidian University, Xi'an 710071*
*Peoples R China*

**Abstract**

In broadcast encryption system certain users may leak their decryption keys to build pirate decoders, so traitor tracing is quite necessary. There exist many codes based traitor tracing schemes. As pointed out by Billet and Phan in ICITS 2008, these schemes lack revocation ability. The ability of revocation can disable identified malicious users and users who fail to fulfill the payments, so that the broadcast encryption system can be more practical. Recently, Zhao and Li presented a construction of codes based tracing and revoking scheme which achieves user revocation as well as traitor tracing. However, their scheme is only secure against chosen plaintext attacks under selective-adversary model with random oracle. In this paper, we obtain a new construction of codes based tracing and revoking scheme which is proved secure against chosen ciphertext attacks under adaptive-adversary model without random oracle. Our idea is to insert codeword into Boneh and Hamburg's identity based broadcast encryption scheme to retain the ability of user revocation and use Boneh and Naor's method to trace traitors. Our fully secure scheme is roughly as efficient as Zhao and Li's scheme while the security is enhanced.

*Keywords:* Broadcast encryption, traitor tracing, collusion secure codes, user revocation, copyright protection.

*Email address:* sevenzhao@hotmail.com, lihui@mail.xidian.edu.cn (Xingwen Zhao, Hui Li)

## 1. Introduction

Broadcast encryption is efficient in distributing encrypted content to subscribers via an insecure open channel so that only the qualified receivers can recover the messages. It is very useful and enjoys many applications such as pay-TV systems, distribution of copyrighted digital content via CD and DVD. In these applications, certain users (called traitors) may leak their decryption keys to build pirate decoders, so the ability of traitor tracing is necessary.

Chor et al. [1] presented the first traitor tracing scheme against pirate decoders. Since then, many traitor tracing schemes against pirate decoders have been proposed and among them there is a kind of traitor tracing schemes called codes based schemes (such as [2, 3, 4, 5, 6, 7, 8]). In these schemes, each user is assigned decryption keys according to each bit of his/her codeword. When a pirate decoder is captured, the tracer analyzes the keys used in each bit position and recovers the codeword embedded in the decoder. The recovered codeword can be used to trace back to at least one of the traitors. However, Billet and Phan stated in [6] that it is still an open problem whether the revocation ability can be achieved in codes based traitor tracing scheme.

Recently, Zhao and Li [9] presented a construction of codes based tracing and revoking scheme. They embed collusion secure code into each user's decryption key of Park et al.'s public key broadcast encryption scheme [10], so the sender can send messages to a set of designated receivers. The users outside the set are revoked. When traitor tracing is needed, their scheme recovers the embedded codeword from the pirate decoder by using the method by Boneh and Naor [5]. Their scheme achieves user revocation as well as traitor tracing. Moreover, their scheme is constant in ciphertext length and resistant to public collaboration attacks [11]. However, their scheme is only secure against chosen plaintext attacks under selective-adversary model with random oracle.

### *1.1. Our Contributions*

We describe a new construction of codes based tracing and revoking scheme which is secure against chosen ciphertext attacks under adaptive-adversary model without random oracle. Our idea is to insert codeword into each decryption key of Boneh and Hamburg's identity based broadcast encryption scheme [12] and use the idea of Gentry [13] to achieve fully security with a tight reductio. The ability of user revocation is inherit from Boneh

and Hamburg's scheme and traitor tracing is achieved by using the method by Boneh and Naor [5] where only one tracing position is considered in each broadcasting. Our scheme is almost as efficient as Zhao and Li's scheme [9].

*1.2. Related Works on Traitor Tracing against Pirate Decoders*

Many traitor tracing schemes against pirate decoders have been proposed since the introduction of traitor tracing in [1], we can roughly classify them into three categories.

The first category is called combinatorial, as presented in [1, 14, 15, 16]. In these schemes, certain subsets of decryption keys are carefully selected to be put in each decoder. It is possible for the tracer to identify one of the traitors, by analyzing what keys are used in the captured pirate decoder. The second category is called algebraic, as described in [17, 18, 19, 20, 21, 22, 23, 24, 25, 26]. In these schemes, algebraic methods are employed when assigning private keys to users, and public broadcasting is allowed since public-key techniques are employed. The third category is called codes (e.g. collusion secure codes [5] and IPP codes [27]) based schemes, in which the ideas of two previous classes are combined. For instance, the schemes presented in [2, 3, 4, 5, 6, 7, 8, 9] belong to this category. In these schemes, each key is assigned to certain user according to each bit of his/her codeword. The tracer analyzes the keys used in each bit positions, and recover the codeword embedded in the captured decoder. The recovered codeword can trace back to at least one of the traitors.

Some schemes [3, 23, 24, 25] allow public traceability, which means that the tracing action is not limited to the tracing authority and it can be performed by anyone.

When traitors are identified, it is highly desirable to render them useless. We notice that not all traitor tracing schemes support revocation. Many schemes only care about the tracing of traitors, and they do not consider the revocation of traitors. The tracing and revoking abilities are combined in some schemes [19, 16, 23, 24, 9] so the schemes may be more practical.

Some works [28, 11] concentrate on attacks against traitor tracing schemes. Kiayias and Pehlivanoglu [28] described pirate evolution attack against subset-cover [16] based traitor tracing schemes. Pirate evolution attack means that a traitor with a number of keys can build many generations of pirate decoders (such behavior is called pirate evolution) and it is costly for the system to disable them generation by generation. Billet and Phan described a new attack in [11] named "Pirates 2.0" (called public collaboration in this paper)

mainly against traitor tracing schemes based on traceability codes (such as collusion secure codes and IPP codes) and schemes based on subset-cover framework [16]. Their work shows that malicious users can leak certain part of their decryption keys in a public place, so pirate decoders can be built from the public information. Each traitor cannot be identified because many users also hold the same partial keys that are leaked in public.

There are two schemes [29, 30] that are resistant to pirate evolution attack. In scheme of [29], the subsets are partitioned more quickly than [16] so that the colluding traitors can be detected and disabled in 2 generations of media key block. In scheme of [30], wildcarded identity based encryption [31] is used to link the decryption key for each user. Each set of decryption key is no longer separated as in [16], so the key evolution attack to subset-cover based scheme is prevented. There are several schemes [4, 8, 32, 30, 9] that are resistant to public collaboration attacks. In schemes of [4, 8, 30], wildcarded identity based encryption [31] is used to connect all separated parts of each user's decryption key so the decryption key should be leaked as a whole in order for the key to be useful. The traitor will be immediately located if he leaks decryption key as a whole. In scheme of [32, 9], the decryption keys are bound to each user's distinct identity. Thus, leaking any partial decryption key will also identify the traitor immediately.

## 2. Preliminaries

### *2.1. Collusion Secure Codes*

The definition of collusion secure codes required for constructing our tracing and revoking scheme is described, which is similar to that in [5].

- For a codeword $\bar{w} \in \{0,1\}^L$ we write $\bar{w} = \bar{w}_1 \dots \bar{w}_L$, where $\bar{w}_i \in \{0,1\}$ is the $i$th bit of $\bar{w}$ for $i = 1, \dots, L$.
- Let $W = \{\bar{w}^{(1)}, \dots, \bar{w}^{(t)}\}$ be a set of codewords in $\{0,1\}^L$. For a codeword $\bar{w} \in \{0,1\}^L$, if for all $i = 1, \dots, L$ there is a $j \in \{1, \dots, t\}$ satisfying that $\bar{w}_i = \bar{w}_i^{(j)}$, We say that the codeword $\bar{w}$ is feasible for $W$. For example, if $W$ consists of the two words $\binom{0\ 1\ 1\ 0\ 0}{1\ 0\ 1\ 1\ 0}$, then all codewords of the form $[\binom{0}{1}\ \binom{1}{0}\ 1\ \binom{0}{1}\ 0]$ are feasible for $W$.
- For a set of codewords $W \subseteq \{0,1\}^L$ we use $F(W)$ to denote the feasible set of $W$, which is the set containing all codewords that are feasible for $W$.

We use a pair of polynomial time algorithms $(G, T)$ to denote the collusion secure code. They are defined as follows:

- The code generating algorithm $G$ is a probabilistic algorithm that takes a pair $(N, \epsilon)$ as input, where $N$ is the number of codewords to output and $\epsilon \in (0, 1)$ is a security parameter. A pair $(\Gamma, \mathrm{TK})$ is output. Here $\Gamma$ (called a code) contains $N$ codewords in $\{0, 1\}^L$ for some $L > 0$ (called the code length). TK is called the tracing key.

- The tracing algorithm $T$ is a deterministic algorithm that takes a pair $(\bar{w}^*, \mathrm{TK})$ as input, where $\bar{w}^* \in \{0, 1\}^L$. A subset $S$ of $\{1, \ldots, N\}$ is output. Informally, the elements in $S$ are accused of constructing the codeword $\bar{w}^*$.

We define the collusion resistant property of collusion secure code $(G, T)$ using the following game between a challenger and an adversary. Let $N$ be an integer and $\epsilon \in (0, 1)$. Let $C$ be a subset of $\{1, \ldots, N\}$. Both the challenger and adversary are given $(N, \epsilon, C)$ as input. Then the game proceeds as follows:

1. The challenger runs $G(N, \epsilon)$ to obtain $(\Gamma, \mathrm{TK})$ where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$. It sends the set $W := \{\bar{w}^{(i)}\}_{i \in C}$ to the adversary.
2. The adversary outputs a word $\bar{w}^* \in F(W)$.

We say that the adversary $\mathcal{A}$ wins the game if $T(\bar{w}^*, \mathrm{TK})$ is empty or not a subset of $C$. We denote $Adv_{CR}^{\mathcal{A}, G(N,\epsilon), T, C}$ as the advantage that $\mathcal{A}$ wins the collusion resistant game.

A collusion secure code $(G, T)$ is said to be fully collusion resistant if for all polynomial time adversaries $\mathcal{A}$, all $N > 0$, all $\epsilon \in (0, 1)$, and all $C \subseteq \{1, \ldots, N\}$, we have $Adv_{CR}^{\mathcal{A}, G(N,\epsilon), T, C}$ is less than $\epsilon$.

A collusion secure code $(G, T)$ is said to be $t$-collusion resistant if for all polynomial time adversaries $\mathcal{A}$, all $N > t$, all $\epsilon \in (0, 1)$, and all $C \subseteq \{1, \ldots, N\}$ of size at most $t$, we have $Adv_{CR}^{\mathcal{A}, G(N,\epsilon), T, C}$ is less than $\epsilon$.

### *2.2. Bilinear Pairings and Complexity Assumption*

We review the definition of bilinear pairings [33], the decisional $v$-modified bilinear Diffie-Hellman assumption [10] and the decisional $q$-truncated bilinear Diffie-Hellman exponent assumption [13] in brief.

**Bilinear Pairings:** Let $\mathbb{G}$ be a (multiplicative) cyclic group of prime order $p$ and $g$ is a generator of $\mathbb{G}$. A one-way map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear pairing if the following conditions hold.

- Bilinear: For all $u, v \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$, i.e., if $g$ generates $\mathbb{G}$, then $e(g, g)$ generates $\mathbb{G}_T$.
- Computability: There exists an efficient algorithm for computing $e(u, v)$, $\forall u, v \in \mathbb{G}$.

**The decisional $q$-Truncated Bilinear Diffie-Hellman Exponent Assumption:** Given a vector of $2q + 2$ elements ($g'$, $g$, $g^{\alpha}$, $g^{\alpha^2}$, ..., $g^{\alpha^q}$, $Z$) $\in \mathbb{G}^{q+2} \times \mathbb{G}_T$ to decide whether $Z = e(g', g)^{\alpha^{q+1}}$.

We call it the q-TBDHE [13] problem for convenience. Actually, this is the truncated decisional version of q-BDHE problem [34] as described in [13]. The q-BDHE problem can be solved once the q-TBDHE problem is solved whereas not, so the q-TBDHE problem is at least as difficult as the q-BDHE problem. We refer our reader to [13] for detailed description.

*2.3. Protocol Model*

The protocol model consists of five algorithms (Setup, KeyGen, Encrypt, Decrypt, Trace) that are described as follows.

- **Setup**($1^{\lambda}$, $M$, $N$). On inputting $1^{\lambda}$, the maximum number of receivers allowed in each broadcast $M$ and the total number of users in the system $N$, the algorithm outputs the public parameter PK, the private key of the system SK and a secret trace-key TK.
- **KeyGen**($ID_u$, $SK$). On inputting an identity $ID_u$ with system index $u \in \{1, \ldots, N\}$ and the private key of the system SK, the algorithm outputs the corresponding private decryption key $\mathrm{SK}_u$.
- **Encrypt**(PK, $S$, $TSK$). On inputting the public parameter PK, a set of designated receivers $S$ and a temporary session key $TSK$, the algorithm generates a broadcast ciphertext header $Hdr$. The messages are symmetrically encrypted using $TSK$ to generate ciphertext body $C$. We refer to the ciphertext length as the length of $Hdr$ in which a session key is protected and distributed by the tracing and revoking scheme.

- **Decrypt**($\mathrm{SK}_u$, $S$, $Hdr$, PK). On inputting the set of private keys $\mathrm{SK}_u$ of user $u$, the set of designated receivers $S$, a broadcast ciphertext header $Hdr$ and the public parameter PK, the algorithm returns the recovered temporary session key $TSK$ or $\perp$. $TSK$ is used to decrypt the messages from ciphertext body $C$.

- **Trace**$^{\mathcal{D}}$(TK). Given a captured pirate decoder $\mathcal{D}$ and private trace-key TK, the algorithm queries $\mathcal{D}$ as a black box and then outputs a set of traitors $T \subseteq \{1, \ldots, N\}$.

*2.4. Security Requirements*

- **Correctness**. If a user is included in the set of designated receivers, he/she can recover the message from the ciphertext if the sender and the receiver operate honestly.

- **Semantic Security**. Any users that are not included in the set of designated receivers, cannot obtain any information of messages from in the ciphertext.

  We define the IND-ID-CCA2 security (security against chosen ciphertext attack under adaptive-adversary model) of our tracing and revoking system using the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$.

  - **Setup**. $\mathcal{B}$ runs $G(N, \epsilon)$ to obtain ($\Gamma$, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword for user with system index $i$. The challenger also generates public parameters PK. It sends $\Gamma$ and PK to the adversary.
  - **Phase 1**. The adversary $\mathcal{A}$ adaptively issues key generation query on $ID_i$ and decryption query on $(ID_i, S, Hdr)$.
  - **Challenge**. $\mathcal{A}$ sends $(S^*, K_0, K_1)$ to $\mathcal{B}$, where $S^* \subseteq \mathbb{U}$ and the identities of $S^*$ have never been queried the private keys in Phase 1. The challenger randomly chooses $\beta \in \{0, 1\}$ and runs algorithm Encrypt to obtain $(Hdr^*, K_\beta)$. It then gives $Hdr^*$ to adversary $\mathcal{A}$.
  - **Phase 2**. $\mathcal{A}$ issues additional key generation query $ID_i$, where $ID_i \notin S^*$ and decryption query $Hdr \neq Hdr^*$, for any identity of $S^*$. In both cases, $\mathcal{A}$ responds as in Phase 1. These queries may be adaptive.

– **Guess**. Finally, $\mathcal{A}$ outputs a guess $\beta' \in \{0,1\}$ and wins if $\beta = \beta'$.

The advantage of the adversary $\mathcal{A}$ in the above game is defined as $|Pr[\beta = \beta'] - 1/2|$.

**Definition 1.** The tracing and revoking system is ($t$, $\epsilon$, $q_k$, $q_d$) IND-ID-CCA2 secure if all $t$-time IND-ID-CCA2 adversaries making at most $q_k$ key generation queries and at most $q_d$ decryption queries have advantage at most $\epsilon$ in winning the IND-ID-CCA2 security game.

- **Collusion Resistant**. The collusion resistant property of the proposed scheme is defined using the following game between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$. Let $(G, T)$ be a collusion secure code. Let $N$ be an integer and $\epsilon \in (0,1)$. Then the game proceeds as follows:
  1. $\mathcal{B}$ runs $G(N, \epsilon)$ to obtain ($\Gamma$, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword for user $i$. $\mathcal{B}$ also generates public parameters PK. It sends $\Gamma$ and PK to $\mathcal{A}$.
  2. $\mathcal{A}$ selects a set of identities $S^*$ with the corresponding subset of system indexes denoted as $C$ ($C \subset \{1, \ldots, N\}$). $\mathcal{A}$ can query $\mathcal{B}$ for decryption keys of the users in $C$. $\mathcal{B}$ generates the keys and gives them to $\mathcal{A}$.
  3. $\mathcal{B}$ selects a set of system indexed $S \subseteq \{1, \ldots, N\}$ for which $\mathcal{A}$ can recover the messages, and asks $\mathcal{A}$ to decrypt ciphertexts for $S$ a number of times. Finally the challenger recovers a codeword $\bar{w}^*$.

  We say that $\mathcal{A}$ wins the game if the output of $T(\bar{w}^*, \text{TK})$ is empty or not a subset of $C \cap S$.

- **Public Collaboration Resistant**. It is required by the property of public collaboration resistance that if any user releases any partial key in public, the identity of the corresponding user will be surely detected though we cannot prevent him from giving out his decryption key.

## 3. CCA2 Secure Scheme Under Adaptive-Adversary Model without Random Oracle

We describe our construction of codes based tracing and revoking scheme in this section, by embedding codeword into the decryption key of IBBE

scheme by Boneh and Hamburg [12] (similar IBBE scheme can also be found in [35]). The scheme is proved secure against chosen ciphertext attacks under adaptive-adversary model. The scheme achieves O($m$) public keys size, O($mL$) private keys size, and O(1) ciphertext length, where $L$ is the length of codeword and $m$ is the maximum number of receivers allowed in each broadcast. We describe construction idea firstly and then the details of the new scheme.

### *3.1. Construction Idea*

**Adaptive-Adversary Security**. Adaptive-adversary security (also called full security) means that the simulator in the IND-ID-CCA2 game can freely answer decryption queries, key generation queries and challenge query, and does not require the adversary to submit its target identities at the beginning of the game.

By following the idea of Gentry [13], our scheme achieves a tight reduction in security proof. Suppose we are given an $M$-TBDHE problem, i.e. given a vector $(g', g, g^{\alpha}, \ldots, g^{\alpha^M}, Z) \in \mathbb{G}^{M+2} \times \mathbb{G}_T$ to decide whether $Z = e(g', g)^{\alpha^{M+1}}$. We allow the adversary to query at most $M$-1 times. The decryption key structure of IBBE scheme in [12] is modified to add new parameters ($g_2$, $g_3$ and $f(x)$). In the simulation, we select $2L+2$ random polynomials $f_{i,b}(x) = a_M x^M + \sum_{j=0}^{M-1} a_{i,b,j} x^j$, $f_2(x) = \sum_{i=0}^{M} b_i x^i$, $f_3(x) = \sum_{i=0}^{M} c_i x^i$ from $\mathbb{Z}_p^*[x]$ of degree $M$ where $i = 1, \ldots, L$ and $b = 0, 1$. The public elements $w_{i,b}, g_2, g_3$ are defined as $w_{i,b} = g^{f_{i,b}(\alpha)}$ ($\forall i = 1, \ldots, L$ and $b = 0, 1$), $g_2 = g^{f_2(\alpha)}$, $g_3 = g^{f_3(\alpha)}$ that can be calculated from $\{g^{\alpha^i}, i = 0, ..., M\}$. We set $f(x) = -\frac{b_M}{c_M} x - \frac{a_M}{c_M}$ so the equation $a_M + t_{ID} b_M + f(t_{ID}) c_M = 0$ can help to bring down the exponent of the key element $(w g_2^{t_{ID}} g_3^{f(t_{ID})})^{\alpha}$ to a polynomial of degree $M$, then we can compute it from $\{g^{\alpha^i}, i = 0, ..., M\}$. When generating challenge ciphertext, what the challenger needs to compute are polynomials of degree $M$ (also can be computed from $\{g^{\alpha^i}, i = 0, ..., M\}$), with the help of $Z$. If $Z = e(g', g)^{\alpha^{M+1}}$, the challenging ciphertext is perfectly simulated, and the simulator has the same advantage in making the right decision to $M$-TBDHE problem as the adversary in making the right choice to the security game. If $Z$ is uniformly random, the ciphertext header is uniformly random and is independent from the adversary's view. The random ciphertext header does not help the adversary to guess the bit choice. Thus, we obtain a tight reduction for the proof of adaptive-adversary security. Our readers can refer to [13] for more details of the skill.

**CCA2 Security**. Papers [36, 37] presented two methods for ensuring security against chosen-ciphertext attacks. In our scheme, we employ the technique of [37] to achieve security against adaptive chosen-ciphertext attacks (CCA2 security). As described in [37], we need a secure encapsulation scheme as ($\mathsf{Init}$, $\mathsf{S}_{encap}$, $\mathsf{R}_{encap}$) and a strong one-time message authentication code as ($\mathsf{Mac}$, $\mathsf{Vrfy}$). $\mathsf{Init}(1^\lambda)$ is called to generate $\mathsf{pub}$. When in encryption, $\mathsf{S}_{encap}(1^\lambda$, $\mathsf{pub})$ is called to obtain a new random set of ($r$, $\mathsf{com}$, $\mathsf{dec}$), and ($m \circ \mathsf{dec}$) is encrypted to the new set of receivers $S\cup$ "com" to obtain the ciphertext $C$. We use ($m \circ \mathsf{dec}$) to denote concatenating $\mathsf{dec}$ to the message $m$. By calling $\mathsf{Mac}$ with the input $r$, the sender generates a message authentication code $\mathsf{tag}$ for $C$, which can be denoted as $\mathsf{tag} \leftarrow \mathsf{Mac}_r(C)$. Finally, the ciphertext of the form $\langle \mathsf{com}, C, \mathsf{tag} \rangle$ is sent. Each receiver uses his decryption key to recover ($m \circ \mathsf{dec}$), and calls $\mathsf{R}_{encap}(\mathsf{pub}, \mathsf{com}, \mathsf{dec})$ to recover $r$. If $\mathsf{Vrfy}_r(C, \mathsf{tag})=1$, the receiver outputs $m$. The related proof for CCA2 security in Theorem 1 is omitted for brevity.

### *3.2. Detailed Scheme*

The proposed tracing and revoking scheme works as follows:

**Setup**($1^\lambda$, $M$, $N$). Given security parameter $1^\lambda$, the maximal number of receivers $M$ in each broadcast and the total number of users supported in the system $N$, PKG selects $\epsilon \in (0, 1)$ and runs collusion secure code generation algorithm $\mathbf{G}(N, \epsilon)$ to generate a pair ($\Gamma$, TK). The set $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ contains $N$ codewords in $\{0, 1\}^L$, where $L$ is the codeword length. $TK$ is the tracing key for $\Gamma$. $\bar{w}^{(u)}$ is assigned to user with system index $u$, with $1 \leq u \leq N$. PKG also selects $\mathbb{G}$, $\mathbb{G}_T$ and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ as defined in Section 2, and $g$ is a generator of $\mathbb{G}$. $g_1 = g^\alpha$, where $\alpha \in_R \mathbb{Z}_p^*$. $p$ is a prime larger than $2^\lambda$. $H : \mathbb{G}_T \rightarrow \mathbb{G}_T \times \mathbb{Z}_p^*$ is a collision-resistant hash function. PKG randomly chooses $g_2$, $g_3$, $w_{1,1}$, $w_{1,0}$, $\ldots$, $w_{L,1}$, $w_{L,0}$, $z \in \mathbb{G}$, and $f(x) = ax + b$, where $a, b \in \mathbb{Z}_p^*$. If $g_2 = g_3^{-a}$ or $w_{i,b} = g_3^{-b}$ for any $i \in \{1, \ldots, L\}$ and $b \in \{1, 0\}$, chooses another $f(x)$ again. PKG also chooses a $(M + 1)$-vector $h = (h_0, h_1, \ldots, h_M) \leftarrow_R \mathbb{G}^{M+1}$ of random generators so that $h_i = g^{v_i}$ for $i = 0, \ldots, M$ with a randomly chosen $v = (v_0, \ldots, v_M) \leftarrow_R \mathbb{Z}_p^{M+1}$. $\mathsf{Init}(1^\lambda)$ is run to obtain $\mathsf{pub}$ for encapsulation scheme. The public parameters $MPK$ are ($\mathbb{G}$, $\mathbb{G}_T$, $p$, $g$, $g_1$, $g_2$, $g_3$, $w_{1,1}$, $w_{1,0}$, $\ldots$, $w_{L,1}$, $w_{L,0}$, $z$, $f(x)$, $h$, $H$, $\mathsf{pub}$), and PKG keeps $(v, \alpha)$ as the private key $MSK$.

**KeyGen**($ID$, $MSK$). When a user with identity $ID$ joins the system, PKG assigns to him an unused system index $u \in \{1, \ldots, N\}$ and the codeword $\bar{w}^{(u)}$. Then, PKG picks random $r_1, \ldots, r_L, t_{1,ID}, \ldots, t_{L,ID} \in \mathbb{Z}_p^*$ and computes

$$\begin{aligned} K_{ID,i} &= (K_{i,1}, K_{i,2}, T_{i,0}, \ldots, T_{i,M-1}, t_{i,ID}) \\ &= ((w_{i,b} \cdot g_2^{t_{i,ID}} g_3^{f(t_{i,ID})})^{\alpha} z^{r_i}, g^{r_i}, h_1^{r_i} h_0^{-IDr_i}, \\ &\quad h_2^{r_i} h_1^{-IDr_i}, \ldots, h_M^{r_i} h_{M-1}^{-IDr_i}, t_{i,ID}), \end{aligned}$$

for all $i \in \{1, \ldots, L\}$ and $b = \bar{w}_i^{(u)}$. $(K_{ID,1}, \ldots, K_{ID,L})$ are sent to the user via a secure channel. $(ID, u, \bar{w}^{(u)}, K_{ID,1}, \ldots, K_{ID,L})$ are recorded. Then User checks whether $K_{ID,i}$ satisfies the following relations for each $i \in \{1, \ldots, L\}$, $b = \bar{w}_i^{(u)}$ and $j \in \{0, \ldots, M-1\}$:

$$\begin{aligned} e(K_{i,1}, g) &= e(w_{i,b}, g_1) \cdot e(g_1, g_2)^{t_{i,ID}} \\ &\quad \cdot e(g_1, g_3)^{f(t_{i,ID})} \cdot e(z, K_{i,2}), \\ e(g, T_{i,j}) &= e(K_{i,2}, h_{j+1} h_j^{-ID}). \end{aligned}$$

**Encrypt**($MPK$, $S$, $TSK$). to encrypt a temporary session key $TSK \in \mathbb{G}_T$, the sender

1. selects a tracing position $j \in \{1, \ldots, L\}$;

2. runs $\mathsf{S}_{encap}(1^{\lambda}, \mathsf{pub})$ to obtain a new random set as $(r, \mathsf{com}, \mathsf{dec})$;

3. For the receiver set $S = \{ID_1, \ldots, ID_{n-1}\}$ where $n \leq M$, the sender sets $S' = S \cup \{\mathsf{com}\}$ and expands $P(X) \in \mathbb{Z}_p[X]$ as

$$\begin{aligned} P(X) &= \prod_{ID_i \in S'} (X - ID_i) \\ &= \rho_n X^n + \rho_{n-1} X^{n-1} + \cdots + \rho_1 X + \rho_0; \end{aligned}$$

4. chooses $s_1, s_0 \in_R \mathbb{Z}_p^*$ and computes

$$
\begin{aligned}
C_1 &= \langle C_{1,0}, C_{1,1}, C_{1,2}, C_{1,3}, C_{1,4} \rangle \\
&= \langle (TSK \circ \mathsf{dec}) \oplus H(e(g_1, w_{j,1})^{s_1}), \\
&\quad g^{s_1}, (z \cdot h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^{s_1}, \\
&\quad e(g_1, g_2)^{s_1}, e(g_1, g_3)^{s_1} \rangle; \\
C_0 &= \langle C_{0,0}, C_{0,1}, C_{0,2}, C_{0,3}, C_{0,4} \rangle \\
&= \langle (TSK \circ \mathsf{dec}) \oplus H(e(g_1, w_{j,0})^{s_0}), \\
&\quad g^{s_0}, (z \cdot h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^{s_0}, \\
&\quad e(g_1, g_2)^{s_0}, e(g_1, g_3)^{s_0} \rangle;
\end{aligned}
$$

5. computes $\mathsf{tag} = \mathsf{Mac}_r(C_1, C_0)$.

$C_1$ is the part of ciphertext for codeword bit 1 in tracing position $j$. We denote the computations of $C_1$ as $C_1 \leftarrow BEnc(j, 1, S, TSK, r, \mathsf{com}, \mathsf{dec})$, which means encrypting temporary session key $TSK$ to the receiver set $S$ using tracing position $j$, codeword bit 1 and additional parameters ($r$, com, dec). $C_0$ is the part of ciphertext for codeword bit 0 in tracing position $j$. Likewise, the computations of $C_0$ can be denoted as $C_0 \leftarrow BEnc(j, 0, S, TSK, r, \mathsf{com}, \mathsf{dec})$. $\langle j, \mathsf{com}, C_1, C_0, \mathsf{tag} \rangle$ will be broadcast to all users as the ciphertext header $Hdr$, and $TSK$ will be used as session key to symmetrically encrypt message to generate the ciphertext body.

**Decrypt**($\mathrm{SK}_u$, $S$, $Hdr$, MPK). User with the identity $ID_u \in S$ parses header $Hdr$ as $\langle j, \mathsf{com}, C_1, C_0, \mathsf{tag} \rangle$. For tracing position $j$, codeword bit $b = \bar{w}_j^{(u)}$ and the private key $K_{ID_u,j} = (K_{j,1}, K_{j,2}, T_{j,0}, \ldots, T_{j,M-1}, t_{j,ID_u})$, the user

1. parses $C_b$ as $(C_{b,0}, C_{b,1}, C_{b,2}, C_{b,3}, C_{b,4})$;

2. sets $S' = S \cup \{\mathsf{com}\}$ and expands $P_{ID_u}(X) \in \mathbb{Z}_p[X]$ as

$$
\begin{aligned}
P_{ID_u}(X) &= \prod_{ID_j \in S' \setminus \{ID_u\}} (X - ID_j) \\
&= y_{n-1}^{(ID_u)} X^{n-1} + y_{n-2}^{(ID_u)} X^{n-2} + \ldots \\
&\quad + y_1^{(ID_u)} X + y_0^{(ID_u)}
\end{aligned}
$$

and computes the decryption key as

$$\begin{aligned} D_{ID_u} &= K_{j,1} \cdot T_{j,0}^{y_0^{(ID_u)}} \cdot T_{j,1}^{y_1^{(ID_u)}} \ldots T_{j,n-1}^{y_{n-1}^{(ID_u)}} \\ &= (w_{j,b} \cdot g_2^{t_{j,ID_u}} \cdot g_3^{f(t_{j,ID_u})})^{\alpha} \\ &\quad \cdot (z \cdot h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^r, \\ d_{ID_u} &= K_{j,2} = g^r, \\ t_{ID_u} &= t_{j,ID_u}; \end{aligned}$$

since the following equations hold

$$\begin{aligned} \rho_n &= y_{n-1}^{(ID_u)}; \\ \rho_{n-1} &= y_{n-2}^{(ID_u)} - ID_u y_{n-1}^{(ID_u)}; \\ &\ldots \\ \rho_1 &= y_0^{(ID_u)} - ID_u y_1^{(ID_u)}; \\ \rho_0 &= -ID_u y_0^{(ID_u)}. \end{aligned}$$

3. then he recovers the session key as follows. He sets $t = t_{j,ID_u}$ and calculates $e(g_1, w_{j,b})^s$ as

$$\begin{aligned} & \frac{e(C_{b,1}, D_{ID_u})}{e(C_{b,2}, d_{ID_u}) \cdot C_{b,3}^t \cdot C_{b,4}^{f(t)}} \\ = & \frac{e(g^s, (w_{j,b} \cdot g_2^t g_3^{f(t)})^{\alpha} (z h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^r)}{e((z h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^s, g^r) e(g_1, g_2)^{st} e(g_1, g_3)^{sf(t)}} \\ = & e(g^s, w_{j,b}^{\alpha}). \end{aligned}$$

He recovers the temporary session key as $(TSK \circ \mathsf{dec}) = C_{b,0} \oplus H(e(g_1, w_{j,b})^s)$;

4. runs $\mathsf{R}_{encap}(\mathsf{pub}, \mathsf{com}, \mathsf{dec})$ to obtain $r$, and checks whether $\mathsf{Vrfy}_r(C_1, C_0, \mathsf{tag})$=1.

If holds, the user outputs $TSK$ as temporary session key. Else, $\perp$ is output.

**Trace**$^{\mathcal{D}}$(TK). Given a perfect pirate decoder $\mathcal{D}$ with complete decryption keys, the trusted party queries decoder $\mathcal{D}$ as a black-box oracle. We say the

decoder is perfect if it will always decrypt well-formed ciphertext correctly [5]. We assume that the captured decoder can recover the messages encrypted for receivers set $S$. For $j = 1, \ldots, L$, the sender selects a random temporary session key $TSK \in_R \mathbb{G}_T$, another key $K_R \in_R \mathbb{G}_T$ ($K_R \neq TSK$). The sender runs $\mathsf{S}_{encap}(1^\lambda, \mathsf{pub})$ to obtain a new random set as $(r, \mathsf{com}, \mathsf{dec})$ and computes

$$\begin{aligned} C_1 &\leftarrow BEnc(j, 1, S, TSK, r, \mathsf{com}, \mathsf{dec}), \\ C_0 &\leftarrow BEnc(j, 0, S, K_R, r, \mathsf{com}, \mathsf{dec}), \\ \mathsf{tag} &= \mathsf{Mac}_r(C_1, C_0) \end{aligned}$$

as described in the algorithm Encrypt. The ciphertext header $Hdr$ is $\langle j, \mathsf{com}, C_1, C_0, \mathsf{tag}\rangle$. A message $m$ is symmetrically encrypted using the session key $TSK$ to generate ciphertext body $C$. $(Hdr, S)$ and $C$ are fed to the decoder as in normal broadcasting. If the pirate decoder outputs $m$, the trusted party decides that the decoder contains a codeword $\bar{w}^*$ with $\bar{w}_j^* = 1$. Otherwise, $\bar{w}_j^* = 0$. When the tracing on all positions $(1, \ldots, L)$ is completed, the recovered codeword $\bar{w}^* = (\bar{w}_1^* \ \ldots \ \bar{w}_L^*)$ is put to the tracing algorithm of collusion secure code $\mathbf{T}(\bar{w}^*, \mathrm{TK})$ to obtain a set of traitors $T \cap S \subseteq \{1, \ldots, N\}$, with $T = \mathbf{T}(\bar{w}^*, \mathrm{TK})$.

*3.3. Security Analysis*

**Correctness.** The correctness is straightforward due to the above computations.

**Theorem 1.** *If there is ($t$, $\epsilon$, $q_k$, $q_d$)-adversary against IND-ID-CCA2 security game of the proposed scheme, the M-TBDHE assumption can be broken with probability $\epsilon'$ within time $t'$, where $q_k + q_d < M$, $t' \approx t + O(M(q_k+q_d)T_{exp}) + O(M(q_k+q_d)T_{mul}) + O(q_d T_{pair})$, $\epsilon' \approx \epsilon$, $T_{exp}$, $T_{pair}$ and $T_{mul}$ are the average time for exponentiation, pairing and multiplication in $\mathbb{G}$ respectively.*

**Proof:** Suppose such adversary exists, we denote it as $\mathcal{A}$. Challenger $\mathcal{B}$ is given an M-TBDHE problem, i.e. $\mathcal{B}$ is given a vector $(g', g, g^\alpha, \ldots, g^{\alpha^M}, Z) \in \mathbb{G}^{M+2} \times \mathbb{G}_T$ to decide whether $Z = e(g', g)^{\alpha^{M+1}}$.

**Setup.** $\mathcal{B}$ selects random functions $f_{1,1}(x)$, $f_2(x)$, $f_3(x) \in (\mathbb{Z}_p^*)[x]$ of degree $M$, where $f_{1,1}(x) = a_M x^M + \sum_{j=0}^{M-1} a_{1,1,j} x^j$, $f_2(x) = \sum_{j=0}^{M} b_j x^j$, $f_3(x) = \sum_{j=0}^{M} c_j x^j$. Let $g_1 = g^\alpha$, $w_{1,1} = g^{f_{1,1}(\alpha)}$, $g_2 = g^{f_2(\alpha)}$, $g_3 = g^{f_3(\alpha)}$, $f(x) = -\frac{b_M}{c_M} x - \frac{a_M}{c_M}$. If $g_2 = g_3^{b_M/c_M}$ or $w_{1,1} = g_3^{a_M/c_M}$, randomly chooses $f_{1,1}(x)$,

$f_2(x)$, $f_3(x)$ again. After that, additional random functions $f_{1,0}(x)$, $f_{2,1}(x)$, $f_{2,0}(x)$, ..., $f_{L,1}(x)$, $f_{L,0}(x) \in (\mathbb{Z}_p^*)[x]$ of degree $M$ are selected, ensuring they are of the form $f_{i,k}(x) = a_M x^M + \sum_{j=0}^{M-1} a_{i,k,j} x^j$, for all $i = 1, \ldots, L$ and $k = 0, 1$. Let $w_{i,k} = g^{f_{i,k}(\alpha)}$, for all $i = 1, \ldots, L$ and $k = 0, 1$. If $w_{i,k} = g_3^{a_M/c_M}$ holds for some $(i, k)$, $\mathcal{B}$ chooses $f_{i,k}(x)$ again. $\mathcal{B}$ also chooses a $(M+1)$-vector $h = (h_0, h_1, \ldots, h_M)$, a collision-resistant hash function $H$ as described in the proposed scheme. $\mathcal{B}$ keeps the $h$-related vector $v = (v_0, \ldots, v_M)$ private. $\mathcal{B}$ also selects random $\gamma \in \mathbb{Z}_p^*$, sets $z = g^\gamma$. $\mathsf{Init}(1^\lambda)$ is run to obtain $\mathsf{pub}$ for encapsulation scheme. Then $\mathcal{B}$ sends the public parameters ($\mathbb{G}$, $\mathbb{G}_T$, $p$, $g$, $g_1$, $g_2$, $g_3$, $w_{1,1}$, $w_{1,0}$, ..., $w_{L,1}$, $w_{L,0}$, $z$, $f(x)$, $h$, $H$, $\mathsf{pub}$) to $\mathcal{A}$.

**Phase 1.** $\mathcal{A}$ is free to query as follows.
**KeyGen query.** $\mathcal{A}$ may request the KeyGen algorithm run for an arbitrary identity $ID$. $\mathcal{B}$ assigns to $ID$ an unused system index $u \in \{1, \ldots, N\}$ and the codeword $\bar{w}^{(u)}$. Then, $\mathcal{B}$ picks random $r_1$, ..., $r_L$, $t_{1,ID}$, ..., $t_{L,ID} \in \mathbb{Z}_p^*$ satisfying $w_{i,k} \cdot g_2^{t_{i,ID}} g_3^{f(t_{i,ID})} \neq 1$ for $i = 1, \ldots, L$ and $k = \bar{w}_i^{(u)}$. $\mathcal{B}$ computes

$$\begin{aligned} K_{ID,i} &= (K_{i,1}, K_{i,2}, T_{i,0}, \ldots, T_{i,M-1}, t_{i,ID}) \\ &= (g^{\sum_{j=0}^{M-1}(a_{i,k,j}+t_{i,ID}b_j+f(t_{i,ID})c_j)\alpha^{j+1}} z^{r_i}, \\ &\quad g^{r_i}, h_1^{r_i} h_0^{-ID \cdot r_i}, h_2^{r_i} h_1^{-ID \cdot r_i}, \ldots, \\ &\quad h_M^{r_i} h_{M-1}^{-ID \cdot r_i}, t_{i,ID}), \end{aligned}$$

for all $i \in \{1, \ldots, L\}$ and $k = \bar{w}_i^{(u)}$. Because $f(t_{i,ID}) = -\frac{b_M}{c_M} t_{i,ID} - \frac{a_M}{c_M}$, so we have $a_M + t_{i,ID} b_M + f(t_{i,ID}) c_M = 0$. Thus

$$\begin{aligned} K_{i,1} &= (g^{\sum_{j=0}^{M-1}(a_{i,k,j}+t_{i,ID}b_j+f(t_{i,ID})c_j)\alpha^{j+1}}) \cdot z^{r_i} \\ &= g^{\sum_{j=0}^{M-1}(a_{i,k,j}+t_{i,ID}b_j+f(t_{i,ID})c_j)\alpha^{j+1}} \\ &\quad \cdot g^{(a_M+t_{i,ID}b_M+f(t_{i,ID})c_M)\alpha^{j+1}} \cdot z^{r_i} \\ &= (g^{a_M \alpha^M + \sum_{j=0}^{M-1} a_{i,k,j}\alpha^j} \cdot g^{t_{i,ID} \sum_{j=0}^{M} b_j \alpha^j} \\ &\quad \cdot g^{f(t_{i,ID}) \sum_{j=0}^{M} c_j \alpha^j})^\alpha \cdot z^{r_i} \\ &= (g^{f_{i,k}(\alpha)} \cdot g^{t_{i,ID} f_2(\alpha)} \cdot g^{f(t_{i,ID}) f_3(\alpha)})^\alpha \cdot z^{r_i} \\ &= (w_{i,k} \cdot g_2^{t_{i,ID}} g_3^{f(t_{i,ID})})^\alpha \cdot z^{r_i}. \end{aligned}$$

$K_{i,1}$ has the correct form and $K_{ID,i}$ is a valid private key for $ID$ and $\bar{w}_i^{(u)}$ on codeword position $i$. Also, $K_{ID,i}$ is randomly distributed because of the randomness of $t_{i,ID}$ and $r_i$. $(K_{ID,1}, \ldots, K_{ID,L})$ are returned to $\mathcal{A}$. $(ID, u, \bar{w}^{(u)}, K_{ID,1}, \ldots, K_{ID,L})$

are recorded by $\mathcal{B}$.

**Decryption query.** $\mathcal{A}$ sends $(ID, S, Hdr)$ to $\mathcal{B}$. $\mathcal{B}$ checks whether $ID \in S$ holds. If so, $\mathcal{B}$ obtains private key $K_{ID}$ for $ID$ as follows: $\mathcal{B}$ finds the system index $u$ of $ID$, the codeword $\bar{w}^{(u)}$ and the decryption key in the records. If no such item exists, $\mathcal{B}$ assigns to $ID$ an unused system index $u \in \{1, \ldots, N\}$ and the codeword $\bar{w}^{(u)}$. $\mathcal{B}$ chooses randomly $r_1, \ldots, r_L, t_{1,ID}, \ldots, t_{L,ID} \in \mathbb{Z}_p^*$ so that $w_{i,k} \cdot g_2^{t_{i,ID}} g_3^{f(t_{i,ID})} \neq 1$ for $i = 1, \ldots, L$ and $k = \bar{w}_i^{(u)}$. $\mathcal{B}$ sets $t_i = t_{i,ID}$ and computes

$$\begin{aligned}
K_{ID,i} &= (K_{i,1}, K_{i,2}, T_{i,0}, \ldots, T_{i,M-1}, t_i) \\
&= (g^{\sum_{j=0}^{M-1}(a_{i,k,j}+t_i b_j + f(t_i)c_j)\alpha^{j+1}} \cdot z^{r_i}, \\
&\quad g^{r_i}, h_1^{r_i} h_0^{-ID \cdot r_i}, h_2^{r_i}\ h_1^{-ID \cdot r_i}, \ldots, \\
&\quad h_M^{r_i} h_{M-1}^{-ID \cdot r_i}, t_i), \\
&= ((w_{i,k} \cdot g_2^{t_i} g_3^{f(t_i)})^{\alpha} \cdot z^{r_i}, g^{r_i}, \\
&\quad h_1^{r_i} \cdot h_0^{-ID \cdot r_i}, h_2^{r_i} \cdot h_1^{-ID \cdot r_i}, \ldots, \\
&\quad h_M^{r_i} \cdot h_{M-1}^{-ID \cdot r_i}, t_i)
\end{aligned}$$

for all $i \in \{1, \ldots, L\}$ and $k = \bar{w}_i^{(u)}$. $(ID, u, \bar{w}^{(u)}, K_{ID,1}, \ldots, K_{ID,L})$ are recorded by $\mathcal{B}$. Then $\mathcal{B}$ runs the Decrypt algorithm to verify the tag and recover the message, which is returned to $\mathcal{A}$. If verification fails, $\mathcal{B}$ returns $\perp$.

**Challenge.** $\mathcal{A}$ sends $(S^*, K_0, K_1)$ to $\mathcal{B}$, where $S^* = \{ID_1, \ldots, ID_{n-1}\}$ $(n \leq M)$ and any identity in $S^*$ has never been queried for the private key in Phase 1. $\mathcal{B}$ obtains a new random set of $(r, \mathsf{com}, \mathsf{dec})$, and set the receiver set as $S^* = S^* \cup \{\mathsf{com}\}$. Then he expands $P(X)$ as follows,

$$\begin{aligned}
P(X) &= \prod_{ID_i \in S^*} (X - ID_i) \\
&= \rho_n X^n + \rho_{n-1} X^{n-1} + \cdots + \rho_1 X + \rho_0.
\end{aligned}$$

$\mathcal{B}$ randomly chooses $K_\beta$, $\beta \in \{0, 1\}$, tracing position $i$, random $s \in \mathbb{Z}_p$ and computes

$$
\begin{aligned}
C^*_{1,0} &= (K_\beta \circ \mathsf{dec}) \oplus H(Z^{a_M} e(g', g)^{\sum_{j=0}^{M-1} a_{i,1,j}\alpha^{j+1}}),\\
C^*_{1,1} &= g',\\
C^*_{1,2} &= (g')^{\gamma+\sum_{j=0}^{n} v_j \rho_j},\\
C^*_{1,3} &= Z^{b_M} \cdot e(g', g)^{\sum_{j=0}^{M-1} b_j \alpha^{j+1}},\\
C^*_{1,4} &= Z^{c_M} \cdot e(g', g)^{\sum_{j=0}^{M-1} c_j \alpha^{j+1}};\\
C^*_{0,0} &= (K_\beta \circ \mathsf{dec}) \oplus H((Z^{a_M} e(g', g)^{\sum_{j=0}^{M-1} a_{i,0,j}\alpha^{j+1}})^s),\\
C^*_{0,1} &= (g')^s,\\
C^*_{0,2} &= ((g')^{\gamma+\sum_{i=0}^{n} v_i \rho_i})^s,\\
C^*_{0,3} &= (Z^{b_M} \cdot e(g', g)^{\sum_{i=0}^{M-1} b_i \alpha^{i+1}})^s,\\
C^*_{0,4} &= (Z^{c_M} \cdot e(g', g)^{\sum_{i=0}^{M-1} c_i \alpha^{i+1}})^s.
\end{aligned}
$$

$\mathcal{B}$ sets $C^* = (C^*_{1,0}, C^*_{1,1}, C^*_{1,2}, C^*_{1,3}, C^*_{1,4})$, $C^*_0 = (C^*_{0,0}, C^*_{0,1}, C^*_{0,2}, C^*_{0,3}, C^*_{0,4})$ and calculates $\mathsf{tag} = \mathsf{Mac}_r(C^*_1, C^*_0)$. For any $ID_u \in S^*$ with codeword bit $b = \bar{w}_i^{(u)}$, the private key $K_{ID_u,i} = (K_{i,1},\ K_{i,2},\ T_{i,0},\ \ldots,\ T_{i,M-1}, t_{i,ID_u})$ for tracing position $i$, and the polynomial

$$
\begin{aligned}
P_{ID_u}(X) &= \prod_{ID_j \in S^* \setminus \{ID_u\}} (X - ID_j)\\
&= y_{n-1}^{(ID_u)} X^{n-1} + y_{n-2}^{(ID_u)} X^{n-2} + \ldots\\
&\quad + y_1^{(ID_u)} X + y_0^{(ID_u)},
\end{aligned}
$$

we notice that if $b = 1$,

$$
\begin{aligned}
&\frac{e(C^*_{1,1}, K_{i,1}(T_{i,0})^{y_0^{(ID_u)}} (T_{i,1})^{y_1^{(ID_u)}} \ldots (T_{i,n-1})^{y_{n-1}^{(ID_u)}})}{e(C^*_{1,2}, K_{i,2}) \cdot (C^*_{1,3})^{t_{i,ID_u}} \cdot (C^*_{1,4})^{f(t_{i,ID_u})}}\\
&= Z^{a_M} \cdot e(g', g)^{\sum_{j=0}^{M-1} a_{i,1,j}\alpha^{j+1}}.
\end{aligned}
$$

It does not help $\mathcal{B}$ to decide whether $Z = e(g', g)^{\alpha^{M+1}}$ even $\mathcal{B}$ can generate decryption keys for $ID_u \in S^*$. We can observe the following deductions.

Let $s' = log_g g'$. If $Z = e(g', g)^{\alpha^{M+1}}$, we have

$$
\begin{aligned}
C^*_{1,0} &= (K_\beta \circ \mathsf{dec}) \\
&\quad \oplus H(e(g', g)^{a_M \alpha^{M+1} + \sum_{j=0}^{M-1} a_{i,1,j} \alpha^{j+1}}) \\
&= (K_\beta \circ \mathsf{dec}) \oplus H(e(g', g^{f_{1,1}(\alpha)})^\alpha) \\
&= (K_\beta \circ \mathsf{dec}) \oplus H(e(g_1, w_{1,1})^{s'}), \\
C^*_{1,1} &= g' = g^{s'}, \\
C^*_{1,2} &= (g')^{\gamma + \sum_{j=0}^{n} v_j \rho_j} = (z \cdot h_0^{\rho_0} h_1^{\rho_1} \ldots h_n^{\rho_n})^{s'}, \\
C^*_{1,3} &= Z^{b_M} \cdot e(g', g)^{\sum_{j=0}^{M-1} b_j \alpha^{j+1}} = e(g', g^{f_2(\alpha)})^\alpha \\
&= e(g_1, g_2)^{s'}, \\
C^*_{1,4} &= Z^{c_M} \cdot e(g', g)^{\sum_{j=0}^{M-1} c_j \alpha^{j+1}} = e(g', g^{f_3(\alpha)})^\alpha \\
&= e(g_1, g_3)^{s'}.
\end{aligned}
$$

$C^*_1$ is the valid ciphertext header part encrypting $K_\beta$ for tracing position $i$ with codeword bit value 1 under the randomness of $s'$. It is similar for the case of $b = 0$ under the randomness of $ss'$. Let $\mathsf{tag} = \mathsf{Mac}_r(C_1, C_0)$. $Hdr = \langle j, \mathsf{com}, C_1, C_0, \mathsf{tag} \rangle$ is the valid ciphertext header encrypting $K_\beta$ for tracing position $i$. If $Z$ is uniformly random, the ciphertext is uniformly random and independent from the adversary's view. Thus the ciphertext does not help the adversary to guess the bit choice.

**Phase 2.** $\mathcal{A}$ may query adaptively as Phase 1, with some exceptions as follows.

In KeyGen query on $ID_i$, we require $ID_i \notin S^*$. In Decrypt query on $Hdr$, we require $Hdr \neq Hdr^*$. In both cases, $\mathcal{B}$ responds as described in Phase 1.

**Guess.** $\mathcal{A}$ returns the guessed $\beta'$ to $\mathcal{B}$. If $\beta = \beta'$, $\mathcal{B}$ decides $Z$ is equal to $e(g', g)^{\alpha^{M+1}}$. Else, $\mathcal{B}$ decides $Z$ is unequal to $e(g', g)^{\alpha^{M+1}}$.

Each key generation query requires O($M$) exponentiations and O($M$) multiplications in $\mathbb{G}$, and each decryption query requires O($M$) exponentiations, O($M$) multiplications, O(1) pairings in $\mathbb{G}$, so the time required by $\mathcal{B}$ is $t' \approx t$ + O($M(q_k+q_d)T_{exp}$) + O($M(q_k+q_d)T_{mul}$) + O($q_d T_{pair}$), where $T_{exp}$, $T_{pair}$ and $T_{mul}$ are the average time for exponentiation, pairing and

multiplication in $\mathbb{G}$. Only if the adversary can guess the right $s'$ or $ss'$ that will expose $K_\beta$, the adversary can return the right choice. If we use $P_{eq}$ to denote the probability of $Z = e(g', g)^{\alpha^{M+1}}$, $P_{neq}$ to denote the probability of $Z \neq e(g', g)^{\alpha^{M+1}}$, $\epsilon_1$ to denote the advantage of $\mathcal{A}$ in guessing the right choice in $P_{eq}$, and $\epsilon_2$ to denote the advantage of $\mathcal{A}$ in guessing the right choice in $P_{neq}$. We have $\epsilon = P_{eq} \cdot \epsilon_1 + P_{neq} \cdot \epsilon_2 = P_{eq} \cdot \epsilon_1 + P_{neq} \cdot (1/2 + 2/p)$, and $\epsilon' = P_{eq} \cdot (\epsilon_1 - 2/p) + P_{neq} \cdot 1/2$. Thus, we have $\epsilon' \approx \epsilon$.

Usually, we need additional proof to prove it negligible that the adversary is able to construct a valid ciphertext header for decryption oracle as $\langle j, \mathsf{com}^*, C_1, C_0, \mathsf{tag}^* \rangle$ satisfying $\mathsf{com}^* \neq \mathsf{com}$ or $\mathsf{tag}^* \neq \mathsf{tag}$. Since it is similar to the proof in [37], we omit it for brevity. □

**Theorem 2.** *At most t users (with a same value for a same tracing position) colluding in the proposed scheme, cannot recover the message encrypted for the other value of the same tracing position.*

**Proof:** (Sketch Proof.) The proof for Theorem 2 is quite similar to the proof for Theorem 1, except that the challenging ciphertext is computed only on the target codeword bit value for which the adversary is never issued any decryption key. The sketch phases are as follows.

**Setup.** This phase is the same as **Setup** in Theorem 1.

**Phase 1.** This phase is almost the same as **Phase 1** in Theorem 1 except that there should be at least one codeword bit value for which the adversary is never issued decryption keys. For instance, with respect to all queried $ID_u$, the challenger has never issued decryption keys for tracing positions $j$ and $i$ on the values $\bar{w}_j^{(u)} = 0$ and $\bar{w}_i^{(u)} = 1$.

**Challenge.** This phase is almost the same as **Challenge** in Theorem 1. The exceptions are that the tracing position is restricted to the positions where there is certain codeword bit value the adversary is never issued decryption keys for, and that the challenging ciphertext is computed only on the selected codeword bit value. For instance, the tracer can select $j$ as the tracing position and the ciphertext is computed only on $\bar{w}_j^{(u)} = 0$.

**Phase 2.** This phase is almost the same as **Phase 2** in Theorem 1 except that the adversary cannot be issued decryption key for codeword bit value

$\bar{w}_j^{(u)} = 0$ which is already used in the challenging ciphertext.

**Guess.** This phase is the same as **Guess** in Theorem 1.

As described above, if the adversary has any advantage in breaking Theorem 2, we has the same advantage to break the M-TBDHE assumption. The detailed proof is omitted for brevity. □

**Theorem 3.** *Our tracing and revoking scheme is t-collusion secure assuming the scheme is IND-ID-CCA2 secure, the employed collusion secure code is t-collusion secure and Theorem 2 holds.*

**Proof:** The $t$-collusion resistant game of our scheme is interacted between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$ as described in Section 2.

Suppose $(G, T)$ denotes a collusion secure code and $N$ is a positive integer and $\epsilon \in (0, 1)$. $\mathcal{B}$ runs $G(N, \epsilon)$ to obtain ($\Gamma$, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ where user with index $i$ will receive the codeword $\bar{w}^{(i)}$. $\mathcal{B}$ also generates an instance of our tracing and revoking scheme with public key PK. It sends $\Gamma$ and PK to $\mathcal{A}$. After that, $\mathcal{A}$ submits to $\mathcal{B}$ a set of identities $S^*$ with the subset of system indexes denoted as $C$ with $|C| \leq t$. $\mathcal{A}$ can query $\mathcal{B}$ for decryption keys of users in $C$. $\mathcal{B}$ generates decryption keys as described in the proposed scheme and forwards them to $\mathcal{A}$.

Now the challenger $\mathcal{B}$ can query $\mathcal{A}$ on decryptions. For each tracing position $i = 1, \ldots, L$, the challenger $\mathcal{B}$ queries $\mathcal{A}$ with message $m$ encrypted as described in algorithm **Trace** to a set $S$ of receivers. There are four cases for the tracing queries:

- Case 1: $\mathcal{A}$ does not hold any valid decryption key of users in $S$. The adversary will always output a random message other than $m$, as proved in Theorem 1. The probability that $\mathcal{A}$ responds with the right message is at most $1/|\mathcal{M}|$, where $|\mathcal{M}|$ is the number of messages in the message space. In this case, $\mathcal{B}$ changes to another set to continue. $\mathcal{B}$ will be deceived with probability at most $1/|\mathcal{M}|$;

- Case 2: $\mathcal{A}$ holds decryption keys for at least one user in $S$ (we use $S_A$ to denote these users whose decryption keys are held by $\mathcal{A}$, where $S_A = S \cap C$), and all codewords of users in $S_A$ contain "1" in current tracing position $i$. That is to say, all $\bar{w}^{(j)}(\forall j \in S_A)$ satisfy $\bar{w}_i^{(j)} = 1$. Thus, $\mathcal{A}$ will always output $m' = m$. The recovered bit $\bar{w}_i^*$ will always be 1.

Since all codewords of users in $S_A$ do not contain "0" in position $i$, the probability that $\mathcal{A}$ outputs $m' \neq m$ is less than $Adv_{FS}^{\mathcal{A}}$, the probability that the adversary breaks Theorem 2. $Adv_{FS}^{\mathcal{A}}$ is negligible as implied in Theorem 2;

- Case 3: $\mathcal{A}$ holds decryption keys for at least one user in $S$ (we denote the set of these users as $S_A$), and all codewords of users in $S_A$ contain "0" in current tracing position $i$. That is to say, all $\bar{w}^{(j)}(\forall j \in S_A)$ satisfy $\bar{w}_i^{(j)} = 0$. Thus, the adversary will always output $m' \neq m$. The recovered bit $\bar{w}_i^*$ will always be 0. Since all codewords in $S_A$ do not contain "1" in position $i$, the probability that the adversary outputs $m$ is less than $Adv_{FS}^{\mathcal{A}}$, the probability that the adversary breaks Theorem 2;

- Case 4: The adversary holds decryption keys for at least one user in $S$ (these users are denoted as $S_A$), and codewords of users in $S_A$ contain both "0" and "1" in current tracing position $i$. No matter what message the adversary outputs ($m$ or others), $\bar{w}_i^*$ must be in the feasible set of all codewords corresponding to $S_A$.

We use $W_{S_A}$ to denote the set of codewords corresponding to $S_A$. Therefore, the final recovered codeword $\bar{w}^* \in F(W_{S_A})$. From the assumption that collusion secure code ($G$, $T$) is $t$-collusion resistant, the probability that $T(\bar{w}^*$, TK) is empty or not a subset of $S_A$ is less than $\epsilon$. Thus, the probability that the adversary breaks the property of $t$-collusion resistance of our tracing and revoking scheme is less than $(1/|\mathcal{M}|)^L + 2L \cdot Adv_{FS}^{\mathcal{A}} + \epsilon$. □

As we can see, our scheme is fully collusion resistant when $t = N$.

**Theorem 4.** *The proposed scheme is public collaboration resistant.*

**Proof:** In our scheme, if some traitor leaks certain partial key $d^*$ in public, it can operate in two cases.

- Case 1: The traitor leaks $d_{ID,i}^*$ together with its identity $ID$, the key index $i$ of $d_{ID,i}^*$ in its set of decryption keys and the corresponding codeword bit value $\bar{w}_i^{(u)}$, so any one who picks up $d_{ID,i}^*$ can use it immediately. In this case, the traitor's identity $ID$ is exposed already.

- Case 2: The traitor leaks only the partial key $d^* = (K_{i,1}, K_{i,2}, T_{i,0}, \ldots, T_{i,M-1}, t_{i,ID})$ for some unknown $i$ and nothing else. In this case, the tracer can still learn the traitor's identity by comparing at most $2L + N$ rounds. For all $w_{1,1}$, $w_{1,0}$, $\ldots$, $w_{L,1}$, $w_{L,0}$, the tracer tests whether there is certain $w_{j,b}$ ($j \in \{1, \ldots, L\}$ and $b \in \{1, 0\}$) satisfying the following equation:

$$e(K_{i,1}, g) \quad = \quad e(w_{j,b}, g_1) \cdot e(g_1, g_2)^{t_{i,ID}} \cdot e(g_1, g_3)^{f(t_{i,ID})} \cdot e(z, K_{i,2}).$$

  When such a $j$ is found, for all $u = 1, \ldots, N$ the tracer tests whether there is a tuple of $(ID, u, \bar{w}^{(u)}, K_{ID,1}, \ldots, K_{ID,L})$ with a value $t_{j,ID}$ in $K_{ID,j}$ satisfying $t_{j,ID} = t_{i,ID}$. If such tuple $(ID, u, \bar{w}^{(u)}, K_{ID,1}, \ldots, K_{ID,L})$ exists, the tracer will accuse the user with identity $ID$ of releasing this partial key.

Thus, the partial decryption key in our scheme is self-enforced so that the scheme is resistant to public collaboration. □

## 4. Efficiency Analysis

We compare our scheme with several codes based schemes [2, 3, 5, 6, 7, 8, 9] in Table 1. In the comparison, all schemes are assumed to employ codewords of the same length $L$. In schemes that did not mention the employed public key encryption scheme (such as [5] and [6]), we assume that secure ElGamal encryption over a cyclic group of a large prime order $p$ is used. Table 1 shows that our scheme achieve constant ciphertext length as schemes in [5, 6, 9] do, and our scheme is almost as efficient as the scheme in [9].

## 5. Conclusion and Discussions

Inspired by the idea in [9], we describe a new construction of codes based tracing and revoking scheme. Our scheme is proved secure against chosen ciphertext attacks under adaptive-adversary model without random oracle while the scheme in [9] is only secure against chosen plaintext attacks under selective-adversary model with random oracle. Our scheme can also be extended to scheme adopting identifiable parent property (IPP) codes and scheme against imperfect pirate decoders as described in [9].

We notice that the scheme in [9] is based on $v$-Modified Bilinear Diffie-Hellman Assumption [10] that is not well studied, and our scheme is based on $q$-Truncated Bilinear Diffie-Hellman Exponent Assumption [13]. The above

Table 1: Comparison with Previous Codes Based Works

| | Public Key | Private Key | Ciphertext Length | Encryption Computation | Decryption Computation | Revocation | Pirate 2.0 Resistant |
|---|---|---|---|---|---|---|---|
| [2] | $2LG$ | $LZ_p$ | $2LG$ | $L$(2E+ 1M) | $L$(1E+ 2M) | No | No |
| [3] | 1G+ $(L+1)G_T$ | $LZ_p$+ $LG$ | 2G+ $LG_T$ | $LE_T$+ 2M | $L$(M+ 2P) | No | No |
| [5] | $2LG$ | $LZ_p$ | $1Z_p$+ 2G | (2E+ 1M) | (1E+ 2M) | No | No |
| [6] | $2LG$ | $LZ_p$ | $u(1Z_p+ 2G)$ | $u$(2E+ 1M) | $u$(1E+ 2M) | No | No |
| [7] | $(2L+1)$G | $(L+2)Z_p$ | $2Z_p+ (L+2)$G | (3E+1M) | (2E+2M) | No | No |
| [8] | $(L+2)G_1$ $+2G_2$ | $1G_1$ $+1G_2$ | $2LG_1$ $+2G_2+3Z_p$ | $2(L+1)E_1$ $+2M$ $+2E_2$ $+2E_T$ | $(L$-1)$E_1$ $+(L$-1)M $+2P+1M_T$ | No | Yes |
| [9] | $(N+1)$G $+1Z_p$ | $LG$ | $1Z_p$ $+4G$ $+2G_T$ | $2(\|S\|+1)$M $+4E+2E_T$ | $2(\|S\|$-1)E+$\|S\|$M $+2P+1M_T$ | Yes | Yes |
| Our Scheme | $(M+L+5)$G $+1Z_p$ | $L(M+2)$G $+LZ_p$ | $3Z_p$ $+4G$ $+6G_T$ | $2(\|S\|+2)$M + $2(\|S\|+2)$E $+2P+2E_T+(\|S\|+1)M_p$ | $\|S\|$E+$\|S\|$M $+2P+2E_T$ $+1M_T+\|S\|M_p$ | Yes | Yes |

$L$: the length of codeword; $|S|$: the number of users in the set $S$ of receivers;
$N$: the total number of users in the system;
$u$: the number of codeword positions used in encryption [6];
G: element in $\mathbb{G}$; $G_1$: element in $\mathbb{G}_1$; $G_2$: element in $\mathbb{G}_2$; $G_T$: element in $\mathbb{G}_T$;
$Z_p$: element in $\mathbb{Z}_p$; P: pairing in $\mathbb{G} \times \mathbb{G}$ or $\mathbb{G}_1 \times \mathbb{G}_2$;
E: exponentiation in $\mathbb{G}$; $E_1$: exponentiation in $\mathbb{G}_1$; $E_2$: exponentiation in $\mathbb{G}_2$;
M: multiplication (or division) in $\mathbb{G}$; $M_1$: multiplication (or division) in $\mathbb{G}_1$;
$M_p$:multiplication in $\mathbb{Z}_p$; $M_T$: multiplication (or division) in $\mathbb{G}_T$.

two assumptions are both $q$-based, so future works may focus on constructing codes based tracing and revoking schemes based on static complexity assumptions.

## References


[1] B. Chor, A. Fiat, M. Naor, Tracing traitors, in: Y. Desmedt (Ed.), CRYPTO, Vol. 839 of Lecture Notes in Computer Science, Springer, 1994, pp. 257–270.

[2] A. Kiayias, M. Yung, Traitor tracing with constant transmission rate, in: L. R. Knudsen (Ed.), EUROCRYPT, Vol. 2332 of Lecture Notes in Computer Science, Springer, 2002, pp. 450–465.

[3] H. Chabanne, D. H. Phan, D. Pointcheval, Public traceability in traitor tracing schemes, in: R. Cramer (Ed.), EUROCRYPT, Vol. 3494 of Lecture Notes in Computer Science, Springer, 2005, pp. 542–558.

[4] M. Abdalla, A. W. Dent, J. Malone-Lee, G. Neven, D. H. Phan, N. P. Smart, Identity-based traitor tracing, in: T. Okamoto, X. Wang (Eds.), Public Key Cryptography, Vol. 4450 of Lecture Notes in Computer Science, Springer, 2007, pp. 361–376.

[5] D. Boneh, M. Naor, Traitor tracing with constant size ciphertext, in: P. Ning, P. F. Syverson, S. Jha (Eds.), ACM Conference on Computer and Communications Security, ACM, 2008, pp. 501–510.

[6] O. Billet, D. H. Phan, Efficient traitor tracing from collusion secure codes, in: R. Safavi-Naini (Ed.), ICITS, Vol. 5155 of Lecture Notes in Computer Science, Springer, 2008, pp. 171–182.

[7] Y.-R. Chen, W.-G. Tzeng, A public-key traitor tracing scheme with an optimal transmission rate, in: S. Qing, C. J. Mitchell, G. Wang (Eds.), ICICS, Vol. 5927 of Lecture Notes in Computer Science, Springer, 2009, pp. 121–134.

[8] X. Zhao, F. Zhang, Traitor tracing against public collaboration, in: F. Bao, J. Weng (Eds.), ISPEC, Vol. 6672 of Lecture Notes in Computer Science, Springer, 2011, pp. 302–316.

[9] X. Zhao, H. Li, Codes based tracing and revoking scheme with constant ciphertext, in: T. Takagi, G. Wang, Z. Qin, S. Jiang, Y. Yu (Eds.), ProvSec, Vol. 7496 of Lecture Notes in Computer Science, Springer, 2012, pp. 318–335.

[10] J. H. Park, H. J. Kim, M. Sung, D. H. Lee, Public key broadcast encryption schemes with shorter transmissions, Broadcasting, IEEE Transactions on 54 (3) (2008) 401–411. doi:10.1109/TBC.2008.919940.

[11] O. Billet, D. H. Phan, Traitors collaborating in public: Pirates 2.0, in: A. Joux (Ed.), EUROCRYPT, Vol. 5479 of Lecture Notes in Computer Science, Springer, 2009, pp. 189–205.

[12] D. Boneh, M. Hamburg, Generalized identity based and broadcast encryption schemes, in: J. Pieprzyk (Ed.), ASIACRYPT, Vol. 5350 of Lecture Notes in Computer Science, Springer, 2008, pp. 455–470.

[13] C. Gentry, Practical identity-based encryption without random oracles, in: S. Vaudenay (Ed.), EUROCRYPT, Vol. 4004 of Lecture Notes in Computer Science, Springer, 2006, pp. 445–464.

[14] D. R. Stinson, R. Wei, Combinatorial properties and constructions of traceability schemes and frameproof codes, SIAM J. Discrete Math. 11 (1) (1998) 41–53.

[15] A. Fiat, T. Tassa, Dynamic traitor training, in: M. J. Wiener (Ed.), CRYPTO, Vol. 1666 of Lecture Notes in Computer Science, Springer, 1999, pp. 354–371.

[16] D. Naor, M. Naor, J. Lotspiech, Revocation and tracing schemes for stateless receivers, in: J. Kilian (Ed.), CRYPTO, Vol. 2139 of Lecture Notes in Computer Science, Springer, 2001, pp. 41–62.

[17] K. Kurosawa, Y. Desmedt, Optimum traitor tracing and asymmetric schemes, in: K. Nyberg (Ed.), EUROCRYPT, Vol. 1403 of Lecture Notes in Computer Science, Springer, 1998, pp. 145–157.

[18] D. Boneh, M. K. Franklin, An efficient public key traitor tracing scheme, in: M. J. Wiener (Ed.), CRYPTO, Vol. 1666 of Lecture Notes in Computer Science, Springer, 1999, pp. 338–353.

[19] M. Naor, B. Pinkas, Efficient trace and revoke schemes, in: Y. Frankel (Ed.), Financial Cryptography, Vol. 1962 of Lecture Notes in Computer Science, Springer, 2000, pp. 1–20.

[20] S. Mitsunari, R. Sakai, M. Kasahara, A new traitor tracing, IEICE transactions on fundamentals of electronics, communications and computer sciences E85-A (2) (2002) 481–484.

[21] Y. Dodis, N. Fazio, Public key trace and revoke scheme secure against adaptive chosen ciphertext attack, in: Y. Desmedt (Ed.), Public Key Cryptography, Vol. 2567 of Lecture Notes in Computer Science, Springer, 2003, pp. 100–115.

[22] D. Boneh, A. Sahai, B. Waters, Fully collusion resistant traitor tracing with short ciphertexts and private keys, in: S. Vaudenay (Ed.), EUROCRYPT, Vol. 4004 of Lecture Notes in Computer Science, Springer, 2006, pp. 573–592.

[23] D. Boneh, B. Waters, A fully collusion resistant broadcast, trace, and revoke system, in: A. Juels, R. N. Wright, S. D. C. di Vimercati (Eds.), ACM Conference on Computer and Communications Security, ACM, 2006, pp. 211–220.

[24] S. Garg, A. Kumarasubramanian, A. Sahai, B. Waters, Building efficient fully collusion-resilient traitor tracing and revocation schemes, in: E. Al-Shaer, A. D. Keromytis, V. Shmatikov (Eds.), ACM Conference on Computer and Communications Security, ACM, 2010, pp. 121–130.

[25] J. H. Park, D. H. Lee, Fully collusion-resistant traitor tracing scheme with shorter ciphertexts, Des. Codes Cryptography 60 (3) (2011) 255–276.

[26] J. H. Park, H. S. Rhee, D. H. Lee, Fully collusion-resistant trace-and-revoke scheme in prime-order groups, Journal of Communications and Networks 13 (5) (2011) 428–441.

[27] H. D. L. Hollmann, J. H. van Lint, J.-P. M. G. Linnartz, L. M. G. M. Tolhuizen, On codes with the identifiable parent property, J. Comb. Theory, Ser. A 82 (2) (1998) 121–133.

[28] A. Kiayias, S. Pehlivanoglu, Pirate evolution: How to make the most of your traitor keys, in: A. Menezes (Ed.), CRYPTO, Vol. 4622 of Lecture Notes in Computer Science, Springer, 2007, pp. 448–465.

[29] H. Jin, J. B. Lotspiech, Defending against the pirate evolution attack, in: F. Bao, H. Li, G. Wang (Eds.), ISPEC, Vol. 5451 of Lecture Notes in Computer Science, Springer, 2009, pp. 147–158.

[30] D. H. Phan, V. C. Trinh, Identity-based trace and revoke schemes, in: X. Boyen, X. Chen (Eds.), ProvSec, Vol. 6980 of Lecture Notes in Computer Science, Springer, 2011, pp. 204–221.

[31] M. Abdalla, D. Catalano, A. W. Dent, J. Malone-Lee, G. Neven, N. P. Smart, Identity-based encryption gone wild, in: M. Bugliesi, B. Preneel, V. Sassone, I. Wegener (Eds.), ICALP (2), Vol. 4052 of Lecture Notes in Computer Science, Springer, 2006, pp. 300–311.

[32] P. D'Arco, A. L. P. del Pozo, Fighting pirates 2.0, in: J. Lopez, G. Tsudik (Eds.), ACNS, Vol. 6715 of Lecture Notes in Computer Science, 2011, pp. 359–376.

[33] D. Boneh, M. Franklin, Identity-based encryption from the weil pairing, in: CRYPTO 2001, Vol. 2139 of Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 213–229.

[34] D. Boneh, X. Boyen, E.-J. Goh, Hierarchical identity based encryption with constant size ciphertext, in: R. Cramer (Ed.), EUROCRYPT, Vol. 3494 of Lecture Notes in Computer Science, Springer, 2005, pp. 440–456.

[35] N. Attrapadung, B. Libert, E. de Panafieu, Expressive key-policy attribute-based encryption with constant-size ciphertexts, in: D. Catalano, N. Fazio, R. Gennaro, A. Nicolosi (Eds.), Public Key Cryptography, Vol. 6571 of Lecture Notes in Computer Science, Springer, 2011, pp. 90–108.

[36] R. Canetti, S. Halevi, J. Katz, Chosen-ciphertext security from identity-based encryption, in: C. Cachin, J. Camenisch (Eds.), EUROCRYPT, Vol. 3027 of Lecture Notes in Computer Science, Springer, 2004, pp. 207–222.

[37] D. Boneh, J. Katz, Improved efficiency for cca-secure cryptosystems built using identity-based encryption, in: A. Menezes (Ed.), CT-RSA, Vol. 3376 of Lecture Notes in Computer Science, Springer, 2005, pp. 87–103.