Generic Construction of Trace and Revoke Schemes

Murat Ak*, Aggelos Kiayias †, Serdar Pehlivanoğlu ‡, Ali Aydın Selçuk§

May 6, 2013

Abstract

Broadcast encryption (BE) is a cryptographic primitive that allows a broadcaster to encrypt digital content to a *privileged* set of users and in this way prevent *revoked* users from accessing the content. In BE schemes, a group of users, called *traitor*s may leak their keys and enable an adversary to receive the content. Such malicious users can be detected through *traitor tracing* (TT) schemes. The ultimate goal in a content distribution system would be combining traitor tracing and broadcast encryption (resulting in a *trace and revoke* system) so that any receiver key found to be compromised in a tracing process would be revoked from future transmissions.

In this paper, we propose a generic method to transform a broadcast encryption scheme into a trace and revoke scheme. This transformation involves the utilization of a fingerprinting code over the underlying BE transmission. While fingerprinting codes have been used for constructing traitor tracing schemes in the past, their usage has various shortcomings such as the increase of the public key size with a linear factor in the length of the code. Instead, we propose a novel way to apply fingerprinting codes that allows for efficient parameters while retaining the traceability property. Our approach is based on a new property of fingerprinting codes we introduce, called *public samplability*.

We have instantiated our generic transformation with the BE schemes of [4, 13, 20] something that enables us to produce trace and revoke schemes with novel properties. Specifically, we show (i) a trace and revoke scheme with constant private key size and short ciphertext size, (ii) the first ID-based trace and revoke scheme, (iii) the first publicly traceable scheme with constant private key size and (iv) the first trace and revoke scheme against pirate rebroadcasting attack in the public key setting.

Keywords: Digital rights management, broadcast encryption, traitor tracing, fingerprinting codes.

*Department of Computer Engineering, Bilkent University, 06800, Ankara, Turkey email: muratak@cs.bilkent.edu.tr

†Department of Computer Science and Engineering, University of Connecticut, Storrs, CT, USA email: aggelos@cse.uconn.edu

‡Department of Computer Engineering, Zirve University, Gaziantep, Turkey email: serdar.pehlivanoglu@zirve.edu.tr

§Department of Computer Engineering, Bilkent University, 06800, Ankara, Turkey email: selcuk@cs.bilkent.edu.tr

1 Introduction

In a digital content distribution setting, the content is encrypted such that the intended authorized users, having access to the decryption keys, are capable of receiving the transmission. However, this might not be sufficient to achive an adequate level of access control. Indeed it may be required to revoke on the fly a subset of receivers from a certain transmission. Systems with such capability have been referred to as *broadcast encryption* by Fiat and Naor [18].

A shortcoming of such schemes in general is the possibility of the illegal redistribution of the content by some authorized receivers. This can be possible by producing a malicious decoder that circumvents the access control used by the content distribution system. Following the standard terminology, the decoder created by an adversary is called a pirate decoder, the users that divulge their keys to the adversary are called traitors and such keys are called traitor keys. The sender may want to restrict this type of behavior since such adversarial behavior introduces unauthorized receivers in the system. Traitor tracing is a deterrence mechanism where an authority is capable of performing a forensic analysis against any working pirate decoder and through the analysis recover at least one of the traitor keys that was used in its construction. Traitor tracing emerged first in the work of Chor, Fiat and Naor [10] as a solution to this problem.

We categorize the traitor tracing mechanism as non-black box tracing if it is possible to extract the key-information from the decoder through reverse engineering techniques. Such examples in the literature include [3, 29]. In many settings, the non-black box approach is inapplicable for many possible reasons, e.g. it may be expensive or can be deterred through obfuscation, or the tracer has remote access to the decoder. We categorize the mechanism as black-box if the tracing authority interacts with the pirate decoder in a black-box manner: querying the decoder and observing the response of the decoder. The majority of the works, [2, 6, 8, 10, 17, 24, 28, 31], in the traitor tracing literature support black-box tracing.

Trace and Revoke Schemes: The ultimate goal in a content distribution system would be combining traitor tracing and broadcast encryption so that any receiver key found to be compromised in a tracing process would be revoked from future transmissions. This is introduced by Naor and Pinkas in [33]. However, it is not possible to achieve this trivially, and a naive combination of both mechanisms would severely fail as discussed in the many subsequent works [9, 26, 32]. The subset cover framework of [32] leads to a number of schemes [21, 22] which rely on combinatorial structures and support somewhat weak tracing in the symmetric setting (the tracing is not guaranteed to identify a traitor but rather disables the pirate decoder). This weakness leads to a new type of attack called Pirate Evolution in [25]. Further work on trace and revoke schemes includes the public-key schemes of Boneh et.al [9] and Furukawa and Attrapadung [16].

Tracing and Revoking Pirate Rebroadcasts: Beyond the above pirate decoder

attack, any content distribution system is vulnerable to the more serious attack of rebroadcasting: in a pirate rebroadcast the adversary instead of producing a malicious decoder it simply publishes the content. Evidently, this defeats any mechanism that requires an interaction with the pirate decoder with some specially designed ciphertexts like the above mechanisms we discussed so far. Pirate rebroadcasting is introduced as an attack concept by Fiat and Tassa [19] and further studied in [36]. Needless to say, merely tracing pirate rebroadcasts is of little use by itself and one should be able to revoke the involved traitor keys.

A trace and revoke scheme that is able to guard against pirate rebroadcasts is implemented as part of the AACS standard [1]. The security and performance of this scheme was analyzed by Jin and Lotspiech [23] with further analysis performed in [26] by Kiayias and Pehlivanoglu that revealed some limitations of that construction. In [26], tracing and revoking pirate rebroadcasting was formally modeled and a scheme for tracing and revoking an unlimited number of users was introduced. This is the only efficient trace and revoke scheme available and is restricted to the symmetric case setting with an underlying combinatorial key-distribution method based on subset cover framework.

Public Traceability: In Eurocrypt 2005, Chabanne, Phan and Pointcheval [11] introduced the notion of public traceability where tracing requires no secrets. A two user solution was presented in [11] and further improved to the multiuser setting with short transmissions in [17] and [34] by employing fingerprinting codes. However, the public key and the private key sizes are all linear in length of the fingerprinting code employed for key distribution. The trace and revoke scheme of [9] is also publicly traceable with shorter key sizes, i.e. $O(\sqrt{n})$; still the scheme requires high bandwidth, i.e. it has a ciphertext length of $O(\sqrt{n})$.

Technical Background: The majority of the black-box traitor tracing schemes share the same tracing strategy that is called 'hybrid coloring' in [24] or 'linear tracing' in [28] and is inherent in almost all black-box traitor tracing mechanisms. This strategy can be summarized in the following fashion: The pirate decoder is queried with a sequence of special tracing ciphertexts that are gradually randomizing the way receivers decrypt. In this sequence, while the first special ciphertext is decryptable by all receivers, the last one is decryptable by none. In between, a 'walking procedure' is executed where the i-th type of tracing ciphertext disables the first i receivers in decrypting the transmission. This is repeated many times to approximate the success rates of the decoder in decrypting each type of tracing ciphertext. Finally, the traitor key used in the construction of the pirate decoder is inferred by an analysis of the success rates.

This technique yields a trivial traitor tracing system with each user having a unique decryption key. The ciphertext size would be very high (as much as $n/2$) on average and n in the worst case. For better trade-offs, the same technique can trivially be applied over more flexible key-distribution methods like the schemes based on fingerprinting codes [10, 24] or combinatorial structures [21, 22, 32]. In the

public key setting, a number of tracing schemes (e.g. [8, 9, 16, 31]) also build their tracing strategies on the 'linear tracing' technique: the pirate decoder is queried with specially crafted tracing ciphertexts that enables the walking procedure. The difficulty of designing such a scheme is illustrated in the example of [31] which is shown vulnerable independently by [27] and [30].

Fingerprinting codes[7, 10, 38] are one of the basic mathematical tools in the design of tracing mechanisms. The fingerprinting codes, in the context of tracing, have been used (in almost all of the schemes they are employed including but not limited to [2, 6, 10, 17, 23, 24, 34, 36]) to perform key-distribution so that each receiver gets a unique set of keys.

1.1 Our Goal

Recently, new applications of fingerprinting codes have been introduced in [26, 28], where the code is imposed on the interaction of the tracer and the pirate decoder to observe the way the decoder responds back. This is a quite different approach compared to the conventional use of fingerprinting codes for individualizing each receiver during key-distribution (as in the case of virtually all earlier works we cited above). This new application of fingerprinting codes leads to strong results: [26] introduces the first trace and revoke system against pirate rebroadcasts with unlimited number of traitors and revocations and [28] introduces a faster tracing strategy that can be used to replace the linear tracing strategy cited above.

Inspired by the above results, the crux of our design is that we partition the enabled set of users into a number of subsets (let us denote the size of the partition by q). A broadcast encryption is prepared for each subset resulting in q different encryptions that makes up the regular transmission. In tracing ciphertexts, we choose the partitions based on a q-ary fingerprinting code and apply the standard tracing strategies (for instance the linear tracing strategy that progressively randomizes some of the ciphertexts) to locate a subset which contains a traitor. Applying this basic step over the length of the code will identify a traitor key used in the pirate decoder by utilizing the fingerprinting code tracing algorithm.

There is a subtle challenge related to the design idea above: the statistical difference between the choice of partitions in both regular and tracing transmissions should be negligible so that the pirate decoder will not become aware of tracing. A trivial attempt would be using the same fingerprinting code in the regular transmission. The downside of this approach is that it requires the generated fingerprinting code to be part of the public key which will inflate the public key size with an additional $O(n \cdot \ell)$ overhead (where n is the number of users and ℓ is the length of the code)

Our solution is to prepare the regular transmission through a sampling algorithm that simulates the code and partitions the set of enabled receivers in such a way that is indistinguishable from the partition in a tracing transmission. Towards this goal, we introduce the concept of the public samplability of a fingerprinting code and

prove that the open Chor-Fiat-Naor fingerprinting code [10] is publicly samplable and thus suitable to be employed in our generic design. Formally, we say that a q-ary fingerprinting code `F` is *publicly samplable* by `Z()`, if the sampler `Z()`, for any n and any fixed index j in the range $1, 2, \ldots, \ell$, samples a partition for subset $S \subseteq [n]$ that is statistically indistinguishable from a partition based on the j-th column of the code generated by `F`.

1.2 Our Results

The present work has the following major contributions:

1. We present a generic transformation of a broadcast encryption scheme into a trace and revoke scheme. The transformation preserves the public and private key sizes of the underlying scheme while expands the ciphertext length a q factor that is related to the traitor coalition size the scheme will be resistant to.

As it is evident in the following Table 1 where we give three instantiations of our generic transformation applied to the BE schemes of [4, 13, 20] with the use of open Chor-Fiat-Naor code of [10], our results outperform the existing trace and revoke schemes of [9, 16]. In particular, we obtain the first trace and revoke scheme with constant private key size in the standard model. The scheme of [16] can be proven in generic group model[1]

Note that the schemes of [15] and [35] support a weaker traceability (they do not guarantee to identify a traitor but rather disable the pirate decoder) along the lines of the subset cover framework based tracing and revoking [32]. As mentioned such schemes are susceptible to "pirate evolution attacks" and we exclude them from the comparison.

Trace&Revoke	Public Key Size	Private Key Size	Ciphertext Size	Security & Type
`BW`[9]	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	Adaptive
`FA`[16]	$O(n)$	$O(1)$	$O(\sqrt{n})$	Ad/Generic GM
Our Results				
`T&R-BGW1[4]`	$O(n)$	$O(1)$	$O(w^2)$	Static
`T&R-BGW2[4]`	$O(\sqrt{n})$	$O(1)$	$O(w^2\sqrt{n})$	Static
`T&R-Del1[13]`	$O(n)$	$O(1)$	$O(w^2)$	Static/ID-based
`T&R-Del2[13]`	$O(\sqrt{n})$	$O(1)$	$O(w^2\sqrt{n})$	Static/ID-based
`T&R-GW1[20]`	$O(m)$	$O(1)$	$O(w^2)$	Semi-Static
`T&R-GW2[20]`	$O(n)$	$O(1)$	$O(w^2)$	Ad/ROM/ID-based
`T&R-GW3[20]`	$O(\sqrt{n})$	$O(1)$	$O(w^2\sqrt{n})$	Ad/ID-based

Table 1: m is a bound on the number of recipients in a single broadcast and w is the number of traitors.

[1] For more information about the generic group model as an assumption see [14]

2. Of particular interest, the generic construction instantiated by [13] and [20] yields the first identity based trace and revoke scheme against both static and adaptive adversary. Recall again that the ID-based scheme of [35] supports a weaker tracability, hence we do not consider it for a comparison in here.

3. We define, for the first time, the concept of the public samplability of a fingerprinting code which is crucial in the design of our construction. This also highlights an advantage of open fingerprinting codes over secret codes despite the fact that secret codes like [7, 38] are shorter.

4. The publicly traceable schemes of [17] and [34] have private and public-keys proportional to the length of the fingerprinting code. The trace and revoke scheme of [9] also supports public tracing but still the private key size and the ciphertext length is a function of the number of users. Our generic construction does not require any tracing secret key, hence supports fully public traceability as well as revocation. This gives the first publicly traceable schemes with constant private key sizes while achieving short transmissions as a function of the number of traitors (proportional to the alphabet of the underlying fingerprinting code).

5. In [26], tracing and revoking pirate rebroadcasting was formally modeled and a scheme for tracing and revoking an unlimited number of users was shown. This is the only known efficient trace and revoke scheme, but restricted to the symmetric case with an underlying combinatorial key-distribution method based on the subset cover framework.

Our generic construction, by adapting the way the ciphertext is prepared, fulfills the need for tracing and revoking pirate rebroadcasts in the public key setting. The instantiations provided in the table presented above would work smoothly in this setting as well leading to a number of schemes suitable for tracing and revoking pirate rebroadcasts with the efficiency parameters and security types stated.

2 Preliminaries and Definitions

2.1 Broadcast Encryption

A broadcast encryption (BE) scheme is a method for encrypting messages in a way that only a set of privileged users will be able to decrypt it, and even if all revoked users collude, they cannot get any information about the message.

In a typical content distribution setting where broadcast encryption is used, a hybrid approach of encryption is performed: the BE scheme is used to broadcast a cryptographic key which further will be used to encrypt the actual message with a standard symmetric key encryption scheme. This hybrid approach is called as KEM-DEM mechanism (Key Encapsulation-Data Encapsulation) in the literature (cf. [12]).

A BE scheme in the KEM setting consists of three algorithms (`KeyDist`, `Encrypt`, `Decrypt`): In this paper we will denote the set of all users $\{1, 2, \ldots, n\}$ by $[n]$.

- `KeyDist`(1^n) generates private keys sk_i for users $i \in [n]$ and a public key PK.
- `Encrypt`(PK, S) prepares a header hdr and a key K for receiver set $S \subseteq [n]$.
- `Decrypt`(PK, sk_i, S, hdr), using the private key sk_i, decrypts the header hdr to retrieve the key K.

The pair $\langle S, hdr \rangle$ is called full header and transmitted as a broadcast cipher in all of the broadcast encryption schemes we included in Table 1. Hence, a receiver will have access to the information of the enabled set S to run the decryption algorithm.

2.1.1 Correctness

A BE scheme in the KEM model is *correct*, if a privileged user can recover the key K by decrypting the header hdr. Formally stated, a BE scheme is correct if $\forall PK, \forall S \subseteq [n], \forall i \in S$, `Decrypt`$(PK, sk_i, S, hdr) = K$, whenever $(PK, sk_1, \ldots, sk_n) \leftarrow$ `KeyDist`(1^n) and $(hdr, K) \leftarrow$`Encrypt`(PK, S).

2.1.2 Broadcast Security

We say the scheme satisfies the broadcast security if a user can recover the key K only if it is in the privileged set, i.e. non-revoked. The formalization of this security concept is through the following security game: an adversary is given the keys of all non-revoked users of his choice, and challenged with a header and key pair. In one world, the pair is a valid pair produced to revoke chosen set of users, while in the other world the key is replaced with a random key. The goal of the adversary is to guess the world he is playing in. We consider the IND-CCA attack scenario in the KEM setting:

Game 1 (Broadcast KEM-IND-CCA Game) *Both polynomial time adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ are given the number of users, n.*

- ***Initialize.*** *$\mathcal{A}$ chooses a subset $S^* \subset [n]$.*
- ***Setup.*** *$\mathcal{C}$ runs* `KeyDist`(1^n) *to generate private keys $sk_1, \ldots, sk_n$ and the public key PK. The challenger $\mathcal{C}$ sends the public key PK and sk_i for $i \notin S^*$ to $\mathcal{A}$.*
- ***Decryption Queries.*** *$\mathcal{A}$ makes polynomially many queries of the form (i, S, hdr) where $i \in S$ and $S \subseteq S^*$. $\mathcal{C}$ responds with $K \leftarrow$* `Decrypt`(S, PK, sk_i, hdr).
- ***Challenge.*** *$\mathcal{C}$ runs algorithm* `Encrypt`(PK, S^*) *and obtains (hdr^*, K^*). The challenger randomly chooses a key K' from the symmetric key space $\mathcal{K}_{\mathcal{SYM}}$ and sets $K_0 = K^*$ and $K_1 = K'$. For a randomly chosen bit $b \in \{0, 1\}$, the challenger send the challenge string (hdr, K_b) to the adversary $\mathcal{A}$.*

- ***Guess.*** *$\mathcal{A}$ guesses b' and wins if $b' = b$.*

Definition 1 *A broadcast encryption scheme B is ϵ-secure in the KEM-IND-CCA model if for any polynomial time attacker $\mathcal{A}$ we have*

$$Adv_{\mathcal{A}} = |Pr[\mathcal{A} \text{ wins}] - 1/2| = \left|Pr[b' = b] - 1/2\right| \leq \epsilon$$

where $Adv_{\mathcal{A}}$ denotes the advantage of the attacker $\mathcal{A}$ for winning the security game described above.

Observe that in Game 1, the adversary chooses the set it will attack before getting the public key. The security against such adversarial model is called static security as in the schemes of [4] and [13]. Unlike this model, the attacker may wait until the public key is published. It selects the target set S^* and requests the private keys of users not in the set S^* after the public key is available. This is called adaptive attack (see [20] for a construction that satisfies adaptive security) and the above security game should be adapted accordingly. We can think of another version for the ID-based broadcast encryption schemes where there is no pre-defined user set. In such a setting, the attacker usually chooses the set of IDs it wishes to attack at the beginning. Since there is no significant differences between the security games of standard public key schemes and ID-based schemes, we omit the details of the game for ID-based setting in here.

2.2 Trace and Revoke

We consider tracing and revoking in the black-box model, where the adversary creates a pirate decoder. In order for the tracing algorithm to identify a traitor we need to make a necessary assumption that the pirate decoder succeeds in decrypting ciphertexts intended for at least one subset with a non-negligible probability. Otherwise, it is theoretically impossible to assert any tracing capability since it is trivial to construct such a decoder without any decryption keys. Therefore, throughout the paper, we say a decryption box is a (σ, S)-pirate (or σ-decoder) if its rate of correctly decrypting broadcasts to set S is at least σ. We denote such a pirate decoder by $\mathcal{D}_S^\sigma$. Upon encountering a decoder, we will assume that S is known to the tracer. This is a reasonable assumption that holds for almost all existing trace and revoke schemes in the literature like in [9, 15, 16]. A working pirate decoder eventually will also reveal its σ value which can be approximated by the tracer. Hence, from now on we will assume that both S and σ can be extracted from the description of $\mathcal{D}_S^\sigma$:

A T&R system consists of the following four algorithms:

- `Setup(`1^n`)` generates private keys $sk_1, \ldots, sk_n$, public key PK and possibly a tracing key TK.
- `Transmit(`PK, S`)` prepares a header hdr and a symmetric key K.

- $\texttt{Receive}(PK, sk_i, S, hdr)$, using the private key sk_i, decrypts the header hdr to retrieve the key K. It will be successful if and only if user i is in the set S.

- $\texttt{Trace}(S, \mathcal{D}_S^\sigma, PK, TK)$ identifies a set of traitors, denoted by $A \subseteq S$, whose key(s) must have contributed in the construction of the pirate decoder $\mathcal{D}_S^\sigma$.

We again call the pair $\langle S, hdr \rangle$ full header. The trace and revoke schemes of [9, 16], that we included in Table 1 for comparison, transmit the full header as a broadcast. Hence, the receivers will be able to access the information of S to run the decryption algorithm.

Black Box Tracing: In this paper, we consider black-box tracing against resettable (i.e., the decoder does not maintain state during the tracing process) and available (i.e., the decoder remains available as long as the tracing process wishes to experiment with it.) pirate decoders. In the literature, almost all of the positive results in designing traitor tracing schemes (including the schemes that we compare to our constructions) are successful against such decoders.

Correctness and security definitions for T&R schemes are the same as their BE counterparts. So we skip them here. There is one additional property for T&R systems, though, which is traceability. Traceability is defined via the following game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$:

Game 2 (Tracing Game) *Both $\mathcal{A}$ and $\mathcal{C}$ are given the number of users, n, and the upper bound t on traitor coalition size.*

- ***Request.*** *$\mathcal{A}$ chooses a traitor subset T of size at most t and requests their private keys from $\mathcal{C}$.*

- ***Provide.*** *$\mathcal{C}$ runs $\texttt{Setup}(1^n)$ to obtain the keys. Then, $\mathcal{C}$ sends all sk_i such that $i \in T$ and the public key PK to $\mathcal{A}$. It keeps the tracing key TK.*

- ***Forge decoder.*** *$\mathcal{A}$ chooses a set S, and creates a resettable and available (σ, S)-pirate decoder box $\mathcal{D}_S^\sigma$ which, by definition, correctly decrypts the broadcasts to set S with probability at least σ. It outputs $\mathcal{D}_S^\sigma$.*

- ***Trace.*** *The challenger $\mathcal{C}$ runs $\texttt{Trace}(S, \mathcal{D}_S^\sigma, PK, TK)$ and outputs a set $A \subseteq S$ that is accused of containing a traitor.*

We say that the attacker $\mathcal{A}$ wins the game if the set A is empty or it is not a subset of T. Having this definition, we say that $\texttt{T}= (\texttt{Setup}, \texttt{Transmit}, \texttt{Receive}, \texttt{Trace})$ is a T&R scheme with tracing success probability α against t-coalition σ-pirates if no polynomial time attacker $\mathcal{A}$, forging a σ-decoder by corrupting a traitor coalition of size t, can win the game described above with probability more than $1 - \alpha$.

2.3 Fingerprinting Codes

A codeword x over an alphabet Q is an ℓ-tuple $x[1], \ldots, x[\ell]$ where $x[i] \in Q$ for $1 \leq i \leq \ell$. We call a set of codewords $\mathcal{C} \subseteq Q^\ell$ with size n by (ℓ, n, q)-code given $|Q| = q$. Each codeword x in an (ℓ, n, q)-code can be considered as providing a unique way of accessing to some specific object or functionality. In such setting, an adversary is modeled as corrupting a number of users and retrieving their codewords. The adversary, then, runs a `Forge` algorithm that produces a non-user codeword $p \in Q^\ell$ that provides an access to the same functionality. This codeword is called pirate codeword.

In the context of forgery, the set $desc(\mathcal{C}_T) = \{x \in Q^\ell : x[i] \in \{a[i] : a \in \mathcal{C}_T\}, 1 \leq i \leq \ell\}$ is called the descendent set of $\mathcal{C}_T \subseteq \mathcal{C}$ where $x[i], a[i]$ are the i-th symbols of the related vectors. So, piracy inside an (ℓ, n, q)-code $\mathcal{C}$ is equivalent to producing a valid pirate codeword $p \in desc(\mathcal{C}_T)$ out of the codewords available to a traitor coalition T. Such restriction on the pirate codeword production is called '*marking assumption*' and it holds in any reasonable piracy setting (including all the related works we refer in this work).

Fingerprinting codes are defined by two algorithms, `CodeGen` and `Identify`. `CodeGen`(1^n) outputs a pair $(\mathcal{C}, tk)$ where $\mathcal{C}$ is an (ℓ, n, q)-code with alphabet Q such that $|Q| = q$, and tk is a key for identifying purposes which can also be empty. `Identify`$(\mathcal{C}, tk, p)$, on observation of a pirate codeword p, outputs either $\perp$ or a codeword index t which is supposed to be the index of a corrupted user. The performance of fingerprinting codes is evaluated according to their capability of identifying traitor codewords.

Definition 2 *We say that a q-ary fingerprinting code* (`CodeGen`, `Identify`) *is an (α, w)-identifier if the following holds: Given $(tk, C) \leftarrow$* `CodeGen`(1^n)*, and a* `Forge` *algorithm satisfying marking assumption,*

$$\forall T \subseteq U \ s.t. \ |T| \leq w$$

$$Pr\left[\emptyset \subsetneq \texttt{Identify}(\mathcal{C}, tk, p) \subseteq T\right] \geq 1 - \alpha$$

where $p \leftarrow$ `Forge`(C)*.*

If the fingerprinting code provides a traitor identification in the above setting, where the generated code $\mathcal{C}$ is not kept hidden from the `Forge` algorithm (note that the disclosure of $\mathcal{C}$ does not contradict with the marking assumption since the piracy is made possible through the marks available to the pirate.), then we call it open fingerprinting code. If the `Forge` algorithm is restricted to the information of $C_T = \{c_j | j \in T\}$ with $\mathcal{C} = \{c_1, \ldots, c_n\}$, then we call the fingerprinting code secret code. The binary fingerprinting codes of Boneh-Shaw [7] and Tardos [38] codes are examples of secret codes, while an open fingerprinting code is introduced in [10]. We further say that the code is (i) *w-identifier* fingerprinting code if the failure probability $\alpha = 0$ holds and (ii) fully collusion resistant fingerprinting code if $w = n$ holds.

3 Generic Trace and Revoke scheme

In our generic construction, the idea is to transform a broadcast encryption scheme (BE scheme) into a trace and revoke scheme. The message, to be broadcasted, is transmitted to an enabled set S as follows: a partition of $S = \{S_1, S_2, \ldots, S_q\}$ is first computed and the message is encrypted for each subset S_i separately using the broadcast encryption scheme. This transformation preserves the revocation capability of the underlying broadcast encryption scheme.

A tracing mechanism can be coupled with the above transformation by following a two-step strategy: (i) we compute different partitions of the enabled set and (ii) for each partition, we query the pirate decoder with a sequence of specially designed tracing ciphertexts to find and mark a subset as containing a traitor (such technique can be found in [24, 28, 32] and will be explained later in the section). At the end of this strategy, we collect a sequence of subsets which are marked as containing a traitor. If the choice of the partitions are based on a fingerprinting code, the collected information leads to the identification of a traitor.

However, there is a subtle challenge related to the tracing idea above: the statistical difference between the choice of partitions in both regular and tracing ciphertexts should be negligible so that the pirate decoder will not become aware of tracing. A trivial attempt would be partitioning the subset S according to the same fingerprinting code that we use in tracing. This will ensure the structural indistinguishability of the regular transmission from the tracing transmission. The downside of this approach is that it requires the generated fingerprinting code to be part of the public key which will inflate the public key size with an additional $O(\ell \cdot n)$ overhead where ℓ is the length of the code and n is the number of receivers.

Our solution to that challenge is to prepare the regular transmission through a a sampling algorithm that simulates the code and partitions the set of enabled receivers in such a way that is indistinguishable from the partition in a tracing transmission. Towards this goal, we introduce the concept of the public samplability of a fingerprinting code:

Definition 3 *Let* $\mathtt{F} = (\ \mathtt{CodeGen}, \mathtt{Identify}\ )$ *be a* q*-ary fingerprinting code over an alphabet* $Q = \{1, 2, \ldots, q\}$*. We consider a sampling algorithm* $\mathtt{Z}$ *that, on input* n *and some auxiliary information* aux*, samples a partition* $V = \{V_1, \ldots, V_q\}$ *for set* $\{1, 2, \ldots, n\}$*.*

We say $\mathtt{F}$ *is* publicly samplable *by* $\mathtt{Z}(1^n, aux)$ *with* ϵ *probability of failure, if the distribution for* V *is statistically indistinguishable from the distribution of* $S^* = \{S_1^*, \ldots, S_q^*\}$ *with probability at least* $1-\epsilon$ *where* S^* *is defined over the choice of (i) an* (ℓ, n, q) *code* $\mathcal{C} = \{c_1, \ldots, c_n\}$ *generated by* $\mathtt{CodeGen}(1^n)$ *and (ii) the column-index* $j \in [\ell]$ *such that*

$$S_s^* = \{i \in [n] : c_i[j] = s\}, \quad \text{for} \quad s \in [q]$$

We are now ready for the formal description: Let $\mathtt{B}$ be a BE scheme consisting of three algorithms $\mathtt{BKeyDist}(1^n)$, $\mathtt{BEnc}(PK_{\mathtt{B}}, S)$, and $\mathtt{BDec}(PK_{\mathtt{B}}, sk_i, S, hdr)$, we design the key distribution algorithm of our generic scheme $\mathtt{T}$ as follows:

- $\mathtt{TRKeyDist}(1^n)$ The algorithm runs the key distribution algorithm $\mathtt{BKeyDist}(1^n)$ of $\mathtt{B}$. This will produce a public key $PK_{\mathtt{B}}$ and a set of private keys sk_i, $i \in [n]$, which will be distributed to the receivers. The algorithm further chooses a symmetric encryption scheme, $\mathtt{Sym} = (\mathtt{SEnc}, \mathtt{SDec})$, and a description of a fingerprinting code $\mathtt{F} = (\mathtt{CodeGen}, \mathtt{Identify})$ that is *publicly samplable* by $\mathtt{Z}(1^n, aux)$. We note that the actual codewords are not generated at this moment. Hence we do not require any tracing key while the algorithm $\mathtt{Z}$ and aux will be published as part of the public key $PK_{\mathtt{T}} =< 1^n, PK_{\mathtt{B}}, \mathtt{Sym}, \mathtt{F}, \mathtt{Z}, aux >$.

We design our transmission algorithm to be employed as a KEM mechanism, i.e. there is no message as input, instead a random key K is chosen to be transmitted which will next serve as a key in a later step that is called the data encapsulation mechanism (DEM) step.

- $\mathtt{Transmit}(PK_{\mathtt{T}}, S)$ The algorithm first choses a random key K to be transmitted and a partition $\{V_1, V_2, \ldots, V_q\}$ of $[n]$ is sampled by the algorithm $\mathtt{Z}(1^n, aux)$. It sets $S_i = V_i \cap S$ for each $i = 1, \ldots, q$. The transmission algorithm then runs the encryption algorithm of the BE scheme for each subset and broadcasts the message $c = (c_1||c_2|| \ldots ||c_q)$ where, for each $i = 1, \ldots, q$, we have $c_i = hdr_i||e_i$ and

$$(hdr_i, K_i) \leftarrow \mathtt{BEnc}(PK_{\mathtt{B}}, S_i), \quad e_i \leftarrow SEnc_{K_i}(K)$$

Remark: in some broadcast encryption schemes that support key-encapsulation (e.g. the scheme of [9]), the broadcaster has no control on the choice of the message (K_i's in our construction) transmitted. For this reason, we can not force the same K to be produced by the broadcast encryption $\mathtt{BEnc}(PK_{\mathtt{B}}, S_i)$ for each i. As a remedy for this issue, we encrypt the randomly chosen key K with the symmetric encryption scheme $\mathtt{Sym}$ under the keys $K_1, \ldots, K_q$. This solution will make our transformation applicable to any broadcast encryption.

We next describe the $\mathtt{Receive}$ algorithm:

- $\mathtt{Receive}(PK_{\mathtt{T}}, sk_j, \{S_i\}_{i\in[q]}, c)$ The j-th user parses the public key $PK_{\mathtt{B}}$ from $PK_{\mathtt{T}}$. It locates the index $k \in [q]$ for which $j \in S_k$ holds and parses $hdr_k||e_k$ from $(c = c_1||c_2|| \ldots c_q)$. Then it uses the decryption function of $\mathtt{B}$ to decrypt hdr_k and retrieves the key K as follows:

$$K^* = \mathtt{BDec}(PK_{\mathtt{B}}, sk_j, S_k, hdr_k), \quad K = SDec_{K^*}(e_k)$$

In the description above, we deviate from the original definition of the `Receive` algorithm by inputting the partition $\{S_i\}_{i\in[q]}$ in replace of the set S. This is a simple syntactic modification as the underlying broadcast encryptions (at least those from the Table 1 that we apply the transformation) require the full headers $\langle S_i, hdr_i\rangle$, for $i \in [q]$, to be transmitted.

As part of the tracing mechanism, a q-ary fingerprinting code $\mathcal{C} = \{c_1, \ldots, c_n\}$ of length ℓ is produced by running $\mathcal{F}.\mathtt{CodeGen}(1^n)$. Instead of sampler $\mathtt{Z}(1^n, aux)$, the generated code will be used to partition the enabled set S. We compute ℓ different partitions: the j-th type of partition, denoted by S_j, is associated with the j-th column of the code. More specifically, we set $S_{j,i} = S \cap \{k : c_k[j] = i\}$ and compute $S_j = \{S_{j,1}, S_{j,2}, \ldots, S_{j,q}\}$ to be the partition.

The tracing algorithm proceeds by applying a standard tracing strategy (this strategy is called 'hybrid coloring' in [24] or 'linear tracing' in [28]): the tracing center prepares tracing ciphertexts such that some subsets may fail to decrypt the transmission. In this direction, a tracing ciphertext of type (j, v), denoted by $\mathtt{Transmit}(PK_{\mathtt{T}}, S, \mathcal{C}, j, v)$, has the following characteristics: (i) S_j is computed to be the partition and (ii) the first v ciphertexts out of q are substituted with encryption of random messages.

While the tracing transmission of type $(j, 0)$ can be decryptable by all users, those of type (j, q) totally hides the information of the message encrypted. This suggests that the tracer can progressively randomize the pattern of the ciphertext until a position s is identified such that the tracer substantially differentiates the way the pirate decoder responds to the tracing transmissions of type $(j, s-1)$ and (j, s). Due to the security claims of underlying schemes, we can conclude the existence of a traitor in set $S_{j,s}$, i.e. a pirate mark of s is observed for the j-th column. We repeat this tracing transmissions for $j = 1$ to $j = \ell$ and produce a pirate codeword w. Finally, we run $\mathcal{F}.\mathtt{Identify}$ algorithm to output a user index that is responsible for the acts of the pirate decoder.

A formal description of the tracing algorithm is given below. For the simplicity, we consider tracing against the pirate decoders of type $\mathcal{D}_S^1$. Such a decoder is called perfect decoder as it correctly decrypts all well-formed ciphertexts. In reality, the pirate may be content with a decoder that works only a fraction of the time, that is formulated in our definition as σ-pirate with $\sigma \leq 1$. A solution for $\sigma < 1$ values will be discussed later in Section 4.1.

- $\mathtt{Trace}(S, \mathcal{D}_S^1, PK_{\mathtt{T}}, TK)$ first parses $1^n, PK_{\mathtt{B}}$, `Sym` and the description of `F` from $PK_{\mathtt{T}}$. It produces a q-ary code $\mathcal{C} = \{c_1, \ldots, c_n\} \leftarrow \mathcal{F}.\mathtt{CodeGen}(1^n)$ and initializes a pirate codeword $w \leftarrow 0^\ell$. Denoting the length of the code $\mathcal{C}$ by ℓ, the tracing algorithm repeats the following sub-procedure for each $j = 1, \ldots, \ell$:

  Create a partition $S_j = \{S_{j,1}, S_{j,2}, \ldots, S_{j,q}\}$ of S where $S_{j,i} \leftarrow S \cap \{k : c_k[j] = i\}$ holds for $i = 1, \ldots, q$. A tracing ciphertext of type $(j, v) \in \{1, \ldots, n\} \times$

$\{0, 1, \ldots, q\}$ is the transmission $c = (c_1||c_2||\ldots||c_q)$ where for $i \in [q]$:

$$c_i = hdr_i||e_i \qquad \begin{array}{l} (hdr_i, K_i) \leftarrow \texttt{BEnc}(PK_{\texttt{B}}, S_i) \\ e_i \leftarrow \begin{cases} SEnc_{K_i}(R), & i \leq v \\ SEnc_{K_i}(K), & i > v \end{cases} \end{array}$$

for randomly chosen R and K. We say the decoder $\mathcal{D}_S^1$ succeeds in decrypting a tracing ciphertext of type (j, v) if it returns K and denote its approximated success probability by $p_{j,v}$. The algorithm locates the smallest s value for which $|p_{j,s} - p_{j,s-1}|$ is non-negligible and set $w_j \leftarrow s$.

After repeating the sub-procedure above, for each $j = 1, \ldots, \ell$, we produce a pirate codeword w. The tracing algorithm, finally, outputs $t \leftarrow \mathcal{F}.\texttt{Identify}(w)$ that is accused of being a traitor index.

Remark (1): We guarantee the existence of a smallest index value s due to the triangular inequality where we have $p_{j,0} = 1$ (we consider a perfect decoder) and $p_{j,q} \approx 0$ (a (j, q)-type tracing ciphertext randomizes all ciphertexts). A lower bound for a non-negligible probability difference will be provided later in tracing analysis of the transformation.

Remark (2): Our transformation does not hide the partition of a transmission, i.e. the adversary can be modeled to be aware of the way partition is computed. Hence, the fingerprinting code that we employ should be an open fingerprinting code.

3.1 Traceability of the Transformation

We, next, formally prove the traceability of our transformation.

Theorem 1 *[Traceability of a Perfect Decoder] Consider the generic T&R scheme* `T` *that is constructed as above by employing a BE scheme* `B`*, a symmetric encryption scheme* `SYM` *and an open fingerprinting code* `F`*.*

Let `B` *be KEM-IND-CCA secure with probability* ϵ_b*, and* `SYM` *be IND-CCA secure with probability* ϵ_s *and* `F` *be an* (ϵ_f, t)*-identifier* q*-ary fingerprinting code that is publicly samplable by sampler* `Z` *with failure probability* ϵ_z *in the sense of Definition 3.*

`T` *is a trace and revoke scheme with success probability* $1 - \epsilon_f - \ell\epsilon$ *against* t*-coalition* 1*-pirate's if it holds that*

$$4q(\epsilon_s + \epsilon_b) + 2\epsilon_z + \frac{1}{|M|} \leq 1$$

where M *is the message space.*

Proof We argue that no polynomial time attacker A that forges a perfect decoder can win the tracing game (Game 2) with some non-negligible probability. More

specifically, we consider a resettable and available pirate decoder $\mathcal{D}_S^1$ constructed for a subset $S \in [n]$ by a coalition of at most t traitors. The tracing process can be considered as three stages: (1) Approximating the success probability of the decoder for each tracing ciphertext of type $(j, v) \in [\ell] \times [q]$, (2) Producing the pirate codeword w and finally (3) Identifying a traitor index.

(1) *Approximation:* We define $\mu_{j,v}$ as the expected number of times the decoder succeeds in experiments of type (j, v) and $\rho_{j,v}$ as the actual number of successes during the approximation process where each experiment is repeated λ times. We would like to bound the approximation difference $|\rho_{j,b} - \mu_{j,b}|$. Choosing $\lambda = \frac{3\ln(8/\epsilon)}{\Delta^2}$, we claim that $|\rho_{j,b} - \mu_{j,b}| \geq \lambda \cdot \Delta$ with probability at most $\epsilon/4$.

Due to the allowed resettability of the decoder after each tracing query we can consider the experiments performed by the tracer are independent. By applying a two-tailed form of the Chernoff bound we will have:

$$Pr[|\rho_{j,b} - \mu_{j,b}| \geq \alpha] \leq 2e^{-\frac{\alpha^2}{3\mu_{j,b}}} \leq 2e^{-\frac{\alpha^2}{3\lambda}}$$

Substituting $\alpha = \lambda \cdot \Delta$ and $\lambda = \frac{3\ln(8/\epsilon)}{\Delta^2}$ we obtain:

$$\begin{array}{lll} 2e^{-\alpha^2/3\lambda} & = 2e^{-\frac{\lambda^2\Delta^2}{3\lambda}} & \leq 2e^{-\Delta^2\lambda/3} \\ & \leq 2e^{-\ln(8/\epsilon)} & \leq \epsilon/4 \end{array}$$

The above analysis conclude the fact that for any $j \in \{1, \ldots, \ell\}$ and $v \in \{1, \ldots, q\}$, we have $|\rho_{j,v} - \mu_{j,v}| \leq \lambda \cdot \Delta$ with probability at least $1 - \epsilon/4$. This fact is equivalent of saying $|p_{j,v} - \sigma_{j,v}| \leq \Delta$ with probability at least $1 - \epsilon/4$ for which $\mu_{j,v} = \lambda \cdot \sigma_{j,v}$ and $\rho_{j,v} = \lambda \cdot p_{j,v}$ holds.

(2) *Pirate Codeword Generation:* The tracer sets $w_j = s$ for the smallest $s \in [q]$ that satisfies $|\rho_{j,s-1} - \rho_{j,s}| \geq \lambda\theta$ and we choose θ to be equal to $\frac{1-2\epsilon_z-1/|M|}{q}$.

We next argue that the pirate codeword w is in the descendant set of the traitor coalition T. This is equivalent of claiming that if $w_j = s$ then $S_{j,s} \cap T \neq \emptyset$ holds for $j = 1, \ldots, \ell$.

By applying the triangular inequality for the equations $|\mu_{j,s-1} - \rho_{j,s-1}| \leq \lambda \cdot \Delta$ and $|\mu_{j,s} - \rho_{j,s}| \leq \lambda \cdot \Delta$, we obtain:

$$|\mu_{j,s-1} - \mu_{j,s}| \geq |\rho_{j,s-1} - \rho_{j,s}| - 2\lambda\Delta$$

with probability at least $(1 - \epsilon/4)^2 \geq 1 - \epsilon/2$.

It follows that if the tracer returns the value s, i.e., $|\rho_{j,s-1} - \rho_{j,s}| \geq \frac{\lambda(1-2\epsilon_z-1/|M|)}{q}$ for the choice of $\Delta = \frac{1-2\epsilon_z-1/|M|}{4q}$, it will happen with probability at least $1 - \epsilon/2$ that

$$\begin{array}{ll} |\mu_{j,s-1} - \mu_{j,s}| & \geq \frac{\lambda(1-2\epsilon_z-1/|M|)}{q} - 2\frac{\lambda(1-2\epsilon_z-1/|M|)}{4q} \\ |p_{j,s-1} - p_{j,s}| & \geq \frac{1-2\epsilon_z-1/|M|}{2q} \end{array}$$

The above suggest that if a value $s \in [q]$ is returned by the tracer, it holds that the probability difference $|p_{j,s-1} - p_{j,s}|$ exceeds the threshold of $2(\epsilon_s + \epsilon_b)$ with probability at least $1 - \epsilon/2$, as we know from the statement of the theorem that $4q(\epsilon_s + \epsilon_b) + 2\epsilon_z + \frac{1}{|M|} \leq 1$. In such case, we claim that $S_{j,s} \cap T \neq \emptyset$. We proceed with proof by contradiction, i.e. assume the converse of the statement $|p_{j,s-1} - p_{j,s}| > 2(\epsilon_s + \epsilon_b)$ and there exists no traitor in set $S_{j,s}$. This contradicts with the security claims of the underlying symmetric encryption scheme `SYM` and broadcast encryption scheme `B`. Indeed, if there is no traitor in set $S_{j,s}$, the pirate decoder can distinguish between the tracing ciphertext of type $(j, s-1)$ and of type (j, s) by only breaking the underlying encryption schemes. Hence, the distinguishing probability is bounded by $2(\epsilon_s + \epsilon_b)$.

On the other hand, we claim that $p_{j,0} \geq 1 - 2\epsilon_z$: this is because a tracing ciphertexts of type $(j, 0)$ for $j = 1, \ldots, \ell$ is different from a regular transmission in the way partition of the subset S is chosen. Recall that `F` is an (ϵ_f, t)-identifier fingerprinting code that is publicly samplable by sampler `Z` with failure probability ϵ_z in the sense of Definition 3. Hence, the pirate decoder $\mathcal{D}_S^1$ would decrypt the tracing ciphertexts of type $(j, 0)$, for all $j \in [\ell]$, with probability at least $1 - 2\epsilon_z$. Otherwise, the decoder can be used to distinguish the way sampler and the fingerprinting code works.

We also know that $p_{j,q} \leq \frac{1}{|M|}$ since a tracing ciphertext of type (j, q) totally hides the information on the message transmitted. Hence the triangular inequality implies that there exists at least one $0 < v \leq n$ such that $|p_{v-1} - p_v| \geq (1 - 2\epsilon_z - 1/|M|)/q$. With an identical argumentation as above we show that when the tracer reaches the v-th interval it will output v with probability $1 - \epsilon/2$. This suggests that the tracer will indeed output a user and not reach the end of all experiments without discovering any candidate corrupted user. Combining the above two results we conclude the pirate codeword generation phase with success probability $1 - \epsilon$.

(3) *Traitor Identification* We argue above that the pirate codeword is in the descendant set of the traitor coalition. In our application of fingerprinting code, the partition in a tracing transmission does not hide the user codewords, i.e. the code is open to the adversary. Hence, `Identify(`w`)` returns a traitor index with probability at least $1 - \epsilon_f$ as long as the fingerprinting code is open (not secret as in the cases of Tardos or Boneh-Shaw codes). This completes the proof of the traceability. The overall failure probability of accusing an innocent user is bounded by $\epsilon_f + \ell\epsilon$ (for the failures in identification, and in approximations, respectively) for the given parameters. ■

3.2 Samplable Fingerprinting Codes

As we argued above, the traceability of our generic construction relies on the existence of publicly samplable open fingerprinting code. Fortunately, the Chor-Fiat-Naor fingerprinting code [10] is such a code.

Theorem 2 *There exists a sampling algorithm* $\mathtt{Z}_{CFN}$ *with auxiliary information* w *(the size of the traitor coalition) such that Chor-Fiat-Naor fingerprinting code resistant against a traitor coalition of size* w *is publicly samplable by* $\mathtt{Z}_{CFN}$ *in the sense of Definition 3. The sampler* $\mathtt{Z}_T$ *requires computation time linear in number of codewords.*

Proof Due to lack of space, we omit the description of Chor-Fiat-Naor code here. Very briefly, it generates a code $\mathcal{C} = \{c_1, \ldots, c_n\} \subset Q^\ell$ randomly over an alphabet Q of size $q = 2w^2$: more specifically, for all choices of $i \in [n]$ and $j \in [\ell]$, we set $c_i[j] = k$ with probability $1/q$ for any $k \in Q$. If the length of the code is $4w^2 \log n/\epsilon$, then the code becomes a w-traceability code with probability $1 - \epsilon$, hence becomes resistant against a traitor coalition of size w.

$\mathtt{Z}(1^n, w)$ will follow the same randomized method to construct a partition $V = \{V_1, \ldots, V_q\}$: for each $i \in [n]$, randomly selects an element from the alphabet Q, say $k \in Q$ and places i in set V_k. The proof of the theorem is now straight-forward as the columns of the generated code are independently sampled and the sampler $\mathtt{Z}$ constructs the partition in exact same way. Note also, the computation time of the sampler is linear in number of codewords n. ■

3.3 Our Instantiations

We instantiate our generic construction with the open fingerprinting code of Chor-Fiar-Naor in [10] and the following three broadcast encryption schemes. The efficiency characteristics of the below instantiations are compared to the existing trace and revoke schemes in the introduction(see Table 1)

`BGW:` One seminal work on public key BE , the scheme of Boneh, Gentry, and Waters [4], proposes a basic scheme, denoted by `BGW1`, and a general scheme `BGW2` that employs several instances of the basic scheme in parallel. Our generic construction instantiated with the schemes of [4] will result in static security with the same performance characteristics.

`Del:` The scheme of Delerablée [13] and the virtually identical scheme of Sakai and Furukawa [37] are examples of ID based broadcast encryption schemes. The scheme of [13] puts a bound m for the number of receivers per transmission, and the public key size is linear in m instead of the number of receivers n. The instantiations we provide in Table 1, are for $m = n$ denoted by scheme `Del1` and for $m = \sqrt{n}$ denoted by scheme `Del2`.

`GW:` In [20] three different schemes with different properties are given. One is a standard BE scheme (not ID-based) which we will call `GW1`. In this scheme, public key is of size $O(m)$ where m is the maximum number of receivers in a broadcast. Ciphertext and private keys are of constant size. This scheme satisfies semi-static security, where the attacker commits to a subset of users before seeing public keys first, and afterwards can choose any subset of it as the final set to be attacked. Second and third schemes we will consider from [20] are identity based BE schemes.

GW2 is an adaptively secure identity based BE scheme in the random oracle model. Achieving adaptive security in the standard model costs a trade off between the public key size and the ciphertext size as in the case of the scheme GW3.

Some Remarks: The round complexity of a black-box tracing mechanism is the number of queries asked to the decoder and it is an important efficiency parameter as formalized in [28]. Most of the schemes in the literature (e.g. [8, 9]) leads to a quadratic number of tracing queries in number of users regardless of the traitor coalition. On the other hand, our generic construction, when employs a fingerprinting code with a bound w on traitor coalition, would result in round complexity of $O(w^4)$. Here in this paper, we applied the linear tracing strategy to locate a subset containing a traitor. As an alternative suggested by [28], we may have used the tracing strategy of [28] which would reduce the round complexity to $O(w^2)$.

A further improvement over the scheme is possible if the fingerprinting code is generated at the time of tracing for $|S|$ many codewords instead of as many as the number of receivers n. This is a substantial improvement over the encryption time as it is sufficient to flip $|S|$ coins. This improvement should be considered for applications where the enabled set of receivers is substantially less than the whole population.

3.4 Broadcast Confidentiality

We next prove the security of our construction regarding the confidentiality of the broadcast messages.

Theorem 3 (Confidentiality) *Let T be a trace and revoke scheme that is constructed through our generic transformation from a BE scheme B, a symmetric encryption scheme S and a q-ary fingerprinting code. T would satisfy the KEM-IND-CCA security for any polynomial time attacker A_T such that*

$$Adv_{A_T} \leq 2q \cdot \epsilon_b + 2q \cdot \epsilon_s$$

holds where B is ϵ_b-secure in the KEM-IND-CCA model and S is ϵ_s-secure in the IND-CPA model. It further holds that if the underlying scheme B supports adaptive security then so does the scheme T.

Proof We will use a game hopping approach to prove the theorem: we will start with the basic confidentiality game for trace and revoke scheme. We next modify the basic game gradually to reach a final game which provides the adversary no advantage. This is a standard proof technique that bounds the advantage of the adversary in the original game by the differences in the subsequent games.

Let G_0 be the KEM-IND-CCA message confidentiality game for trace and revoke schemes. We had passed over the full description of the game in Section 2.2 with a quick reference to the Game 1. The details of this game is depicted in Figure 1. We denote an arbitrary adversary playing the game G_0 against the challenger C_T

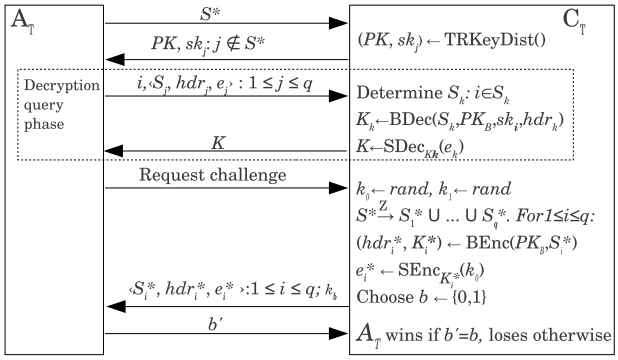

Figure 1: Game G_0: the actual KEM-IND-CCA game.

of the trace and revoke scheme by A_T. In the figure, we considered a static attack model where the adversary commits to the set S^* it wants to attack. The challenger publishes the public key afterwards. In contrast, it is also possible that the adversary commits to the set S^* after it observes the public key. The latter, denoted by adaptive attack, is a stronger attack model as the public key may let the adversary have some non-trivial information that is useful for the choice of the target set. The order of the commitment of the target set and the publication of the public key will not affect the validity of our proof arguments below. The choice of the order is propagated in our transformation smoothly. Hence we will consider the security for static attack model, the adaptive case follows in a similar way.

We proceed with description of the subsequent games. Let W_j denote the event that the adversary A_T wins the j-th game G_j:

Game 0: The first game, depicted in Figure 1 is identical to the KEM security game for trace and revoke schemes. Thus,

$$|Pr[W_0] - \frac{1}{2}| = Adv_{A_T}$$

In this game, the challenger prepares a valid ciphertext. The partition $\{S_i^*\}_{i\in[q]}$ is chosen based on the sampler $\mathtt{Z}$ and constructs the headers

$$(hdr_i^*, K_i^*) \leftarrow \mathtt{BEnc}(PK_\mathtt{B}, S_i^*), \quad e_i^* \leftarrow \mathtt{SEnc}_{K_i^*}(k_0)$$

for all $1 \le i \le q$ where k_0 and k_1 are randomly chosen keys compatible with the symmetric encryption algorithm $\mathtt{SEnc}$. Along with the full headers $\langle S_i^*, hdr_i^*, e_i^*\rangle_{i\in[q]}$, the challenger transmits k_b for a randomly selected bit b. We say the adversary wins the game if it guesses b correctly.

Game 1 through Game q: This sequence of games is identical to the first Game G_0 except the way the challenger prepares the encryption for $e_\cdot^*$. In Game G_j, the encryption e_i^* for $i \le j$ is made under a randomly chosen key K_i^+ instead

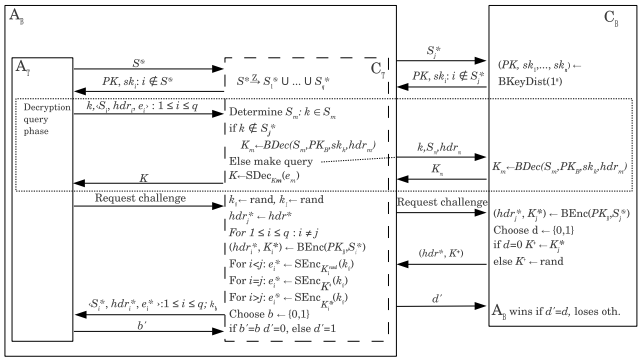

Figure 2: Constructing a broadcast encryption adversary A_B that simulates the challenger of the trace and revoke adversary A_T. Its advantage is reduced to the A_T's ability of distinguishing its views among games G_{j-1} and G_j.

of K_i^*:

$$e_i^* \leftarrow \mathtt{SEnc}_{K_i^+}(k_0)$$

Such modification breaks the relation between the header hdr_i^* and e_i^*. We next claim that there exists a broadcast encryption adversary A_B whose running time is about the same as A_T such that:

$$|Pr[W_{j-1}] - Pr[W_j]| = 2Adv_{A_B}$$

holds for $j = 1, \ldots, q$. We next argue the construction of the adversary A_B, depicted in Figure 2 which intends to break the KEM-IND-CCA security of the broadcast encryption $\mathtt{B}$. The adversary A_B will simulate the challenger C_T of the trace and revoke security game. The simulator will embed the challenge it receives from the broadcast challenger C_B to the challenge requested by the adversary A_T. After receiving the set S^*, the simulator will create the partition $S^* = \{S_1^*, \ldots, S_q^*\}$ immediately and forwards the j-th subset to the broadcast encryption challenger C_B. This is a crucial step to be able to simulate the secret keys of the scheme $\mathtt{T}$ whose keys are basically the keys of the underlying broadcast encryption scheme. The simulator will be able to respond the decryption queries $k, \langle S_i^*, hdr_i^*, e_i^* \rangle_{i \in [q]}$ of the adversary A_T as long as the secret key of the intended user k is available to the adversary A_B. Otherwise, the adversary forwards the decryption query $k, \langle S_j^*, hdr_j^* \rangle$ to the challenger C_B and retrieves the key to decrypt the symmetric encryption e_i^*.

After requesting the challenge from A_T, the adversary A_B simulates the challenger C_T as follows: All the headers $\langle S_i^*, hdr_i^*, e_i^* \rangle$ for $i \neq j$ will be prepared as in the Game G_{j-1}. The challenge received from C_B will base the $j-th$ header of the trace and revoke challenge. Upon receiving (hdr^*, K^+) from the challenger C_B we set $hdr_j^* = hdr^*$ and $e_j^* = \mathtt{SEnc}_{K^+}(k_0)$

Observe, now, that if the challenge of C_B is a valid broadcast ciphertext (this corresponds to the case $d = 0$ in Figure 2), the adversary A_T plays in Game G_{j-1}.

In contrast, A_T plays in Game G_j if the challenge is not valid ($d = 1$ in Figure 2).

Let us compute the winning probability of A_B: (i) if $d = 0$ A_B wins the game if A_T wins the game, hence bounded by $Pr[W_{j-1}]$; (ii) if $d = 1$ A_B wins the game if A_T loses the game, hence bounded by $1 - Pr[W_j]$. This completes our claim that $|Pr[W_{j-1}] - Pr[W_j]| = 2Adv_{A_B}$ which is then upper-bounded by $2\epsilon_b$

At this point, we have reached to game G_q where all BE keys K_i^* are distorted. We continue with q more games gradually replacing the key k_0 with k_i^+'s.

Game q+1 through Game 2q: This sequence of games is identical to the Game G_q except the way the challenger prepares the encryption for $e_\cdot^*$. In Game G_{q+j}, we set:

$$e_i^* \leftarrow \mathtt{SEnc}_{K_i^+}(k_i^+)$$

for $i \leq j$ where k_i^+'s are randomly chosen. Such modification in Game G_{q+j} hides totally the information of k_b in the first j headers. We next claim that there exists a symmetric encryption adversary A_S whose running time is about the same as A_T such that:

$$|Pr[W_{q+j-1}] - Pr[W_{q+j}]| = 2Adv_{A_S}$$

holds for $j = 1, \ldots, q$. We construct the adversary A_S in a similar way we have constructed the adversary A_B. We omit the details of the simulation due to simplicity and similarity. Hence the probability differences above are upper-bounded by $2\epsilon_s$.

Note that the last game G_{2q} gives absolutely no information about k_b thus the probability $Pr[W_{2q}]$ of the adversary winning the game G_{2q} is $\frac{1}{2}$. Applying the triangular inequalities over the probability differences above we obtain:

$$\begin{aligned} 2q \cdot \epsilon_b + 2q \cdot \epsilon_s &\geq \textstyle\sum_{i=1}^{2q} |Pr[W_i] - Pr[W_{i-1}]| \\ &\geq |Pr[W_{2q}] - Pr[W_0]| \\ &\geq |\tfrac{1}{2} - Pr[W_0]| \\ &\geq Adv_{A_T} \end{aligned}$$

which completes the security proof of our generic transformation. ■

4 Stronger Traceability Modes

4.1 Tracing Imperfect Decoders

We proved the traceability of our generic construction against a perfect decoder in Section 3. However, as discussed in [6], any scheme whose traceability is due to a fingerprinting code can fail to identify a traitor key if the decoder chooses not to decrypt some transmissions. Such decoder is called an imperfect decoder and its behavior may lead to some gaps (leaving some bits of the pirate codeword unspecified, denoted by '?'.) which will result to a failure in identification algorithm. In the case of our generic construction, the pirate decoder may refuse to decrypt, even regular transmissions, for particular choices of the partition. The solution

against such behavior is to use a δ-robust fingerprinting code. δ-robust fingerprinting codes would still lead identification of a traitor even if the pirate codeword has up to $\delta \cdot \ell$ many '?' marks. An analysis of such a transformation is provided in [6]. We can apply the same transformation in a similar way to our generic construction to obtain traceability against imperfect decoders.

Due to lack of space, we do not want to detail this transformation as it is supplementary to our main result in this work. We briefly discuss some critique issues related to the transformation:

(i) We should be able to find an open publicly samplable δ-robust fingerprinting codes. Fortunately, extending the length of an open Chor-Fiat-Naor code by a factor of $\frac{1}{1-\delta}$ would be suffice to obtain such code to be employed in the generic construction.

(ii) A special care is needed to find the relation between δ and σ: an imperfect σ-decoder may have an arbitrary decryption probability distribution over the choice of the partition. Regardless of this fact, there is a lower bound on the success probability of the decoder, denote it by γ, to have a non-? mark: based on our traceability proof given in Theorem 1, $\gamma = 4q(\epsilon_s + \epsilon_b) + 2\epsilon_z + \frac{1}{|M|}$. Let us call a partition 'bad partition' if the pirate decoder, on a ciphertext prepared for this partition, has a success probability less than γ. If δ is the fraction of bad partitions, then the decoder's error rate (that is $1 - \sigma$) is at least $\delta(1 - \gamma)$. Solving for δ we obtain $\delta < \frac{1-\sigma}{1-\gamma}$.

(iii) The above calculation is made over the choices of any partition possible through the sampling algorithm. However, in actual tracing we concentrate on the partitions based on the fingerprinting codes. Hence, the notion of public samplability should be revisited such that the density of bad partitions in the output of sampling algorithm should preserve the same density in the output of fingerprinting code. Fortunately, the open Chor-Fiat-Naor code satisfies this property as the code is generated in exact same way its sampler works.

4.2 Public Traceability

In Eurocrypt 2005, Chabanne, Phan and Pointcheval [11] introduced the notion of public traceability where tracing is a procedure that requires no secrets. A two user solution was presented in [11] and further improved to the multiuser setting with short transmissions in [17] and [34]. In above schemes, the public key size and the private key sizes are all linear in length of the fingerprinting code employed for key distribution. The trace and revoke scheme of [9] is also publicly traceable with shorter key sizes, i.e. $O(\sqrt{n})$ many, but requires higher bandwidth, i.e. it has a ciphertext length of $O(\sqrt{n})$.

Our proposed generic construction supports the public traceability as there is no tracing key. The fingerprinting code is used to variate the way receivers decrypt logically without affecting the key-distribution. The encryption is done through a sampler that is of public knowledge, and any third-party can trace by generating

a code. The code may have secrets available to the tracing party but this does not affect any other party to run her tracing capability. Hence, we provide the first publicly traceable schemes that have constant private key sizes with reasonable public key size and ciphertext length.

4.3 Tracing and Revoking Pirate Rebroadcasts

It is possible to obtain a scheme for tracing and revoking pirate rebroadcasting based on our generic construction. In such adversarial setting, an adversary, corrupting a number of traitors, decrypts the message through the key material available to him and rebroadcasts the clear message. Note that the rebroadcast does not reveal any information about the traitor-keys unless the clear message itself is bound to a specific user key. In this direction, we transmit different versions of the content so that each version is decrytable by different set of keys. This is achieved in the literature by watermarking the content. In such setting, traitor-identification is achieved through observing the pattern of watermarks available to the pirate.

Let us provide a simple description on how to make our generic transformation work in the pirate rebroadcasting setting. We first generate the watermarked versions of the content m denoted by $m_1, m_2, \ldots, m_q$. For simplicity, we prefer an encryption in the standard model, a KEM version is possible by replacing m_i with a further level of symmetric encryption key. Similar to the original construction, we have a partition $\{V_1, V_2, \ldots, V_q\}$ of $[n]$ (the choice of the partition is through a sampler in regular transmission and through fingerprinting code in tracing transmissions). Setting $S_i = V_i \cap S$ for each $i = 1, \ldots, q$, we broadcast the message $c = (c_1||c_2||\ldots||c_q)$ where, for each $i = 1, \ldots, q$, we construct $c_i = hdr_i||e_i$ and

$$(hdr_i, K_i) \leftarrow \mathtt{BEnc}(PK_{\mathtt{B}}, S_i), \quad e_i \leftarrow SEnc_{K_i}(m_i)$$

The traceability of the scheme above can be proven in almost exact way as we did for the original transformation in Section 3. We will not require linear tracing strategy as watermarking already differentiates the way we encrypt for different subsets in the partition.

Our generic scheme, instantiated with any of the schemes [4, 13, 20], will lead the first tracing and revoking pirate rebroadcasts in the public key setting with constant private key size and short transmission lengths.

References

[1] AACS - Advanced Access Content System, http://www.aacsla.com, 2007.

[2] M. Abdalla, A. Dent, J. Malone Lee, G. Neven, D. Phan, and N. Smart. Identity-Based Traitor Tracing. Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 4450, pages 361–376, 2007.

[3] D. Boneh, M. K. Franklin: An Efficient Public Key Traitor Tracing Scheme. CRYPTO 1999, pages. 338-353

[4] D. Boneh, C. Gentry, and B. Waters. Collusion resistent broadcast encryption with shorter ciphertexts and private keys. In *CRYPTO'05*, volume 3621 of *LNCS*, pages 258–275. Springer-Verlag, 2005.

[5] D. Boneh, A. Kiayias, H. W. Montgomery: Robust fingerprinting codes: a near optimal construction. Digital Rights Management Workshop 2010: 3-12

[6] D. Boneh and M. Naor. Traitor tracing with constant size ciphertext. In proceedings of the 15th ACM conference on Computer and Communications Security (CCS '08), pp. 501–510, 2008.

[7] D. Boneh and J. Shaw, Collusion-Secure Fingerprinting for Digital Data, IEEE Transactions on Information Theory, Vol. 44(5) pp. 1897-1905, 1998.

[8] D. Boneh, A. Sahai and B. Waters, Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. EUROCRYPT 2006, LNCS 4004, pp. 573-592.

[9] D. Boneh and B. Waters. A fully collusion resistent broadcast, trace, and revoke system. In *CCS '06*, pages 211–220. ACM, 2006.

[10] B. Chor, A. Fiat, and M. Naor, Tracing Traitors, CRYPTO '94, LNCS 839 Springer 1994, pp. 257-270.

[11] Hervé Chabanne, Duong Hieu Phan, David Pointcheval: Public Traceability in Traitor Tracing Schemes. EUROCRYPT 2005: 542-558

[12] R. Cramer, V. Shoup: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. EUROCRYPT 2002: 45-64

[13] C. Delerableé. Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In *ASIACRYPT'07*, volume 4833 of *LNCS*, pages 200–215. Springer-Verlag, 2008.

[14] A. W. Dent: Adapting the Weaknesses of the Random Oracle Model to the Generic Group Model. ASIACRYPT 2002: 100-109

[15] Y. Dodis and N. Fazio Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack . Public Key Cryptography PKC 2003. Lecture Notes in Computer Science, 2002, Volume 2567/2002, pp. 100–115.

[16] J. Furukawa and N. Attrapadung. Fully Collusion Resistant Black-Box Traitor Revocable Broadcast Encryption with Short Private Keys. In *Automata, Languages and Programming*, volume 4596 of *LNCS*, pages 496–508. Springer-Verlag, 2007.

[17] Nelly Fazio, Antonio Nicolosi, Duong Hieu Phan: Traitor Tracing with Optimal Transmission Rate. ISC 2007: 71-88

[18] A. Fiat and M. Naor. Broadcast encryption. In *CRYPTO'93*, volume 773 of *LNCS*, pages 480–491. Springer-Verlag, 1993.

[19] A. Fiat, T. Tassa: Dynamic Traitor Tracing. J. Cryptology 14(3): 211-223 (2001)

[20] C. Gentry, B. Waters. Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In EUROCRYPT '09, Lecture Notes in Computer Science, 2009, Volume 5479/2009, 171-188.

[21] M. T. Goodrich, J. Z. Sun, and R. Tamassia. Efficient tree based revocation in groups of low-state devices. In *CRYPTO'04*, volume 3152 of *LNCS*, pages 511–527. Springer-Verlag, 2004.

[22] D. Halevy and A. Shamir. The LSD broadcast encryption scheme. In *CRYPTO'02*, volume 2442 of *LNCS*, pages 47–60, London, UK, 2002. Springer-Verlag.

[23] H. Jin, J. Lotspiech: Renewable Traitor Tracing: A Trace-Revoke-Trace System For Anonymous Attack. ESORICS 2007: 563-577

[24] A. Kiayias and M. Yung, On Crafty Pirates and Foxy Tracers, ACM CCS-8 Workshop DRM 2001, LNCS 2320 Springer 2002, pp. 22-39.

[25] A. Kiayias, S. Pehlivanoglu, Pirate Evolution: How to Make the Most of Your Traitor Keys, CRYPTO 2007, LNCS 4622 Springer 2007 pp. 448-465

[26] A. Kiayias, S. Pehlivanoglu, Tracing and Revoking Pirate Rebroadcasts, ACNS 2009, LNCS 5536 Springer 2009 pp. 253-271

[27] A. Kiayias, S. Pehlivanoglu: On the security of a public-key traitor tracing scheme with sublinear ciphertext size. Digital Rights Management Workshop 2009: 1-10

[28] A. Kiayias, S. Pehlivanoglu. Improving the Round Complexity of Traitor Tracing Schemes. In *Applied Cryptography and Network Security*, vol. 6123, pages 273–290, 2010.

[29] K. Kurosawa, Y. Desmedt: Optimum Traitor Tracing and Asymmetric Schemes. EUROCRYPT 1998: 145-157

[30] M. Lee, D. Ma, M. Seo: Breaking Two k-Resilient Traitor Tracing Schemes with Sublinear Ciphertext Size. ACNS 2009: 238-252

[31] T. Matsushita, H.Imai, A Public-Key Black-Box Traitor Tracing Scheme with Sublinear Ciphertext Size Against Self-Defensive Pirates. AsiaCrypt04, Lecture Notes in Computer Science 3329.

[32] D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In *CRYPTO'01*, volume 2139 of *LNCS*, pages 41–62. Springer-Verlag, 2001.

[33] M. Naor and B. Pinkas, Efficient Trace and Revoke Schemes, FC 2000 LNCS 1962 Springer 2001, pp. 1–20.

[34] D. H. Phan, R. Safavi-Naini, D. Tonien: Generic Construction of Hybrid Public Key Traitor Tracing with Full-Public-Traceability. ICALP (2) 2006: 264-275

[35] D. H. Phan, V. C. Trinh: Identity-Based Trace and Revoke Schemes. ProvSec 2011: 204-221

[36] R. Safavi-Naini, Y. Wang: Sequential traitor tracing. IEEE Transactions on Information Theory 49(5): 1319-1326 (2003)

[37] R. Sakai and J. Furukawa. Identity-Based Broadcast Encryption. In *Cryptology ePrint Archive*, Report 2007/21.

[38] G. Tardos. Optimal probabilistic fingerprint codes. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, STOC '03, pages 116–125, 2003.