Traitor Tracing against Public Collaboration*

Xingwen Zhao, Fangguo Zhang

School of Information Science and Technology, Sun Yat-sen University
Guangzhou 510275, P.R.China
Guangdong Key Laboratory of Information Security Technology
Guangzhou 510275, P.R.China
sevenzhao@hotmail.com, isszhfg@mail.sysu.edu.cn

Abstract. Broadcast encryption provides a convenient method to distribute digital content to subscribers over an insecure broadcast channel. Traitor tracing is needed because some users may give out their decryption keys to construct pirate decoders. There are many traitor tracing schemes based on collusion secure codes and identifiable parent property codes. However, these schemes are subject to public collaboration of traitors, which is presented by Billet and Phan in EUROCRYPT 2009 as an attack against code-based traitor tracing schemes. In this paper, we describe a generic collusion secure codes based scheme secure against such collaboration. Our scheme is motivated by the idea of identity-based encryption with wildcards (WIBE). We regard the collusion secure codeword for each user as his/her identity, and issue private key accordingly. When in broadcasting, we use a special pattern of WIBE, namely all bit positions in the codewords of intended receivers are set as wildcards. When in tracing, we use another special pattern of WIBE, namely all positions are set as wildcards except the tracing position. By using WIBE, each user is issued one decryption key which should be used as a whole and any incomplete part of the key is useless, while in previous codes based schemes each user holds a number of keys that can be used separately for different bit positions in the codeword. Thus our scheme is resistant to public collaboration, since if the decryption key is disclosed as a whole, it will immediately lead to the accusation of the very traitor. Our idea fits well for code based traitor tracing schemes, no matter collusion secure codes or identifiable parent property codes. We also provide an instance based on Boneh-Boyen-Goh WIBE scheme, achieving constant private key storage cost for each user. Our scheme presents an answer to the problem left open by Billet and Phan.

Key words: Broadcast encryption, traitor tracing, public collaboration.

1 Introduction

Broadcast encryption provides a convenient method to distribute digital content to subscribers over an insecure broadcast channel so that only the qualified users

* This work is supported by the National Natural Science Foundation of China (No. 60773202, 61070168).

can recover the data. Broadcast encryption is quite useful and enjoys many applications including pay-TV systems, distribution of copyrighted materials such as DVD.

Because some users (called traitors) may give out their decryption keys to construct pirate decoders, and some users (also called traitors) may directly spread the decrypted contents over the Internet (known as anonymous attack [15] or pirate rebroadcast [17]), the ability of traitor tracing is needed for broadcast encryption system. Traitor tracing scheme is used to discourage legitimate subscribers from giving away their secret keys and decrypted contents. Therefore, there are two kinds of traitor tracing schemes, i.e. schemes against pirate decoders and schemes against pirate rebroadcast. In this paper, we focus on traitor tracing against pirate decoders. The first traitor tracing scheme against pirate decoders was presented by Chor, Fiat and Naor in [12]. Since then, many works have been presented. Here, we discuss some of them in details.

1.1 Related Works on Traitor Tracing against Pirate Decoders

Since the introduction of traitor tracing by Chor, Fiat and Naor in [12], many traitor tracing schemes against pirate decoders were proposed and they can be roughly classified into three categories.

The first category is called combinatorial, as in [12, 26, 13, 22]. These schemes carefully choose some subsets of keys to be put in each decryption box. By analyzing the keys used in a pirate decoder, it is possible to trace one of the traitors. Another category is called algebraic, as in [19, 6, 23, 21, 8, 9, 14, 24]. These schemes use algebraic method to assign private keys to users and the broadcasting can be done in public since public-key techniques are used. Collusion secure codes based schemes can be regarded as the third category, which combines ideas from the two previous classes. For instance, [18, 10, 7, 3, 11] belong to this category. These schemes assign keys to each user according to each bit of his/her codeword. By analyzing the keys used in each bit positions, the tracer can recover the codeword embedded in the decoder and trace back to at least one of the traitors.

Some schemes [10, 9, 14, 24] allow public traceability, which means the tracing can be performed by anyone and is not limited to the tracing authority.

When traitors are found, it is desirable to make them useless. However, not all traitor tracing schemes support revocation. Many schemes merely consider the tracing of traitors, and they do not consider the revocation of traitors. Some schemes [23, 22, 9, 14] combine the tracing and revoking abilities to make the schemes more practical.

Some works [16, 4] focus on attacks against traitor tracing schemes. Kiayias and Pehlivanoglu [16] presented pirate evolution attack against schemes based on subset-cover revocation framework [22]. In such attack, a traitor holding a number of keys can produces a number of generations of pirate decoders (called pirate evolution) so that the system has to disable them generation by generation costly. Billet and Phan [4] presented new attack named “Pirates 2.0” mainly against schemes based on traceability codes (collusion secure codes and

identifiable parent property codes) and schemes based on subset-cover revocation framework. The attack shows that users can release certain part of their private keys in a public way, so that pirate decoders can be built from the public information. Each traitor remains anonymous because a large number of users contain the same keys as those released in public.

1.2 Our Contributions

In this paper, we describe a generic collusion secure codes based scheme secure against public collaboration of traitors. Our scheme is motivated by the idea of identity-based encryption with wildcards (WIBE). We take the tracing code for each user as his/her identity, and issue private key accordingly. When in broadcasting, we use a special pattern of WIBE, namely all bit positions in the codewords of intended receivers are set as wildcards. When in tracing, we use another special pattern of WIBE, namely all positions are set as wildcards except the tracing position. By using WIBE, the decryption key for each user should be used as a whole and any incomplete part of the key is useless. Thus, our scheme is resistant to public collaboration, since any release of decryption key as a whole will immediately leads to accusation of the very traitor. Our generic scheme can be altered to adopt identifiable parent property codes with a few adjustments. We also present an instance based on Boneh-Boyen-Goh WIBE scheme, in which storage cost for each user is constant.

1.3 Organization

The remainder of this paper is organized as follows. Brief descriptions of our building tools, namely collusion secure codes and identity-based encryption with wildcards, are given in Section 2. In Section 3 protocol model and security requirements for our traitor tracing scheme are defined. In Section 4, we describe our idea and give out the generic tracing scheme against public collaboration. Security analysis on our scheme and some extensions of our scheme are also given in Section 4. An instance based on Boneh-Boyen-Goh WIBE scheme and its performance comparison with previous works is given in Section 5. Section 6 concludes the paper.

2 Building Tools

2.1 Collusion Secure Codes

We first review the definition of collusion secure codes required for constructing our traitor tracing scheme. The definition is similar to that in [7].

- For a word $\bar{w} \in \{0,1\}^L$ we write $\bar{w} = w_1 \dots w_L$, where $w_i \in \{0,1\}$ is the ith bit of $\bar{w}$ for $i = 1, \dots, L$.

- Let $W = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(t)}\}$ be a set of words in $\{0,1\}^L$. We say that a word $\bar{w} \in \{0,1\}^L$ is feasible for W if for all $i = 1, \ldots, L$ there is a $j \in \{1, \ldots, t\}$ such that $\bar{w}_i = \bar{w}_i^{(j)}$. For example, if W consists of the two words $\left(\begin{smallmatrix}0&1&1&0&0\\0&0&1&1&0\end{smallmatrix}\right)$, then all words of the form $[0\ \binom{1}{0}\ 1\ \binom{0}{1}\ 0]$ are feasible for W.
- For a set of words $W \subseteq \{0,1\}^L$ we say that the feasible set of W, denoted $F(W)$, is the set of all words that are feasible for W.

The collusion secure code can be denoted with a pair of polynomial time algorithms (G, T) defined as follows:

- Algorithm G, called a code generator is a probabilistic algorithm that takes a pair (N, ϵ) as input, where N is the number of words to output and $\epsilon \in (0,1)$ is a security parameter. The algorithm outputs a pair (Γ, TK). Here Γ (called a code) contains N words in $\{0,1\}^L$ for some $L > 0$ (called the code length). TK is called the tracing key.
- Algorithm T, called a tracing algorithm, is a deterministic algorithm that takes as input a pair ($\bar{w}^*$, TK) where $\bar{w}^* \in \{0,1\}^L$. The algorithm outputs a subset S of $\{1, \ldots, N\}$. Informally, elements in S are accused of creating the word $\bar{w}^*$.

The collusion resistant property of collusion secure code (G, T) is defined 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, ϵ, C) as input. Then the game proceeds as follows:

1. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$. It sends the set $W := \{\bar{w}^{(N)}\}_{i \in C}$ 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 $T(\bar{w}^*, \text{TK})$ 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.

Definition 1. *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 negligible (less than ϵ).*
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 negligible (less than ϵ).

Our readers can refer to [7] for known results on collusion secure codes. Additionally, Boneh and Naor [7] also constructed δ-robust Boneh-Shaw codes in order to trace high error-rate pirate decoders.

2.2 Identity-based Encryption with Wildcards

Identity-based encryption with wildcards (WIBE) [2] schemes, are a generalization of hierarchical identity-based encryption (HIBE) [5] schemes. The sender in

WIBE can encrypt the messages to a range of users whose identities match a certain pattern. Such a pattern is described by a vector $P = (P_1 \ldots, P_l) \in (\{0,1\}^* \cup \{*\})^l$, where $*$ is the wildcard symbol. If a user's identity $ID = (ID_1, \ldots, ID_{l'})$ satisfies that $l' \leq l$ and for all $i = 1, \ldots, l'$ we have $ID_i = P_i$ or $P_i = *$, we say that identity ID matches the pattern P (denoted as $ID \in_* P$). In other words, in an identity-based encryption with wildcards, each user's identity is arranged in hierarchical structure and encryption can be made to a pattern of identity with wildcards. For instance, an organization Org1 contains two offices Off1 and Off2, and Alice and Bob belong to Off1. Then Alice obtains her identity as Alice.Off1.Org1, and Bob's identity is Bob.Off1.Org1. We can also assign other identities, such as Manager.Off1.Org1 and Manager.Off2.Org1 to the managers in both offices. If someone wants to send a message to Alice only, he/she can encrypt the message on the pattern Alice.Off1.Org1. If the message is for all persons in Off1, then the pattern will be $*$.Off1.Org1. If the message is for all persons in Org1, then the pattern will be $*.*$.Org1. If the message is for the managers in both offices of Org1, then the pattern will be Manager.$*$.Org1.

Formally, a WIBE scheme is a tuple of algorithms (WIBE.Setup, WIBE.KeyGen, WIBE.Encrypt, WIBE.Decrypt) described as follows.

- **WIBE.Setup**(1^λ). It is a probabilistic algorithm that given 1^λ, generates a master key pair (mpk, msk). It publishes the master public key mpk and the trusted authority keeps the master secret key msk private.
- **WIBE.KeyGen**(msk, ID). It is a probabilistic algorithm (run by the trusted authority) that given master secret key msk and user's identity ID, generates a decryption key d_{ID} and sends this key over a secure and authenticated channel to the user.
- **WIBE.Encrypt**(mpk, P, m). It is a probabilistic algorithm that given master public key mpk, an identity patter P and the message m, generates a ciphertext C.
- **WIBE.Decrypt**(mpk, P, C, d_{ID}). It is a deterministic algorithm that given master public key mpk, an identity patter P, the ciphertext C and a user's decryption key d_{ID}, recovers the message m if $ID \in_* P$.

The security of WIBE scheme can be defined using the following IND-WID-CPA game, which is played between a challenger and an adversary $\mathcal{A}$:

1. The challenger generates a master key pair $(mpk, msk) \leftarrow$ **WIBE.Setup**(1^λ).
2. The adversary $\mathcal{A}$ is given access to a key generation oracle that, on input of an identity $ID = (ID_1, \ldots, ID_l)$, returns the secret key $d_{ID} \leftarrow$ **WIBE.KeyGen**(msk, ID) corresponding to that identity. The adversary outputs two equal-length messages (m_0, m_1) and a challenge pattern P^*.
3. The challenger chooses a random bit $b \in \{0,1\}$, and computes the ciphertext $C^* \leftarrow$ **WIBE.Encrypt**(mpk, P^*, m_b).
4. The adversary $\mathcal{A}$ who is given access to a key generation oracle as before, on the input C^*, outputs a bit $b' \in \{0,1\}$.

The adversary wins the IND-WID-CPA game if $b' = b$ and it never queries the key generation oracle on any identity ID which matches the pattern P^*, i.e. any identity $ID \in_* P^*$. The adversary's advantage is defined as $Adv^{\mathcal{A}}_{CPA}=|Pr[b' = b] - 1/2|$.

Definition 2. *A (t, q_K, ϵ)-adversary against the IND-WID-CPA security of the WIBE scheme is an algorithm that runs in time at most t, makes at most q_K queries on key generation oracle, and has advantage at least ϵ in the IND-WID-CPA game described above.*

3 Protocol Model and Security Requirements

3.1 Protocol Model

The protocol model for our scheme consists of four algorithms (Setup, Encrypt, Decrypt, Trace) described as follows.

- **Setup**(1^λ, N). It is a probabilistic algorithm that given 1^λ and the number of users in the system N, outputs a public broadcast-key BK, a secret trace-key TK, and the private user-key SK_u for each user $u \in \{1, \ldots, N\}$.
- **Encrypt**(BK, M). It is a probabilistic algorithm that given a broadcast-key BK and a message M, a broadcast ciphertext C is generated.
- **Decrypt**(SK_u, C). It is an algorithm that given a broadcast ciphertext C and the private user-key SK_u of user u, returns the recovered messages M or $\perp$.
- **Trace**$^{\mathcal{D}}$(TK). It is an algorithm that given a pirate decoder $\mathcal{D}$ and private trace-key TK, it queries decoder $\mathcal{D}$ as a black-box oracle and then outputs a traitor set $T \subseteq \{1, \ldots, N\}$.

3.2 Security Requirements

- **Correctness**. Each honest user is able to recover the messages in normal broadcasting.
- **Semantic Security**. The users cannot obtain any information of messages encrypted in the broadcast ciphertext, if their identities do not matches the encryption pattern. Semantic security is defined in a game similar to IND-WID-CPA game for WIBE in Section 2.
- **Collusion Resistant**. Collusion of users cannot produce a decoder that cannot be traced to any of these users.

  The collusion resistant property of proposed traitor tracing scheme is defined using the following game between a challenger and an adversary. 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. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword (and identity) for user i. The challenger also selects a WIBE scheme with public parameters mpk. It sends Γ and mpk to the adversary.

2. The adversary selects a subset of Γ, denoted as C. The adversary can query the challenger for decryption keys of the codewords in C. The challenger generates the keys as in WIBE and gives them to the adversary.
3. The challenger asks the adversary to decrypt ciphertexts a number of times and recovers a codeword $\bar{w}^*$.

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

– **Resistant against Public Collaboration**. Incomplete public collaboration of traitors cannot generate useful keys for constructing pirate decoders. Public collaboration of complete key from a traitor will immediately lead to the accusation of the very traitor.

4 Traitor Tracing Scheme against Public Collaboration

In this section, we describe our generic code-based traitor tracing scheme secure against public collaboration. We describe our idea firstly, and then present the scheme. For brevity, we only describe a generic scheme based on collusion secure code and scheme based on identifiable parent property code can be obtained by simple adjustments.

4.1 Construction Idea

Our idea is motivated by the all-or-nothing transform method [25] and the idea of identity-based encryption with wildcards.

In collusion secure code based traitor tracing schemes where all keys of each user are used in decryption, such as [18, 10], only one bit (as tracing position) of the codewords is checked in each round of tracing procedure, and the other bits (as normal positions) are all enabled. That is to say, the tracer only requires a valid key for each normal position and does not care about whether the key corresponds to an "1" or a "0". Therefore, we can regard the broadcasting or tracing pattern as $P = (P_1, \ldots, P_i, \ldots, P_L)$, where i is the tracing position, L is the length of codewords and P_j is identity pattern for position j, j=1,..., L. When in broadcasting, the pattern $P = (*, \ldots, *, \ldots, *)$, with tracing position P_i and other normal positions are all wildcards. When in tracing, the pattern $P = (*, \ldots, (1/0), \ldots, *)$, with tracing position $P_i = 1$ or 0 and other position $P_j = *$, for each $1 \leq j \leq L$, $j \neq i$. In other words, the ciphertext for $P_i = 1$ is different from the ciphertext for $P_i = 0$, so as to identify whether the ith position of codeword embedded in the pirate decoder is an "1" or a "0".

Therefore, when we assign a distinct collusion secure code to each user, we can treat such code as each user's identity. If the length of collusion secure code is L, then we use L-level WIBE to assign private key to each user. For instance, if a codeword $\bar{w} \in \{0,1\}^L$ is assigned to a user, the trusted party in L-level WIBE generates a decryption key associating with identity $\bar{w}$ to this user. And the user is allowed to decrypt messages designated for pattern P where $\bar{w} \in_* P$. When in broadcasting, we use the pattern

$$P = (*, \ldots, \tbinom{1}{0}, \ldots, *)$$

where tracing position $P_i = \binom{1}{0}$ and all other positions $P_j = *$ to encrypt the message to all users. $P_i = \binom{1}{0}$ means that both "1" and "0" are enabled in tracing position i. $(*, \ldots, \binom{1}{0}, \ldots, *)$ is equal to $(*, \ldots, *)$ in WIBE scheme with identity in each level satisfying $ID_i \in \{0, 1\}$ for $i = 1, \ldots, L$. We use the pattern of this form $(*, \ldots, \binom{1}{0}, \ldots, *)$ so as to be consistent with the tracing pattern. When in tracing, we use the pattern

$$P = (*, \ldots, 1, \ldots, *)$$

with $P_i = 1$ to encrypt the message m to all users with 1 in ith position. $P_i = 1$ means only "1" is enabled in tracing position i. All users with 0 in ith position will output a random message. We require that the ciphertext has the same form no matter $P_i = \binom{1}{0}$ or $P_i = 1$, which is easy to fulfill as follows: when $P_i = \binom{1}{0}$ there are two normal elements, one is for "1" and the other is for "0"; when $P_i = 1$ there are two elements, one is normal element for "1" and the other is a random element for "0". If the pirate decoder outputs m, we decide that the decoder contains a codeword $\bar{w}^*$ with $\bar{w}_i^* = 1$. Otherwise, we decide that $\bar{w}_i^* = 0$ (for the perfect decoders. We will discuss imperfect decoders later). We repeat the tracing algorithm a number of times (decided later) from bit 1 to bit L, and recover the collusion secure codeword $\bar{w}^*$ from the captured pirate decoder $\mathcal{D}$. By using the identifying algorithm T of collusion secure code, we eventually obtain a set of traitors from the codeword $\bar{w}^*$. Because of the provable security of WIBE, each traitor should release his private key as a whole so as to be useful for constructing pirate decoders. Thus our scheme is resistant to public collaboration of traitors, since any release of the whole private key in public will directly lead to the accusation of the very traitor.

4.2 Proposed Generic Scheme

Based on the building tools, namely collusion secure codes and identity-based encryption with wildcards described in Section 2, we obtain a generic traitor tracing scheme against public collaboration. The scheme is denoted by a tuple of four algorithms (Setup, Encrypt, Decrypt, Trace) as follows.

- **Setup**$(1^\lambda, N)$. It is a probabilistic algorithm, in which a trusted party, given 1^λ and the number of users in the system N, selects $\epsilon \in (0, 1)$ and runs collusion secure code generation algorithm $\mathbf{G}(N, \epsilon)$ to generate a pair (Γ, TK). The set $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ contains N codewords in $\{0, 1\}^L$, where L is the codeword length which is decided by total number of users N, collusion threshold t and ϵ (shown in Section 2). TK is the tracing key for Γ. $\bar{w}^{(u)}$ is assigned to user u as the identity, with $1 \leq u \leq N$. The trusted party selects a WIBE scheme and runs its **WIBE.Setup**(1^λ) to generate a master key pair (mpk, msk). The trusted party runs **WIBE.KeyGen**(msk, $\bar{w}^{(u)}$) to generate decryption key SK_u for each user u. Each decryption key is transferred to user over a secure and authenticated channel which is not considered in this paper. mpk is the master public key and also the broadcast-key BK of the tracing scheme.

- **Encrypt**(BK, m). Anyone who wants to encrypt a message m to all users, given the broadcast-key BK, selects random position $i \in \{1, \ldots, L\}$ to generate a pattern $P = (P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, \binom{1}{0}, \ldots, *)$ and runs **WIBE.Encrypt**(BK, P, m) to obtain a ciphertext C. (i, C) is broadcast to all users.
- **Decrypt**(SK_u, C). Given a broadcast ciphertext (i, C), user u constructs pattern P^* as $(P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, \binom{1}{0}, \ldots, *)$ and uses the private user-key SK_u to run **WIBE.Decrypt**(mpk, P^*, C, $d_{\bar{w}^{(u)}}$). User u returns the recovered messages m or $\perp$.
- **Trace**$^{\mathcal{D}}$(TK). Given a perfect pirate decoder $\mathcal{D}$, the trusted party queries decoder $\mathcal{D}$ as a black-box oracle. For $i = 1, \ldots, L$, the trusted party acts as follows:
  1. generates pattern P as $(P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, 1, \ldots, *)$, where $P_i = 1$;
  2. runs **WIBE.Encrypt**(BK, P, m) to obtain a ciphertext C. (i, C) is fed to the decoder;
  3. if the pirate decoder outputs m, the trusted party decides that the decoder contains a codeword $\bar{w}^*$ with $\bar{w}_i^* = 1$. Otherwise, $\bar{w}_i^* = 0$.

  As we notice that, the decoder will construct the pattern P^* as $(*, \ldots, \binom{1}{0}, \ldots, *)$ and decrypt as normal. However, our encryption pattern is $P = (*, \ldots, 1, \ldots, *)$, so that only decoders with $\bar{w}^* \in_* P$ can recover the message m. Decoders with $\bar{w}^* \in_* (*, \ldots, 0, \ldots, *)$ will return a random message other than m. After the trusted party obtains the recovered codeword $\bar{w}^* = \bar{w}_1^* \ldots \bar{w}_L^*$, it runs the tracing algorithm of collusion secure code as $\mathbf{T}(\bar{w}^*$, TK) and outputs a set of traitors $T \subseteq \{1, \ldots, N\}$.

We require that collusion secure code (G, T) is fully collusion resistant (resp. t-collusion resistant) in order for our generic scheme to be fully collusion resistant (resp. t-collusion resistant). We also require that WIBE scheme is correct and secure against IND-WID-CPA game.

4.3 Security Analysis

Correctness. The correctness is straightforward due to the correctness of WIBE scheme.

Theorem 1. *The generic traitor tracing scheme is semantically secure assuming the WIBE scheme is semantically secure.*

Proof: We suppose the semantic security game of our traitor tracing scheme is played between a challenger CH_{TT} and an adversary ADV_{TT}. They interact as follows:

1. When ADV_{TT} initials the game, CH_{TT} initials IND-WID-CPA game of WIBE and acts as the adversary ADV_{WIBE} in IND-WID-CPA game. When the challenger CH_{WIBE} in IND-WID-CPA game returns mpk, CH_{TT} forwards it to ADV_{TT}.

2. When ADV_{TT} queries decryption key for an identity $ID \in \{0,1\}^L$, CH_{TT} forwards the query to CH_{WIBE}, and then forwards to ADV_{TT} the decryption key returned by CH_{WIBE}. When ADV_{TT} outputs two equal-length messages (m_0, m_1) and a tracing pattern P_i with $i \in \{1, \ldots, L\}$, CH_{TT} constructs a pattern $P^* = (P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, P_i, \ldots, *)$ and forwards (m_0, m_1, P^*) to CH_{WIBE}. In this step, we require that ADV_{TT} has not queried key generation oracle on any identity $ID \in_* P^*$. For instance, if $P_i = \binom{1}{0}$, we require ADV_{TT} has not queried key generation oracle on any codeword; if $P_i = 1$, we require ADV_{TT} has not queried key generation oracle on any codewords with $\bar{w}_i^* = 1$.
3. When CH_{WIBE} returns the challenging ciphertext C^*, CH_{TT} forwards it to ADV_{TT} with the form (i, C^*).
4. ADV_{TT} is allowed to access the key generation oracle as Step 2, with the requirement that ADV_{TT} does not query key generation oracle on any identity $ID \in_* P^*$. Finally, ADV_{TT} outputs a bit $b' \in \{0,1\}$ and CH_{TT} forwards b' to CH_{WIBE} as the answer for the challenge of IND-WID-CPA game.

Our scheme uses the special pattern of WIBE to encrypt messages. If the adversary has advantage to break the semantic security of our traitor tracing scheme, then it can be used to break the semantic security of WIBE with the same advantage. □

Theorem 2. *The generic traitor tracing scheme is t-collusion resistant assuming WIBE is secure against IND-WID-CPA game and collusion secure code (G, T) is t-collusion resistant.*

Proof: The t-collusion resistant game of our proposed scheme is played between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$ as described in Section 3.

Let (G, T) be a collusion secure code. Let N be an integer and $\epsilon \in (0, 1)$. The challenger runs $G(N, \epsilon)$ to obtain (Γ, TK) where $\Gamma = \{\bar{w}^{(1)}, \ldots, \bar{w}^{(N)}\}$ and $\bar{w}^{(i)}$ is the codeword (and identity) for user i. The challenger also selects a WIBE scheme with public parameters mpk. It sends Γ and mpk to the adversary. Then, the adversary selects a subset of Γ, denoted as C. The adversary can query the challenger for decryption keys of the codewords in C. The challenger generates the keys as in WIBE and gives them to the adversary.

When it is time for the challenger to query the adversary on decryptions, for the tracing position $i = 1, \ldots, L$, the challenger queries the adversary with message m encrypted in the pattern $P = (P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, 1, \ldots, *)$, where $P_i = 1$. There are three cases for the decoder:

– Case 1: All codewords held by the adversary contain "1" in tracing position i. Since all codewords match the encryption pattern P, the adversary will always output $m' = m$. The recovered bit $\bar{w}_i^*$ will always be 1. Since the adversary does not contain "0" in tracing position i, the probability that the adversary outputs "0" is less than $Adv_{CPA}^{\mathcal{A}}$, the probability that the adversary breaks the IND-WID-CPA game (so as to distinguish normal broadcasting ciphertext and tracing ciphertext).

- Case 2: All codewords held by the adversary contain "0" in tracing position i. As we proved in Theorem 1, the adversary will always output a random message other than m since the codewords do not match pattern P. The probability that the adversary outputs the right m is at most $1/|\mathcal{M}|$, where $|\mathcal{M}|$ is the number of messages in the message space. The recovered bit $\bar{w}_i^*$ will be 1 with probability at most $1/|\mathcal{M}|$;
- Case 3: The codewords held by the adversary contain both "0" and "1" in tracing position i. No matter whatever the adversary outputs, $\bar{w}_i^*$ must be in the feasible set of W.

Therefore, the final recovered codeword $\bar{w}^* \in F(W)$. 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 W is less than ϵ. Thus, the probability that the adversary breaks the property of t-collusion resistance of our generic traitor tracing scheme is less than $L \cdot Adv_{CPA}^{\mathcal{A}} + L/|\mathcal{M}| + \epsilon$. □

When $t = N$, our generic scheme is fully collusion resistant.

Theorem 3. *Our generic traitor tracing scheme is resistant against public collaboration.*

Proof: (Sketch Proof.) As we described in security requirements in Section 3, we will prove this property in 2 phases. First, we prove that incomplete public collaboration of traitors cannot generate useful keys for constructing pirate decoders. Then, we prove that public collaboration of complete key from a traitor will immediately lead to the accusation of the very traitor.

Phase 1. Each user's identity is a codeword of length L. The decryption key is generated for this very codeword. Incomplete public collaboration means a traitor releases part of decryption key in a public way, so this released part of decryption key is not valid in at least 1 bit position of the encryption pattern. As required by our encryption pattern $(*, \ldots, *)^L$, the decryption key must be valid in all L positions. Thus, this released part of decryption key is not useful for its original codeword. If several parts of incomplete decryption key are able to construct a useful pirate decoder, it presents a contradiction to Theorem 1, in which the adversary with many complete decryption keys cannot obtain information of message encrypted for other codewords, and the adversary without any decryption key cannot obtain information of message encrypted for any codeword.

Phase 2. If a user releases his/her decryption key as a whole, he/she may release the whole codeword as well. If the codeword is released, the tracer can immediately find out the identity by using the tracing algorithm of collusion secure code T and tracing key TK. If no codeword is released, the valid decryption key without the corresponding codeword can recover the correct message with probability 1/2. The tracer can treat the decryption key as imperfect pirate decoder and use the method described in the SubSection 4.4 to recover the codeword and then trace to the owner. □

4.4 Tracing Imperfect Pirate Decoders

Boneh and Naor [7] described a method to trace against imperfect pirate decoders. Their method can also be modified and applied in our scheme to enable such feature. The method is briefly described as follows.

Let δ be the probability that imperfect pirated decoder $\mathcal{D}$ fails to decrypt well-formed ciphertexts. Let $\mathcal{M}$ be the message space. $m' \leftarrow \mathcal{D}(C)$ denotes that decoder $\mathcal{D}$ outputs m' on input C. For $i = 1, \ldots, L$, the tracing algorithm is defined as follows:

repeat the following steps $\lambda \ln L$ times:

$$
\begin{aligned}
m &\xleftarrow{R} \mathcal{M} \\
P &\leftarrow (*, \ldots, \tbinom{1}{0}, \ldots, *), \ where\ P_i = \tbinom{1}{0} \\
c &\leftarrow WIBE.Encrypt(BK, P, m) \\
C &\leftarrow (i, c) \\
m' &\leftarrow \mathcal{D}(C)
\end{aligned}
$$

let p_i be the fraction of times that $m' = m$;

repeat the following steps $\lambda \ln L$ times:

$$
\begin{aligned}
m &\xleftarrow{R} \mathcal{M} \\
P^* &\leftarrow (*, \ldots, 1, \ldots, *), \ where\ P_i^* = 1 \\
c^* &\leftarrow WIBE.Encrypt(BK, P^*, m) \\
C^* &\leftarrow (i, c^*) \\
m' &\leftarrow \mathcal{D}(C^*)
\end{aligned}
$$

let q_i be the fraction of times that $m' = m$.

Define $\bar{w}_i^* \in \{0, 1\}$ as:

$$
\bar{w}_i^* = \begin{cases} 1 \ \text{if } q_i > 0 \\ 0 \ \text{if } q_i = 0 \text{ and } p_i > 0 \\ \text{`?' otherwise} \end{cases}
$$

and $\bar{w}^* = \bar{w}_1^* \ldots \bar{w}_L^*$. By using the δ-robust collusion secure code presented in [7] we obtain a tracing scheme that can trace imperfect pirate decoders as long as

$$\delta < (1/L) - (1/\lambda).$$

For details about tracing imperfect pirate decoders and constructing δ-robust collusion secure codes, please refer to [7].

4.5 Extension to Identifiable Parent Property Codes

The proposed generic scheme can be easily turned into a traitor tracing scheme based on identifiable parent property (IPP) codes against public collaboration. For a q-ary IPP code, what we need to do is to use IPP code as user's identity and allow q symbols in each hierarchical position in encryption, decryption and tracing algorithms, instead of (1, 0) for collusion secure codes. The number of tracing rounds may increase by a factor of $\log q$.

5 An Instance and Performance Evaluation

In this section, we present an efficient instance of traitor tracing scheme based on collusion secure code against public collaboration. Our instance is constructed from Boneh-Boyen-Goh WIBE scheme described in [1]. Then we compare its performance with several collusion secure code based tracing schemes.

5.1 An Instance Based on Boneh-Boyen-Goh WIBE

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be bilinear groups of some large prime order p and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$ be a bilinear map. We also assume that the messages to be encrypted are elements in $\mathbb{Z}_p$. The traitor tracing scheme works as follows:

- **Setup**(1^λ, N). A trusted party, given 1^λ and the number of users in the system N, selects $\epsilon \in (0, 1)$ and runs collusion secure code generation algorithm $\mathbf{G}(N, \epsilon)$ to generate a pair (Γ, 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 Γ. $\bar{w}^{(u)}$ is assigned to user u as the identity, with $1 \leq u \leq N$. A random generator $g \in_R \mathbb{G}_2$ and random number $\alpha \in_R \mathbb{Z}_P$ are selected. Let $g_1 = g^\alpha$. $H(\cdot) : \{1 \ldots L\} \times \{0,1\} \to \mathbb{Z}_p$ and $H_2(\cdot) : \mathbb{G}_T \to \mathbb{Z}_p$ are two collision resistant hash functions (the selection is out of scope of this paper). Random elements g_2, g_3, $h_1, \ldots, h_L$ are selected from $\mathbb{G}_1$. The master public parameters (also the broadcast-key BK) are (g, g_1, g_2, g_3, h_1, $\ldots$, h_L, $H(\cdot)$, $H_2(\cdot)$). g_2^α is kept private. For each identity $\bar{w}^{(u)}$ of user $u \in \{1, \ldots, N\}$, $\bar{w}^{(u)}$ can be expressed as $(\bar{w}_1^{(u)} \ldots \bar{w}_L^{(u)}) \in \{0,1\}^L$. The trusted party generates decryption key for each user u as:

$$SK_u = (d_{u,0}, d_{u,1}) = (g_2^\alpha \cdot (h_1^{H(1,\bar{w}_1^{(u)})} \ldots h_L^{H(L,\bar{w}_L^{(u)})} \cdot g_3)^r, g^r).$$

  Each decryption key is transferred to user over a secure and authenticated channel which is not considered in this paper.
- **Encrypt**(BK, m). Given the broadcast-key BK and a message $m \in \mathbb{Z}_p$, the sender selects random position $i \in \{1, \ldots, L\}$ to generate a pattern $P = (P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, \binom{1}{0}, \ldots, *)$. The senders selects $s \in_R \mathbb{Z}_p$, encrypts

m as follows:

$$\begin{aligned}
&C_1 = g^s;\\
&C_2 = (C_{2,1}, C_{2,0}) = ((g_3 \cdot h_i^{H(i,1)})^s, (g_3 \cdot h_i^{H(i,0)})^s);\\
&C_3 = m \oplus H_2(e(g_2, g_1)^s);\\
&C_4 = (C_{4,1}, \ldots, C_{4,i-1}, C_{4,i+1}, \ldots, C_{4,L})\\
&\quad = (h_1^s, \ldots, h_{i-1}^s, h_{i+1}^s, \ldots, h_L^s),
\end{aligned}$$

where $C_{2,1}$ is for $P_i = 1$, $C_{2,0}$ is for $P_i = 0$ and $C_{4,j}$ is for $P_j = *$ with each $j \neq i$. The ciphertext (i, C_1, C_2, C_3, C_4) is sent to all users.

– **Decrypt**(SK$_u$, C). Parses the SK$_u$ as ($d_{u,0}$, $d_{u,1}$) and the ciphertext C as (i, C_1, $(C_{2,1}, C_{2,0})$, C_3, $(C_{4,1}, \ldots, C_{4,i-1}, C_{4,i+1}, \ldots, C_{4,L})$), user u decrypts as follows:

$$\begin{aligned}
C_2' &= C_{2,\bar{w}_i^{(u)}} \cdot \prod_{j=1, j\neq i}^{L} (C_{4,j})^{H(j,\bar{w}_j^{(u)})}\\
&= (g_3 \cdot \prod_{j=1}^{L} h_j^{H(j,\bar{w}_j^{(u)})})^s;\\
T &= \frac{e(d_{u,0}, C_1)}{e(C_2', d_{u,1})}\\
&= \frac{e(g_2^\alpha (h_1^{H(1,\bar{w}_1^{(u)})} \ldots h_L^{H(L,\bar{w}_L^{(u)})} \cdot g_3)^r, g^s)}{e((g_3 \cdot \prod_{j=1}^{L} h_j^{H(j,\bar{w}_j^{(u)})})^s, g^r)}\\
&= e(g_2^\alpha, g^s)\\
&= e(g_2, g_1)^s;\\
m &= C_3 \oplus H_2(T).
\end{aligned}$$

– **Trace**$^{\mathcal{D}}$(TK). Given a perfect pirate decoder $\mathcal{D}$, the trusted party queries decoder $\mathcal{D}$ as a black-box oracle. For $i = 1, \ldots, L$, the tracing pattern is $P = (P_1, \ldots, P_i, \ldots, P_L) = (*, \ldots, 1, \ldots, *)$ with $P_i = 1$. The trusted party selects $s \in_R \mathbb{Z}_p$, a message m and a random element $R \in_R \mathbb{G}_1$ and acts as follows:

$$\begin{aligned}
&C_1 = g^s;\\
&C_2 = (C_{2,1}, C_{2,0}) = ((g_3 \cdot h_i^{H(i,1)})^s, R);\\
&C_3 = m \oplus H_2(e(g_2, g_1)^s);\\
&C_4 = (C_{4,1}, \ldots, C_{4,i-1}, C_{4,i+1}, \ldots, C_{4,L})\\
&\quad = (h_1^s, \ldots, h_{i-1}^s, h_{i+1}^s, \ldots, h_L^s),
\end{aligned}$$

where $C_{2,0} = R$ is for $P_i = 0$. (i, C_1, C_2, C_3, C_4) is fed to the decoder. If the pirate decoder outputs m, the trusted party decides that the decoder

contains a codeword $\bar{w}^*$ with $\bar{w}_i^* = 1$. Otherwise, $\bar{w}_i^* = 0$. When tracing on all positions is completed, the recovered codeword $\bar{w}^* = (\bar{w}_1^* \ \ldots \ \bar{w}_L^*)$ is put to tracing algorithm for collusion secure code $\mathbf{T}(\bar{w}^*, \mathrm{TK})$ to obtain a set of traitors $T \subseteq \{1, \ldots, N\}$.

5.2 Performance Evaluation

Table 1. Comparison with Previous Works

	Public Key	Private Key	Ciphertext Length	Encryption Computation	Decryption Computation	Public Collaboration Resistant
[18]	$2LG_1$	LZ_p	$2LG_1$	$L(2E_1+ 1M)$	$L(1E_1+ 2M)$	No
[10]	$1G_1+$ $(L+1)G_T$	LZ_p+ LG_1	$2G_1+ LG_T$	$LE_T+ 2M$	$L(M+ 2P)$	No
[7]	$2LG_1$	LZ_p	$1Z_p+ 2G_1$	$(2E+ 1M)$	$(1E+ 2M)$	No
[3]	$2LG_1$	LZ_p	$u(1Z_p+ 2G_1)$	$u(2E+ 1M)$	$u(1E+ 2M)$	No
[11]	$(2L+1)G_1$	$(L+2)Z_p$	$2Z_p+ (L+2)G_1$	$(3E+1M)$	$(2E+2M)$	No
Ours	$(L+2)G_1+$ $2G_2$	$1G_1+ 1G_2$	$(L+1)G_1+$ $1G_2+ 2Z_p$	$(L+3)E_1+ 2M$ $+ 1E_2 + 1E_T$	$(L+1)E_1+$ $LM+ 2P+$ $1M_T$	Yes

L: the length of codeword;
u: the number of codeword positions used in encryption [3];
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 $\mathbb{G}$;
E_1: exponentiation in $\mathbb{G}_1$; E_2: exponentiation in $\mathbb{G}_2$;
M: multiplication (or division) in $\mathbb{G}_1$;
M_T: multiplication (or division) in $\mathbb{G}_T$.

In Table 1 we compare our instance with several collusion secure code based schemes mainly on decryption key (storage cost for each user), ciphertext length, encryption computations, and decryption computations. We assume that all schemes use collusion secure codes of a same length L. Since schemes in [7] and [3] did not mention the public key encryption scheme used for each position, we suppose both schemes used secure ElGamal encryption scheme over a cyclic group of order p, where p is a large strong prime.

From Table 1, we notice that our scheme achieves constant private key storage cost for each user, while other schemes need storage cost linear to L, the length of codeword. We should notice that, our scheme only encrypt one message in each broadcasting and we can encrypt L messages by adding L-1 elements in $\mathbb{Z}_p$.

Our scheme is comparable to schemes [18, 10] employing *All-or-Nothing Transform* [25], where the decoder should include one valid key for each position

of the codeword in order to recover the final messages. *All-or-Nothing Transform* means that the decoder should contain one complete codeword of length L with at least L decryption keys covering all the positions in the codeword in order to decrypt all L parts of ciphertext, since the decoder obtains nothing if it fails to decrypt any part of ciphertext. However, *All-or-Nothing Transform* can only prevent deletion of keys from the pirate decoders but cannot prevent public collaborations.

We also notice that Schemes [7, 3, 11] are efficient in ciphertext length, encryption computation and decryption computation. It is because they encrypt the message using only one position [7, 11] (or constant number of positions [3]) in the broadcasting. However, it means that only one key is needed to construct a useful pirate decoder with decrypting probability of $1/L$ in schemes [7, 11], and u keys are needed to construct a useful pirate decoder with decrypting probability of $1/\binom{L}{u}$ in scheme [3], where $\binom{L}{u}$ denotes the number of ways of choosing u out of L positions. Hence, the ability of traitors to produce useful but untraceable pirate decoders is greatly increased.

6 Conclusion and Extension

Motivated by the idea of *identity-based encryption with wildcards* (WIBE), we describe a generic collusion secure codes based scheme secure against public collaboration. The construction idea also works for identifiable parent property codes. The instance proposed in Section 5 is comparable in efficiency to codes based traitor tracing schemes employing *All-or-Nothing Transform*. Thus, our scheme presents an answer to the problem left open by Billet and Phan [4].

From our idea described in SubSection 4.1, each WIBE scheme can be altered to obtain a code-based traitor tracing scheme, so we can obtain a scheme secure in standard model from the WIBE presented in [20].

To extend our scheme to adopt identifiable parent property codes, one can adjust the parameters to accept multiple alphabets in each hierarchical level.

It was stated by Billet and Phan in [3] as an open problem whether codes based tracing schemes can achieve revocation capabilities or not. We can solve this open problem by combining our traitor tracing scheme with an identity-based broadcast encryption scheme allowing revocation, since they both are special cases of identity-based encryption.

References

1. Abdalla, M., Birkett, J., Catalano, D., Dent, A., Malone-Lee, J., Neven, G., Schuldt, J., Smart, N.: Wildcarded identity-based encryption. Journal of Cryptology pp. 1–41 (2010)
2. Abdalla, M., Catalano, D., Dent, A.W., Malone-Lee, J., Neven, G., Smart, N.P.: Identity-based encryption gone wild. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP (2). Lecture Notes in Computer Science, vol. 4052, pp. 300–311. Springer (2006)

3. Billet, O., Phan, D.H.: Efficient traitor tracing from collusion secure codes. In: Safavi-Naini, R. (ed.) ICITS. Lecture Notes in Computer Science, vol. 5155, pp. 171–182. Springer (2008)
4. Billet, O., Phan, D.H.: Traitors collaborating in public: Pirates 2.0. In: Joux, A. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 5479, pp. 189–205. Springer (2009)
5. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: EUROCRYPT. pp. 440–456 (2005)
6. Boneh, D., Franklin, M.K.: An efficient public key traitor tracing scheme. In: CRYPTO. pp. 338–353 (1999)
7. Boneh, D., Naor, M.: Traitor tracing with constant size ciphertext. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM Conference on Computer and Communications Security. pp. 501–510. ACM (2008)
8. Boneh, D., Sahai, A., Waters, B.: Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: EUROCRYPT. pp. 573–592 (2006)
9. Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: ACM Conference on Computer and Communications Security. pp. 211–220 (2006)
10. Chabanne, H., Phan, D.H., Pointcheval, D.: Public traceability in traitor tracing schemes. In: EUROCRYPT. pp. 542–558 (2005)
11. Chen, Y.R., Tzeng, W.G.: A public-key traitor tracing scheme with an optimal transmission rate. In: Qing, S., Mitchell, C.J., Wang, G. (eds.) ICICS. Lecture Notes in Computer Science, vol. 5927, pp. 121–134. Springer (2009)
12. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: CRYPTO. pp. 257–270 (1994)
13. Fiat, A., Tassa, T.: Dynamic traitor training. In: CRYPTO. pp. 354–371 (1999)
14. Garg, S., Kumarasubramanian, A., Sahai, A., Waters, B.: Building efficient fully collusion-resilient traitor tracing and revocation schemes. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM Conference on Computer and Communications Security. pp. 121–130. ACM (2010)
15. Jin, H., Lotspiech, J., Nusser, S.: Traitor tracing for prerecorded and recordable media. In: Digital Rights Management Workshop. pp. 83–90 (2004)
16. Kiayias, A., Pehlivanoglu, S.: Pirate evolution: How to make the most of your traitor keys. In: CRYPTO. pp. 448–465 (2007)
17. Kiayias, A., Pehlivanoglu, S.: Tracing and revoking pirate rebroadcasts. In: Abdalla, M., Pointcheval, D., Fouque, P.A., Vergnaud, D. (eds.) ACNS. Lecture Notes in Computer Science, vol. 5536, pp. 253–271 (2009)
18. Kiayias, A., Yung, M.: Traitor tracing with constant transmission rate. In: EUROCRYPT. pp. 450–465 (2002)
19. Kurosawa, K., Desmedt, Y.: Optimum traitor tracing and asymmetric schemes. In: EUROCRYPT. pp. 145–157 (1998)
20. MING, Y., SHEN, X., WANG, Y.: Identity-based encryption with wildcards in the standard model. The Journal of China Universities of Posts and Telecommunications 16(1), 64–68, 80 (2009)
21. Mitsunari, S., Sakai, R., Kasahara, M.: A new traitor tracing. IEICE transactions on fundamentals of electronics, communications and computer sciences E85-A(2), 481–484 (2002-02-01)
22. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 2139, pp. 41–62. Springer (2001)
23. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. In: Financial Cryptography. pp. 1–20 (2000)

24. Park, J.H., Lee, D.H.: Fully collusion-resistant traitor tracing scheme with shorter ciphertexts. Designs, Codes and Cryptography pp. 1–22 (2010)
25. Rivest, R.L.: All-or-nothing encryption and the package transform. In: Biham, E. (ed.) FSE. Lecture Notes in Computer Science, vol. 1267, pp. 210–218. Springer (1997)
26. Stinson, D.R., Wei, R.: Combinatorial properties and constructions of traceability schemes and frameproof codes. SIAM J. Discrete Math. 11(1), 41–53 (1998)