This paper appears in the proceedings of the
First International Conference on Cryptography in Vietnam 2006
(25 – 28 September 2006, Hanoi, Vietnam)
Phong Q. Nguyen Ed. Springer-Verlag, LNCS 4341, pages 354–365.

Traitor Tracing for Stateful Pirate Decoders with Constant Ciphertext Rate

Duong Hieu Phan

University College London
h.phan@adastral.ucl.ac.uk

Abstract. Stateful pirate decoders are history recording and abrupt pirate decoders. These decoders can keep states between decryptions to detect whether they are being traced and are then able to take some counter-actions against the tracing process, such as "shutting down" or erasing all internal information. We propose the first constant ciphertext rate scheme which copes with such pirate decoders. Our scheme moreover supports black-box public traceability.

1 Introduction

In the secure distribution of digital content, there are two main types of schemes: *broadcast encryption* schemes, which enable a center to prevent a set of users from recovering the broadcasted information; and *traitor tracing* schemes, which enable the center to trace users who collude to produce pirate decoders. This paper focuses on the traceability property for pirate decoders in the strongest model (according to the hierarchy established by Kiayias and Yung [7]). In [7], the authors described various categories of pirate decoders which are resumed below:

Stateless pirate decoders. These decoders are *resettable* and *available*. A resettable decoder can be reset to its initial state by the tracer at any time. This gives the tracer the advantage of making independent tests during the tracing process. An available pirate decoder is a device that does not take any counter-action against the tracing process and thus is always available for the tracer.

Stateful pirate decoders. In contrast to stateless pirate decoders, these are *history recording* and *abrupt* pirate decoders. A history recording pirate decoder can remember previous queries made by the tracer in order to detect if it is being traced. Abrupt pirate decoders can take some counter-actions against the tracing process such as the "shutting down" mechanism, a process by which pirate decoders erase all internal key information and thus defeat the tracing process. The history recording capability along with abrupt capability can be used by pirate decoders to evade tracing.

Kiayias and Yung [7] also showed an interesting method to convert some types of tracing systems for stateless pirate decoders into tracing systems for stateful ones by embedding robust watermarks in the content. However, previous tracing systems for stateful decoders are inefficient in terms of ciphertext rate.

Schemes with constant transmission rate. These schemes are well-suited to encrypt large messages. An interesting property of these scheme is the efficient black-box traceability, *i.e.* the tracing procedure does not have to open the pirate decoder, but only interacts with it. However, the constant transmission rate is asymptotically achieved and for large plaintexts (due to the use of collision-secure codes and codes with identifiable parent property). We note that the constant ciphertext rate schemes [7, 5, 10] were only designed for stateless pirate decoders.

Public Traceability. Chabanne *et al.* [5] introduced the notion of public traceability where tracing is a black-box and publicly computable procedure which allows an untrusted party to trace pirate decoders. Phan *et al.* [10] introduced the first constant ciphertext rate traitor tracing scheme with public-traceability.

1.1 Contribution

We propose the first traitor tracing scheme for stateful decoders with constant ciphertext rate. Furthermore, our scheme still keeps the desirable properties of previous traitor tracing with constant ciphertext rate, namely black-box traceability and public traceability.

We first propose a basic scheme for stateful pirate decoders by employing watermarking technique [7] in the Phan, *et al.*'s basic scheme [10]. We then introduce an efficient generalization of the basic scheme to obtain an efficient general scheme. Although our basic scheme is significantly less efficient than Phan *et al.*'s basic scheme, the general scheme is almost as efficient as theirs. We moreover point out that the latter cannot deal with stateful pirate decoders.

2 Preliminaries

In this section, we first recall definitions of Public Key Encryption (PKE) and of Data Encapsulation Mechanism (DEM) which will be used in our constructions. We then review the definition of traitor tracing systems with public traceability.

2.1 Public-Key Encryption

A public-key encryption scheme PKE is defined by the three following algorithms:

- The *key generation algorithm* Gen. On input 1^λ, where λ is the security parameter, the algorithm Gen produces a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public and private keys.
- The *encryption algorithm* Enc. Given a message m (in the space of plaintexts $\mathcal{M}$) and a public key pk, $\mathsf{Enc}_{\mathsf{pk}}(m)$ produces a ciphertext c (in the space of ciphertexts $\mathcal{C}$) of m. This algorithm may be probabilistic (involving random coins $r \in \mathcal{R}$), it is then denoted $\mathsf{Enc}_{\mathsf{pk}}(m; r)$.
- The *decryption algorithm* Dec. Given a ciphertext $c \in \mathcal{C}$ and the secret key sk, $\mathsf{Dec}_{\mathsf{sk}}(c)$ gives back the plaintext $m \in \mathcal{M}$.

2.2 Data Encapsulation Mechanism (DEM)

A DEM is a symmetric encryption scheme that consists of the following algorithms:

- *Setup algorithm* $\mathsf{DEM.Setup}(1^\lambda) \rightarrow \mathcal{K}_D$: an algorithm that specifies the symmetric key space $\mathcal{K}_D$.
- *Encryption algorithm* $\mathsf{DEM.Enc}_{dk}(m) \rightarrow \tau$: a deterministic, polynomial-time algorithm that encrypts m into τ, using a symmetric-key $dk \in \mathcal{K}_D$.
- *Decryption algorithm* $\mathsf{DEM.Dec}_{dk}(\tau) \rightarrow m$: a deterministic, polynomial-time algorithm that decrypts τ to m, using a symmetric-key $dk \in \mathcal{K}_D$.

2.3 Traitor Tracing System with Public Traceability

Definition 1 (Pirate Decoder). A pirate decoder D is defined as a probabilistic circuit that takes as input a ciphertext C and outputs some message M or $\perp$.

Definition 2 (Traitor Tracing System with Public Traceability). A Traitor Tracing system with public traceability consists of the following four algorithms:

Setup(N, λ) akes as input N, the number of users in the system, and λ, the security parameter. The algorithm runs in polynomial time in λ and outputs a public key pk and private keys $K_1, ..., K_N$, where K_u is given to user u.

Encrypt(pk, M) encrypts M using the public broadcasting key pk and outputs ciphertext C.

Decrypt(j, K_j, C, pk) decrypts C using the private key K_j of user j. The algorithm outputs a message M or $\perp$.

Trace$(\mathcal{D}, \mathsf{pk})$ is an oracle algorithm that is only given as input the public key pk and a pirate decoder $\mathcal{D}$. The tracing algorithm queries the pirate decoder $\mathcal{D}$ as a black-box oracle, as defined above. It outputs at least a user in $1, 2, \ldots, N$

The system, called a $(N, t)-$TTS, must satisfy the following properties:

Correctness property: for all user $i \in \{1, \ldots, N\}$, and all messages M:

$$\mathsf{Decrypt}(j, K_j, \mathsf{Encrypt}(\mathsf{pk}, M), \mathsf{pk}) = M$$

Traceability property: from a pirate decoder $\mathcal{D}$ produced from a collusion of up to t users, the above tracing algorithm should be able to correctly return at least a user in the collusion producing $\mathcal{D}$.

3 PST Basic Scheme [10]

Below, we briefly review the basic scheme of Phan, Safavi-Naini, Tonien [10] (called PST scheme) which will take part in our construction of general scheme. Their basic scheme, called $\mathsf{PSTBasic}(N, \lambda)$, is described as follows.

Primitives: a public-key encryption PKE and a data encapsulation mechanism DEM.

Setup(N, λ)**:** *Algorithm* $\mathsf{PSTBasic.Gen}(1^\lambda, N) \rightarrow (\mathsf{pk}, sk_1, \ldots, sk_N)$.

The algorithm $\mathsf{PSTBasic.Gen}(.)$ takes as input the number of users N and a security parameter λ. It simply runs the setup algorithm of the public key encryption to define its public key pk and its private keys $sk_1, \ldots, sk_N$:

- For each $i = 1, \ldots, N$, call $\mathsf{PKE.Gen}(1^\lambda) \rightarrow (pk_i, sk_i)$.
- Set $\mathsf{pk} = (pk_1, \ldots, pk_N)$.

Encrypt(pk, m)**:** *Algorithm* $\mathsf{PSTBasic.Enc}(\mathsf{pk}, m) \rightarrow c$.

The algorithm $\mathsf{PSTBasic.Enc}(.)$ takes as input the public key pk, a message m and outputs a ciphertext c. It uses DEM to encrypt the message m and PKE to encrypt the key used in DEM:

- Choose a random dk
- Call $\mathsf{DEM.Enc}_{dk}(m) \rightarrow \tau$
- Compute $h = H(\tau)$ and for each $i = 1, \ldots, N$, call $\mathsf{PKE.Enc}_{pk_i}(dk||h) \rightarrow \sigma_i$.
- Define $c = (\sigma_1, \ldots, \sigma_N, \tau)$.

Decrypt$(i, sk_i, c, \mathsf{pk})$**:** *Algorithm* $\mathsf{PSTBasic.Dec}(sk_i, c) \rightarrow m$ or $\perp$.

The algorithm $\mathsf{PSTBasic.Dec}(.)$ takes as input a secret key sk_i and a ciphertext $c = (\sigma_1, \ldots, \sigma_N, \tau)$ and outputs a message m ot $\perp$:

- Call $\mathsf{PKE.Dec}_{sk_i}(\sigma_i) \rightarrow dk||h$
- If $h \neq H(\tau)$, return $\perp$
- Otherwise, call $\mathsf{DEM.Dec}_{dk}(\tau) \rightarrow m$ and output m

Trace$(\mathcal{D}, \mathsf{pk})$**:**

Algorithm $\mathsf{PSTBasic.Public\text{-}Trace}(\mathsf{pk}, \mathcal{D}) \rightarrow t$.

The algorithm $\mathsf{PSTBasic.Public\text{-}Trace}(.)$ takes as input the public key pk and a pirate decoder $\mathcal{D}$ and outputs a traitor t as follows:

- Choose randoms dk, m, then call $\mathsf{PSTBasic.Enc}_{pk}(m) \rightarrow (\sigma_1, \ldots, \sigma_N, \tau)$.
- For each $i = 1, \ldots, N$, choose random $d'_i \neq dk||h$ such that d'_i has the same length as $dk||h$.
- Call $\mathsf{PKE.Enc}_{pk_i}(d'_i) \rightarrow \sigma'_i$.
- Calculate the following probabilities:

- $p_0 = Pr[\mathcal{D}(\sigma_1, \sigma_2, \ldots, \sigma_N, \tau) = m]$
- $p_1 = Pr[\mathcal{D}(\sigma'_1, \sigma_2, \ldots, \sigma_N, \tau) = m]$
- ...
- $p_n = Pr[\mathcal{D}(\sigma'_1, \sigma'_2, \ldots, \sigma'_N, \tau) = m]$.

– If $|p_i - p_{i-1}|$ is not negligible, output $t = i$ as a traitor.

We first show that the above scheme can not be used for stateful pirate decoders.

Proposition 3. *A stateful pirate decoder can defeat the above tracing algorithm.*

Proof. Assume that user 1 and user N collude to produce a pirate decoder $\mathcal{D}$, whenever $\mathcal{D}$ receives a ciphertext $(\sigma'_1, \sigma_2, \ldots, \sigma_N, \tau)$, it can detect whether a tracing procedure has been applied. Therefore, by applying a standard delaying technique (such as the one used in [7]), the stateful pirate decoder can defeat the tracing algorithm: upon detecting tracing, the decoder might continue to work by returning the message m (by decrypting the ciphertext σ_N) for a random number of trials and then start returning a random message $m^\star$. By this strategy, any user $2, \ldots, N$ can be claimed to be guilty.

4 Basic Scheme against Stateful Pirate Decoders

We now transform the PSTBasic scheme for stateless decoders to a scheme for stateful decoders. In our construction, inspired by Kiayias and Yung's method, we employ a watermarking scheme. The basic scheme is therefore quite inefficient. However, in the next section, we will show how to use this basic scheme to construct an efficient general scheme.

Adequate presentation of a message. In almost all applications of traitor tracing, a slight modification of data does not affect the user. For example, users are not affected by a slight modification of a pixel in a figure or small changes in spaces between words in text display. For a data M, we call "adequate" presentations of M varied copies which can replace M without affecting the users. A decoder is usable if, from a ciphertext of M, it returns an adequate presentation of M.

Formally, as in [7], we restrict ourselves to plaintext spaces for which the following watermarking assumption is true:

Assumption 4 (Watermarking Assumption) *For some t, h, there is a probabilistic algorithm (t, h)-$\mathcal{W}$ such that, given any $M \in \mathcal{M}$, it produces h "versions" of M, $M_1, M_2, ..., M_h$, such that the following are true:*

(i) M_i are "adequate" presentations of M

(ii) there is an algorithm $\mathcal{W}'$ such that for any algorithm $\mathcal{A}$ that generates a M' given $M_{j_1}, \ldots, M_{j_k}$, $\mathcal{W}'$ given M' traces back to one of the M_{j_l} , provided that M' is an adequate presentation of M, and that k is below a certain threshold t.

This assumption has been used in [6, 7] and can be achieved in most audio or video streams. We can thus make use of watermarking techniques as those of [9, 2] to support the tracing process.

We remark that a $(t, h)-$watermarking scheme is also a $(t', h')-$ watermarking scheme, for all $t' \leq t, h' \leq h$. Therefore, possessing a $(t, h)-$watermarking scheme, we can use it as a $(N, N)-$watermarking scheme, for $N \leq t$ and $N \leq h$.

Basic scheme. Our basic scheme, called $\mathsf{StatefulBasic}(N, \lambda)$ is described as follows:

Primitives: a public-key encryption PKE, a data encapsulation mechanism DEM and a watermarking algorithm (N, N)-$\mathcal{W}$.

Setup(N, λ)**:** It simply runs $\mathsf{PSTBasic}(N, \lambda)$ with the same parameters, and outputs pk as the public encryption key and $sk_1, \ldots, sk_N$ as private keys.

Encrypt(pk, M)**:** *Algorithm* $\mathsf{StatefulBasic.Enc}(\mathsf{pk}, m) \rightarrow c$.

The algorithm $\mathsf{StatefulBasic.Enc}(.)$ takes as input the public key pk, a message m and outputs a ciphertext c. It uses $\mathcal{W}$ to produce "adequate" presentations of m, DEM to encrypt the messages and PKE to encrypt the keys used in DEM:

- Call $\mathcal{W}(m) \rightarrow m_1, \ldots, m_N$.
- Choose random keys $dk_1, \ldots, dk_N$
- For each $i = 1, \ldots, N$, call $\mathsf{DEM.Enc}_{dk_i}(m_i) \rightarrow \tau_i$
- Compute $h_i = H(\tau_i)$ and for each $i = 1, \ldots, N$, call $\mathsf{PKE.Enc}_{pk_i}(dk_i || h_i) \rightarrow \sigma_i$.
- Define $c = (\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$.

Decrypt(j, K_j, C, pk)**:** *Algorithm* $\mathsf{StatefulBasic.Dec}(sk_i, c) \rightarrow m_i$ or $\perp$.

The algorithm $\mathsf{StatefulBasic.Dec}(.)$ takes as input a secret key sk_i and a ciphertext $c = (\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$, it outputs a message m_i or $\perp$:

- Call $\mathsf{PKE.Dec}_{sk_i}(\sigma_i) \rightarrow dk_i || h_i$
- If $h_i \neq H(\tau_i)$, return $\perp$
- Otherwise, call $\mathsf{DEM.Dec}_{dk_i}(\tau_i) \rightarrow m_i$ and output m_i

Trace$(\mathcal{D}, \mathsf{pk})$**:**

Algorithm $\mathsf{StatefulBasic.Public\text{-}Trace}(\mathsf{pk}, \mathcal{D}) \rightarrow t$.

The algorithm $\mathsf{StatefulBasic.Public\text{-}Trace}(.)$ takes as input the public key pk and a pirate decoder $\mathcal{D}$ and outputs a traitor t as follows:

- Choose random dk, m, and call $\mathsf{PSTBasic.Enc}(\mathsf{pk}, m) \rightarrow (\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$. Recall that $\tau_i = \mathsf{DEM.Enc}_{dk_i}(m_i)$ and $m_1, \ldots, m_N$ are outputted by $\mathcal{W}(m)$.
- Give $(\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$ to $\mathcal{D}$.
- Suppose that $\mathcal{D}(\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N) \rightarrow m^\star$
- Call $\mathcal{W}'(m^\star) \rightarrow m_t$ and output t as a traitor.

Traceability. We first consider the traceability of the above system.

Theorem 5. *If W is a $(N, N)-$watermarking scheme,* PKE *and* DEM *are semantically secure, the above scheme is a fully collusion resistant traitor tracing scheme against stateful pirate decoders.*

Proof. We remark that the above tracing only makes one query to the pirate decoder. Moreover, this query is a valid ciphertext. Therefore, if a pirate decoder is usable, it should return an "adequate" presentation $m^\star$ of the original message m.

Let $(\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$ be the query to the pirate decoder and $m_1, \ldots, m_N$ be the N underlying messages of $\sigma_1, \ldots, \sigma_N$. We note that $m_1, \ldots, m_N$ are outputted by $\mathcal{W}(m)$.

Suppose that the pirate decoder $\mathcal{D}$ is produced from a collusion of k users $u_{j_1}, \ldots, u_{j_k}$, we show that all the information the pirate decoder knows from the unique query $(\sigma_1, \ldots, \sigma_N, \tau_1, \ldots, \tau_N)$ are the k messages $m_{j_1}, \ldots, m_{j_k}$. For this, we use a hybrid argument as follows.

Denote by $i_1, \ldots, i_{N-k}$ the $N - k$ indexes that are not in $\{j_1 \ldots j_k\}$. At each q^{th} step (q runs from 1 to $N - K$), we replace σ_{i_q} by $\sigma_{i_q} = \mathsf{PKE.Enc}_{pk_{i_q}}(dk'_{i_q} || h_{i_q})$, where dk'_{i_q} is randomly chosen in the key space, and τ_{i_q} by $\tau_{i_q} = \mathsf{DEM}_{dk'_{i_q}}(m'_{i_q})$, where m'_{i_q} is randomly chosen in the message space. The pirate decoder has thus, after the q^{th} step, no information about m_{i_q}. Because PKE and DEM are semantically secure, the pirate decoder has a negligible advantage to distinguish between two successive steps.

The message $m^\star$ is thus produced from $m_{j_1}, \ldots, m_{j_k}$. At this stage, we can use the algorithm $\mathcal{W}'$ to reveal one of $m_{j_t} \in \{m_{j_1} \ldots m_{j_k}\}$ and can correctly return the user j_t as a traitor. $\square$

Security of Encryption. The PSTBasic scheme can be considered as a particular case of our StatefulBasic scheme with $dk_1 = \ldots = dk_N = dk$ and $m_1 = \ldots = m_N = m$. Using the argument in the security proof of PSTBasic [10], our scheme achieves the same security level, *i.e.* semantic security against "Replayable" CCA adversaries.

Remarks. The above basic scheme is inefficient in comparison with PSTBasic scheme because of its linear ciphertext rate. However, its main advantage is that it can be used for stateful pirate decoders and that the tracing procedure is very efficient with only one query to each pirate decoder. More interestingly, this basic scheme, although inefficient, helps us to construct an efficient general scheme which is almost as efficient as the PST general scheme [10].

The construction of the general scheme is described in the next section.

5 General Scheme for Stateful Pirate Decoders

5.1 IPP codes

Let $\mathcal{Q}$ be an alphabet containing q symbols. If $C = \{w_1, w_2, \ldots, w_N\} \subset \mathcal{Q}^\ell$, then C is called a q-ary code of size N and length ℓ. Each $w_i \in C$ is called a codeword and we write $w_i = (w_{i,1}, w_{i,2}, \ldots, w_{i,\ell})$ where $w_{i,j} \in \mathcal{Q}$ is called the j^{th} component of the codeword w_i.

We define *descendants* of a subset of codewords as follows. Let $X \subset C$ and $u = (u_1, u_2, \ldots, u_\ell) \in \mathcal{Q}^\ell$. The word u is called a descendant of X if for any $1 \leq j \leq \ell$, the j^{th} component u_j of u is equal to a j^{th} component of a codeword in X. In this case, codewords in X are called *parent codewords* of u. For example, $(3, 2, 1, 3)$ is a descendant of the three codewords $(3, 1, 1, 2)$, $(1, 2, 1, 3)$ and $(2, 2, 2, 2)$.

3 1 **1** 2		
1 **2 1 3**	⟵	parent codewords
2 **2** 2 2		
3 2 1 3	⟵	a descendant

We denote by $\mathsf{Desc}(X)$ the set of all descendants of X. For a positive integer c, denote by $\mathsf{Desc}_c(C)$ the set of all descendants of subsets of up to c codewords. Codes with identifiable parent property (IPP codes) are defined as follows.

Definition 6. A code C is called c-IPP if, for any $u \in \mathsf{Desc}_c(C)$, there exists a $w \in C$ such that for any $X \subset C$, if $|X| \leq c$ and $u \in \mathsf{Desc}(X)$, then $w \in X$.

In a c-IPP code, given a descendant $u \in \mathsf{Desc}_c(C)$, we can always identify at least one of its parent codewords. Binary c-IPP codes (with more than two codewords) do not exists, thus in any c-IPP code, the alphabet size $q \geq 3$. Some typical constructions are in [12]. The best known algorithms construct c-IPP codes and c-collusion secure codes [2] with logarithmic length in number of codewords.

5.2 Framework for Combination of Basic Schemes

In [8, 5, 10], the authors constructed general schemes from combinations of basic schemes by using collusion secure codes or IPP codes. We resume this framework, in the case of using IPP codes, as follows:

Primitives: $BasicSch_1, \ldots, BasicSch_\ell$ are basic schemes of q users. $\mathcal{C}$ is a q-ary c-IPP code of length ℓ.

Step 1: Each user is associated to a codeword $w = w_1 \ldots w_\ell$ ($w_i \in \{1, \ldots, q\}$) of $\mathcal{C}$. This codeword determines the user key which contains ℓ sub-keys corresponding to ℓ basic schemes $BasicSch_1, \ldots, BasicSch_\ell$: the w_i^{th} key in scheme $BasicSch_i$, for each $i = 1, 2, \ldots, n$.

Step 2: The general scheme is combined from ℓ basic schemes:

$$GeneralSch = (BasicSch_1, \ldots, BasicSch_\ell)$$

Step 3: For tracing, we first find out the codeword $w^\star$ associated to the pirate decoder $\mathcal{D}$ by finding out each $w_i^\star$ as follows:
- create valid ciphertexts in $BasicSch_1, \ldots, BasicSch_{i-1}, BasicSch_{i+1}, \ldots, BasicSch_\ell$: $c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_\ell$
- create probe ciphertext (ciphertext for tracing) in $BasicSch_i$: $c_i^\star$
- give the combined ciphertext $\mathsf{c} = (c_1, \ldots, c_{i-1}, c_i^\star, ci+1, \ldots, c_\ell)$ to the pirate decoder $\mathcal{D}$. On feedback of $\mathcal{D}$, by using the tracing algorithm in $BasicSch_i$, find out $w_i^\star$.

Step 4: from $w^\star = w_1^\star \ldots w_\ell^\star$, the codeword associated to the pirate decoder, thanks to the identifiable parent property of codes, we can trace back one traitor.

In the PST general scheme, in Step 2, $BasicSch_1, \ldots, BasicSch_\ell$ are independent instances of the PSTBasic. We can replace $BasicSch_1, \ldots, BasicSch_\ell$ by our basic schemes StatefulBasic. However, in this case, we lost a q factor of efficiency in comparison with the PST general scheme, as each of our StatefulBasic scheme loses a linear factor in comparison with PSTBasic scheme.

By using 3-ary IPP codes ($q = 3$), we only loose a small constant factor of 3. We would like nevertheless to improve the efficiency to make it comparable to the PST scheme. The idea is to include a StatefulBasic scheme in a sequence of PSTBasic schemes. More precisely, in Step 2, we replace one of ℓ PSTBasic schemes by one StatefulBasic scheme. Due to the independence between basic schemes, such a combination works well for encryption and decryption. We are only worrying about the tracing capability. Below, we present such a combination with traceability. We note however that this technique of combination cannot be used for constructions where basic schemes share common data. In [5], in order to improve the efficiency, the general scheme combines basic schemes which share some common data. Our technique is thus not suitable to make the scheme in [5] resistant against stateful pirate decoders.

5.3 General Scheme

Let $C = \{\omega_1, \ldots, \omega_N\}$ be a q-ary c-IPP code that allows collusion of up to c users. The N-user general scheme is a combination of ℓ basic schemes PSTBasic $S_1, S_2, \ldots, S_\ell$ and a basic scheme StatefulBasic, each basic scheme supports q users:

Setup: Given security parameters λ and c, the algorithm works as follows:

- For each $j = 1, \ldots, \ell$, call the algorithm $\mathsf{PSTBasic.Gen}(1^\lambda, q)$ to generate an encryption key pk_j and q decryption keys $sk_{j,1}, \ldots, sk_{j,q}$ for the q-user system S_j.
- For each $j = 1, \ldots, \ell$, call the algorithm $\mathsf{StatefulBasic.Gen}(1^\lambda, q)$ to generate an encryption key $\overline{pk_j}$ and q decryption keys $\overline{sk}_{j,1}, \ldots, \overline{sk}_{j,q}$ for the q-user system $\overline{S}_j$.
- Public key pk is defined by the tuple $(pk_i)_{i=1,\ldots,\ell}, (\overline{pk}_i)_{i=1,\ldots,\ell}$ and the code C.
- Private key K_i of each user i (for $i = 1, \ldots, N$) contains a codeword $w_i \in C$, $\overline{sk}_i$, the ℓ-tuple key $sk_{1,w_{i,1}}, sk_{2,w_{i,2}}, \ldots, sk_{\ell,w_{i,\ell}}$ and the ℓ-tuple key $\overline{sk}_{1,w_{i,1}}, \overline{sk}_{2,w_{i,2}}, \ldots, \overline{sk}_{\ell,w_{i,\ell}}$, where $w_{i,j} \in \mathcal{Q} = \{1, 2, \ldots, q\}$ is the symbol at the j^{th} position of the codeword w_i.

$$K_i \doteq (w_i, sk_{1,w_{i,1}}, sk_{2,w_{i,2}}, \ldots, sk_{\ell,w_{i,\ell}}, \overline{sk}_{1,w_{i,1}}, \overline{sk}_{2,w_{i,2}}, \ldots, \overline{sk}_{\ell,w_{i,\ell}})$$

Encrypt(pk, m): The plaintext space of the ℓ-key system is $\mathcal{M}^\ell$. On input $m = (m_1, m_2, \ldots, m_\ell)$, the encryption algorithm works as follows:
- an index k is randomly chosen $k \xleftarrow{R} i = 1, \ldots, \ell$.

– the k^{th} component c_k is encrypted by StatefulBasic $\overline{S}_k$:

$$c_k = \mathsf{StatefulBasic.Enc}_{\overline{pk}_k}(m_k) = (\sigma_{k,1}, \ldots, \sigma_{k,q}, \tau_{k,1}, \ldots, \tau_{k,q})$$

– for $j = 1, \ldots, \ell$, $j \neq k$, the j^{th} component c_k is encrypted by PSTBasic S_j:

$$c_j = \mathsf{PSTBasic.Enc}_{pk_j}(m_j) = (\sigma_{j,1}, \ldots, \sigma_{j,q}, \tau_j)$$

– the ciphertext $\mathsf{c} \doteq (k, c_1, c_2, \ldots, c_\ell)$

Decrypt$(j, K_j, \mathsf{c}, \mathsf{pk})$**:** On the ciphertext $(k, c_1, c_2, \ldots, c_\ell)$, user i uses his secret key to compute:

$$m_k = \mathsf{StatefulBasic.Dec}_{\overline{sk}_{k,w_{i,k}}}(c_k)$$

$$m_j = \mathsf{PSTBasic}_{sk_{j,w_{i,j}}}(c_j), \text{ for } j = 1, \ldots, \ell, j \neq k$$

Trace$(\mathcal{D}, \mathsf{pk})$**:** Let $E = \{1, \ldots \ell\}$. Repeat the following steps until $E = \varnothing$.

– randomly choose an index $k \xleftarrow{R} i = 1, \ldots, \ell$, $E = E \backslash \{k\}$
– choose a message $m = (m_1, m_2, \ldots, m_\ell)$
– create components: $c_j = \mathsf{PSTBasic.Enc}_{pk_j}(m_j)$, for $j = 1, \ldots, \ell, j \neq k$
– create component $c_k = \mathsf{StatefulBasic.Enc}_{\overline{pk}_k}(m_k)$
– define $\mathsf{c} = (k, c_1, c_2, \ldots, c_\ell)$ and give c to the pirate decoder $\mathcal{D}$
– on feedback $m' = (m'_1, m'_2, \ldots, m'_\ell)$, extract the component m'_k and use the same argument used in the tracing algorithm of StatefulBasic scheme to get w_k.
– from the descendant codeword $w = (w_1 || \ldots || w_\ell) \in \mathcal{Q}^\ell$, identify one of its parent codewords. The user associated with this codeword is a traitor.

Traceability. We first consider the traceability of the above system.

Proposition 7. *Suppose that $\mathcal{D}$ is a stateful pirate decoder, the above tracing algorithm can correctly trace back a traitor.*

Proof. We remark that, in the above tracing algorithm, query ciphertexts are identical to valid ciphertexts. Therefore, a pirate decoder cannot detect whether it is being traced. Consequently, a stateful pirate decoder always decrypts correctly as it would do. In this case, a stateful pirate decoder is not more powerful than a stateless one.

Using the same arguments from Theorem 5, the tracing procedure can reveals each w_k as the tracing procedure in StatefulBasic can correctly reveal a traitor. Therefore, the tracing can correctly associate a descendent codeword $w = (w_1 || \ldots || w_\ell)$ of the set of codewords corresponding to the users in the collusion. By the property of IPP codes, the tracing algorithm can thus identify at least one traitor.

Security of Encryption. We now consider the security of the encryption. For this, one could use the following assumption, used in [8, 5, 10]:

Assumption 8 (threshold assumption) *A pirate-decoder that only returns correctly a fraction p of a plaintext of length λ where $1 - p$ is a non-negligible function in λ, is useless.*

We emphasis that, as already mentioned in [8], by employing an all-or-nothing transform [11, 4], this assumption is not necessary.

Proposition 9. *In the general scheme, a collusion of users in the $(\ell - 1)$ basic schemes does not affect the security of the remaining basic scheme.*

Proof. The proof follows the one in [5]. Assume there is an adversary $\mathcal{A}$ that, having information I of the targeted system, called $Basic_1$, and also all the information for ℓ remained systems, called $Basic_2, \ldots, Basic_n$ can get an advantage ε for breaking the system $Basic_1$ (for some goal G). We can then construct an algorithm $\mathcal{B}$ that, having only the information I of the system $Basic_1$, can break the system $Basic_1$ (for the goal G) with advantage ε. Indeed, the algorithm $\mathcal{B}$ can perfectly simulate all the information about $Basic_2, \ldots, Basic_n$ systems by generating itself all parameters for $Basic_2, \ldots, Basic_n$. Because the $Basic_1, \ldots, Basic_n$ systems are totally independent that they do not have any common information, this simulation of $\mathcal{B}$ is perfect. □

This proposition shows that the security of the general scheme is at least the same as the security of each basic scheme. Therefore, the encryption in the above general system is secure.

Efficiency: The ciphertext contains two parts: ciphertext body $(\tau_1, \ldots, \tau_\ell)$ and ciphertext header $(\sigma_{j,1}, \ldots, \sigma_{j,q})_{j=1,\ldots,\ell}$. Between ℓ sub-ciphertext bodies $(\tau_1, \ldots, \tau_\ell)$, $\ell - 1$ sub-ciphertext bodies correspond to the basic schemes PSTBasic and only one sub-ciphertext bodie corresponds to the basic scheme StatefulBasic. As in PSTBasic scheme, the ciphertext body approximately has the same size as the plaintext, and as ℓ is large, the ciphertext body in our general scheme approximately has the same size as the original message. Concerning the ciphertext header, as we use the hybrid framework, each $\sigma_{j,k}$ is significantly smaller than τ_j. Therefore, the header's size is small compared to the message size and the ciphertext rate is almost optimal (rate =1). Without the hybrid argument, the header rate is also small (this rate $= q = 3$) and therefore, the ciphertext rate still remains constant.

6 Conclusion

We proposed the first constant ciphertext rate traitor tracing for stateful pirate decoders. Our scheme moreover supports black-box public traceability. However, due to the use of IPP codes, our scheme does not support full collusion as schemes for stateless pirate decoders in [1, 3]. We raise thus the open question of constructing a fully collusion resistant scheme for stateful pirate decoders with constant ciphertext rate.

Acknowledgments

We would like to thank Antoine Joux and Louis Goubin for helpful discussions and suggestions.

References

1. Dan Boneh, Amit Sahai, and Brent Waters. Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of LNCS, pages 573–592, St. Petersburg, Russia, May 2006. Springer-Verlag, Berlin, Germany.
2. Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data. In Don Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 452–465, Santa Barbara, CA, USA, August 27–31, 1995. Springer-Verlag, Berlin, Germany.
3. Dan Boneh and Brent Waters. A Fully Collusion Resistant Broadcast, Trace and Revoke System, 2006. (To Appear in) Proceedings of 13th ACM Conference on Computer and Communications Security (*ACM CCS 2006*). `http://crypto.stanford.edu/~dabo/papers/tr.pdf`.
4. Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz and Amit Sahai. Exposure-Resilient Functions and All-or-Nothing Transforms. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 453–469, Bruges, Belgium, May 14–18, 2000. Springer-Verlag, Berlin, Germany.
5. Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public Traceability in Traitor Tracing Schemes. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 542–558, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany.

6. Amos Fiat and Tamir Tassa. Dynamic Traitor Traicing. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 354–371, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.
7. Aggelos Kiayias and Moti Yung. On Crafty Pirates and Foxy Tracers. In T. Sander, editor, *ACM Workshop in Digital Rights Management – DRM 2001*, volume LNCS 2320, pages 22–39. Springer, 2001.
8. Aggelos Kiayias and Moti Yung. Traitor Tracing with Constant Transmission Rate. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 450–465, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany.
9. Joe Kilian, F. Thompson Leighton, Lesley R. Matheson, Talal G. Shamoon, Robert E. Tarjan, and Francis Zane. Resistance of digital watermarks to collusive attacks. In *Proceedings of the 1998 IEEE International Symposium on Information Theory*, page 271, 1998.
10. Duong Hieu Phan, Reihaneh Safavi-Naini, and Dongvu Tonien. Generic Construction of Hybrid Public Key Traitor Tracing with Full-Public-Traceability. In M. Bugliesi, editor, *33rd ICALP*, volume 4052 of *LNCS*, pages 630–648, Venice - Italy, July 9–16, 2006. Springer-Verlag, Berlin, Germany.
11. Ronald L. Rivest. All-or-Nothing Encryption and the Package Transform. In *Proceedings of the 4th FSE*, LNCS 1267. Springer-Verlag, Berlin, 1997.
12. Tran Van Trung and Sosina Martirosyan. New Constructions for IPP Codes. *Des. Codes Cryptography*, 35(2):227–239, 2005.