# Optimal Public Key Traitor Tracing Scheme in Non-Black Box Model

Philippe Guillot[1], Abdelkrim Nimour[2], Duong Hieu Phan[1], and Viet Cuong Trinh[1]

[1]Université Paris 8, LAGA, CNRS, (UMR 7539), Université Paris 13, Sorbonne Paris Cité 2 rue de la liberté, F-93526 Saint-Denis, Cedex
[2]Canal+, Paris, France

**Abstract.** In the context of secure content distribution, the content is encrypted and then broadcasted in a public channel, each legitimate user is provided a decoder and a secret key for decrypting the received signals. One of the main threat for such a system is that the decoder can be cloned and then sold out with the pirate secret keys. Traitor tracing allows the authority to identify the malicious users (are then called traitors) who successfully collude to build pirate decoders and pirate secret keys. This primitive is introduced by Chor, Fiat and Naor in '94 and a breakthrough in construction is given by Boneh and Franklin at Crypto '99 in which they consider three models of traitor tracing: *non-black-box tracing* model, *single-key black box tracing* model, and *general black box tracing* model.
Beside the most important open problem of optimizing the black-box tracing, Boneh-Franklin also left an open problem concerning non-black-box tracing, by mentioning: "it seems reasonable to believe that there exists an efficient public key traitor tracing scheme that is completely collusion resistant. In such a scheme, any number of private keys cannot be combined to form a new key. Similarly, the complexity of encryption and decryption is independent of the size of the coalition under the pirate's control. An efficient construction for such a scheme will provide a useful solution to the public key traitor tracing problem".
As far as we know, this problem is still open. In this paper, we resolve this question in the affirmative way, by constructing a very efficient scheme with all parameters are of constant size and in which the full collusion of traitors cannot produce a new key. Our proposed scheme is moreover dynamic.

**Keywords:** traitor tracing, non-black-box tracing, full collusion, pairings

## 1 Introduction

*Traitor tracing*, introduced in [12], is an important cryptographic primitive in the context of secure content distribution. Traitor tracing is a main ingredient in many practical applications of global networking such as pay-per-view television, satellite transmission. In secure content distribution, the content is encrypted and broadcasted in a public channel, each legitimate user is provided a decoder and a secret key for decrypting the received signals. The main threat in this context is that the decoder can be cloned or be produced and then sold out with the pirate secret keys. Traitor tracing allows the authority to identify the malicious users (are then called traitors) who successfully collude to build pirate decoders and pirate secret keys.

A breakthrough was proposed by Boneh-Franklin in [6] in which an efficient public key traitor tracing scheme was introduced. They considered three following tracing models:

1. *Non-black-box tracing* model considers the situation where the collusion of $t$ traitors can derive a new valid secret key. The tracing algorithm takes as inputs this new valid secret key and outputs at least a traitor in the collusion.
2. *Single-key black box tracing* model extends a bit the non-black-box tracing model. It always considers the scenario that the collusion of $t$ traitors can derive a new valid secret key and then this new valid secret key is embedded in a pirate decoder. The tracing algorithm takes as inputs the pirate decoder and should be able to output the identity of one of the traitors.
3. *General black box tracing* model is the strongest model of tracing in which the tracer cannot open the pirate decoder and only interact with it in a black box manner by sending the ciphertext and observing the output of the pirate decoder. It is required that whenever the pirate can decrypt the ciphertext, the tracer should be able to trace back one of the traitors.

### 1.1 Non-Black-Box Tracing vs. General Black Box Tracing

The *general black box tracing* is evidently the most desired model as it covers all the possible strategies of the pirate. However, all the schemes in this model are still quite impractical. The most efficient black box traitor tracing are code based schemes [15,2,8,17]. However, the main weakness of code based schemes is that the user's secret key is long (at least $O(t^2 \log N)$ where $t, N$ are the number of traitors and of users in the system) and thus it cannot be highly protected as one cannot put a long key in a tamper-resistant memory in a smart-card. Moreover, the leakage of some small part of the key can be efficiently used in the attack as shown in Pirates 2.0 [3]. Therefore, these schemes are still far to be applicable in practice. Algebraic schemes achieve the *general black box tracing* [6,16,9,10] in inefficient ways: either the tracing algorithm is of exponential time complexity [6,16], or the ciphertext size is still large (*i.e.,* $O(\sqrt{N})$) and the constructions make use of bilinear maps in groups of composite order [9,10]. These two last schemes are very interesting in the sense that they can deal with full collusion.

While it seems a very difficult and challenging problem to achieve a practical *general black box tracing*, it's of practical interest in considering the weaker models of the non-black-box tracing and the single-key black box tracing. Moreover, these models are also very practical, there are many scenarios that these models are suitable, as also discussed in [19,14].

Let us explain some details in the context of pay-TV. In the majority of the existing systems, each user has been provided a Set-Top box (STB) and a smartcard. The secret key of the user is stored in the smartcard which has the role of decrypting the session key for every crypto-period (between 2 and 10 seconds), this session key is then transmitted to the STB for decrypting the content. The pirate always wants to minimize the cost of distribution of his solution and in practice, he really wants to try to produce new pirate smartcard to be used in already deployed STBs. It is thus necessary that these pirate smartcard are compatible with the STBs in the fields (including the legitimate STBs). As a consequence, the smartcard should preserve the functionality of the legitimate smartcard and it has to embed a pirate but valid key in the memory. It is often in reality that the authority can reverse this key in the memory of the pirate smartcard and the scenario exactly falls in the non-black-box tracing model. Even if the tracer cannot reserve the memory of the pirate smartcard, we argue that the single-key pirate tracing model is suitable. Indeed, in the modern CAS (Conditional Access Systems), the session key is delivered at the last moment so that there is only a small delay between the time the smartcard decrypts the session key and the time the decoder receive the encrypted content. Therefore, if the pirate card (which is evidently cannot more performant than a legitimate smartcard) always try to decrypt the session key with different, say two, keys, it will fails to decrypt the content in time and will give the STB the session key after the encrypted content arrive for that crypto-period. One could wonder what happens if the pirate decoder only try to detect the presence of tracing algorithm from time to time. Fortunately, the single-key black box tracing algorithms, as in Boneh-Franklin schemes and in our scheme, only need to ask just one query and the decoder is resettable in practice, this strategy of pirate does not work. All in all, we would like to argue that the non-black-box tracing model and the single-key black box tracing model, though much weaker than the general black box tracing model and cannot thus cover all the strategies of the pirate, are still very practical. In fact, there are quite a lot of interesting works that only concentrate on these models, namely [19,14,1].

In a theoretical point of view, it's also a very interesting problem to consider non-black-box tracing because there is still no optimal solution, far from that, in spite of many efforts. Indeed, the Boneh-Fraklin is efficient with respect to the non-black-box tracing and single-key black box tracing but its ciphertext size is still linear in the number of traitors. The Tonien-Safavi scheme [19] and the Junod-Karlov-Lenstra sheme [14] managed to improve the tracing algorithm but the ciphertext size is always linear in the number of traitors. A side effect of this high ciphertext size in the number of traitors is that these schemes cannot be used with full collusion because in this later case, these schemes are worse than the trivial scheme of assigning each user an independent key. Agrawal *et. al.* [1] go one step further by achieving an intermediate level between bounded tracing (when one assumes a maxmixum $t$ number of traitors) and full collusion: they allow the pirate to collect up to $t$ keys and get some bounded partial information about the others keys. We notice that the authors in [1] only considers the non-black-box tracing model and therefore a full collusion resistant scheme in the non-black-box tracing model satisfies immediately their security notion proposed. All in all, there is still an important gap between the efficiency of all these schemes and an optimal solution: the ciphertext size depends on the number of traitors and none of them can deal with full collusion. Our objective is to close this gap.

### 1.2 Our Contributions

We consider the *non-black-box tracing* and the *single-key black box tracing* models for which we propose an optimal scheme in the sense that all the parameters including private key size, public key size, ciphertext size, encryption and decryption time complexity are constant. In addition, our scheme also achieves two interesting properties of a public key traitor tracing scheme: it is fully collusion resistant and dynamic where there is no need to update any parameter when a user joins the system. We also highly improve the time complexity in tracing algorithms, in particular we achieve $O(1)-$time non-black-box tracing. Regarding the single-key black box tracing, we consider both the full access model (where the decoder pirate has to return the correct message for any valid ciphertext) and the minimal access model (where the pirate decoder only needs to return a single bit signifying whether the ciphertext is valid or not). We then design a $O(\log N)-$time full access single-key black box tracing and a $O(N)-$time minimal access single-key black box tracing.

The detailed comparison between our scheme and other schemes is given in the full version of this paper [18]. We notice that our scheme is the only scheme that allows minimal access single-key black box tracing.

The main weakness in our scheme is that the security for the tracing problem is based on a type of $q$-assumption. However, we notice that these types of assumptions have been widely used in security proofs, for example in [4,5], [7,11]. We also prove that the proposed assumptions hold in the generic group.

## 2 Preliminaries

### 2.1 Traitor Tracing Scheme

We refine the definition of a non-black-box public key traitor tracing scheme from [6]. Formally, a non-black-box public key traitor tracing encryption scheme is made up of the following algorithms:

**Setup**($\lambda$)**:** Takes as input the security parameter $\lambda$, it returns a master key $\mathsf{msk}$ and a public key $\mathsf{mpk}$.

**Joint**($i, \mathsf{msk}$)**:** Takes as inputs a user's index $i$, together with the master key, and outputs a user's secret key $sk_i$.

**Encrypt**($M, \mathsf{mpk}$)**:** Takes as inputs a message $M$, together with the public key, and outputs a ciphertext $C$.

**Decrypt**($sk_i, \mathsf{mpk}, C$)**:** Takes as inputs a secret key $sk_i$, public key, and a ciphertext $C$, outputs the corresponding message $M$.

**Trace**$(\mathcal{D}, sk^*, \mathsf{tracing}-\mathsf{key}) \rightarrow i$**:** Takes as input the public key $\mathsf{mpk}$, the $\mathsf{tracing}-\mathsf{key}$, a pirate decoder $\mathcal{D}$ and some valid secret key $sk^*$ embedded in $\mathcal{D}$ and outputs an index $i$ corresponding to an accused traitor.

When the knowledge of the tracer about the pirate decoder is more restricted, one can get the stronger following notions, which were discussed in [6]:

- in the single-key black box tracing model, the tracing algorithm only takes as inputs the public key $\mathsf{mpk}$, the $\mathsf{tracing}-\mathsf{key}$, and interact with a pirate decoder $\mathcal{D}$ with the assumption that the pirate decoder only embed a single valid key $sk^*$.
- in the general black box tracing model, there is no any assumption on the pirate decoder and the tracer can only interact with it. It is however required that $\mathcal{D}$ can decrypt the well-form ciphertexts with a non-negligible probability because otherwise the pirate decoder is useless.

For correctness, we require that for all $i \in \mathbb{N}$, if $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathbf{Setup}(\lambda)$, $sk_i \leftarrow \mathbf{Joint}(i, \mathsf{msk})$ and $C \leftarrow \mathbf{Encrypt}(M, \mathsf{mpk})$ then one should get $M = \mathbf{Decrypt}(sk_i, C)$.
The security of the scheme is defined in terms of two properties: semantic security and tracing security.

*Semantic security* The standard notion of semantic security requires that, for any PPT $\mathcal{A}$, we have $|\Pr[A\ wins] - 1/2|$ is negligible in the following game:

- In the setup phase, the challenger runs $\mathbf{Setup}(\lambda)$ algorithm to get a master key $\mathsf{msk}$ and a public key $\mathsf{mpk}$. It then gives $\mathsf{mpk}$ to $\mathcal{A}$.
- In the challenge phase, $\mathcal{A}$ outputs two messages $M_0, M_1$. The challenger then chooses a bit $b \in \{0, 1\}$ at random, sets $C \leftarrow \mathbf{Encrypt}(M_b, \mathsf{mpk})$, and gives $C$ to $\mathcal{A}$.
- In the guess phase, the attacker $\mathcal{A}$ outputs a bit $b'$. We say $\mathcal{A}$ wins if $b' = b$.

*non-black-box tracing security* We say that a secret key $sk$ is a valid secret key iff there exists some message $M$ in message-domain such that if $C = \mathbf{Encrypt}(M, \mathsf{mpk})$ then one should get $M = \mathbf{Decrypt}(sk, C)$ with probability at least $\frac{1}{2}$.
We say that non-black-box tracing security holds if, for any PPT $\mathcal{A}$, we have $|\Pr[challenger\ wins] - 1/2|$ is considerable in the following game:

- In the setup phase, the challenger runs $\mathbf{Setup}(\lambda)$ algorithm to get a master key $\mathsf{msk}$ and a public key $\mathsf{mpk}$. It then gives $\mathsf{mpk}$ to $\mathcal{A}$.
- In the query phase, $\mathcal{A}$ may adaptively ask corrupt query for user index $i$ and gets $sk_i$.
- At some point $\mathcal{A}$ outputs some $sk^*$ and a pirate decoder $\mathcal{D}$ in which $sk^*$ is embedded in. The challenger then runs $\mathbf{Trace}(\mathcal{D}, sk^*, \mathsf{tracing}-\mathsf{key}) \rightarrow i$. We say that the challenger wins if the secret key $sk^*$ is a valid secret key and the traced index $i$ is in the set of corrupted indexes.

In the single-key black box tracing security, $\mathcal{A}$ only outputs a decoder $\mathcal{D}$ in which only $sk^*, \mathsf{mpk}$ are embedded in it. In the general black box tracing security, $\mathcal{A}$ only outputs a decoder $\mathcal{D}$ with a requirement that $\mathcal{D}$ can decrypt the well-form ciphertexts with a non-negligible probability because otherwise the pirate decoder is useless.

*Full access black box tracing vs Minimal access black box tracing* These two types of models are discussed in [6].

1. In the full access black box tracing model, the tracer can query the pirate decoder on a ciphertext $C$, if $C$ is a well-form ciphertext, he will always receive the corresponding plantext $M$. Otherwise, the pirate decoder can return an arbitrary output (it can return a signal indicating that the ciphertext $C$ is invalid or can maliciously choose a random message $M'$ and return $M'$).
2. In the minimal access black box tracing model, the tracer queries the pirate decoder on a pair $(C, M)$ and only receives a signal: *valid* if the ciphertex $C$ is a valid encryption of $M$, *invalid* if not.

*Dynamic public key traitor tracing scheme* We adapt the definition of a dynamic broadcast encryption in [13] for a public key traitor tracing scheme, note that our definition is in the *strongest* sense because it requires no any update in the parameters of the systems. Indeed:

1. the system setup as well as the ciphertext size are fully independent from the number of users in the system. The number of users in the system is flexible,
2. a new user can join the system at anytime without implying a modification of preexisting user decryption keys and of the encryption key.

### 2.2 Bilinear Maps

Our scheme employs bilinear maps and related assumptions, which we now recall. Let $\mathbb{G}$ and $\mathbb{G}_T$ denote two finite multiplicative abelian groups of large prime order $p > 2^\lambda$ where $\lambda$ is the security parameter. Let $g$ be a generator of $\mathbb{G}$. We assume that there exists an admissible bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, meaning that for all $a, b \in \mathbb{Z}_p$
(1) $e(g^a, g^b) = e(g, g)^{ab}$,
(2) $e(g^a, g^b) = 1$ iff $a = 0$ or $b = 0$,
(3) $e(g^a, g^b)$ is efficiently computable.
$(p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ is then called a bilinear map group system. We now recall the generalization of the Diffie-Hellman exponent assumption in [5] on bilinear map group system.
Let $(p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ a bilinear map group system and $g \in \mathbb{G}$ be a generator of $\mathbb{G}$, and set $g_T = e(g, g) \in \mathbb{G}_T$. Let $s, n$ be positive integers and $P, Q \in \mathbb{F}_p[X_1, \ldots, X_n]^s$ be two $s$-tuples of $n$-variate polynomials over $\mathbb{F}_p$. Thus, $P$ and $Q$ are just two lists containing $s$ multivariate polynomials each. We write $P =$

$(p_1, p_2, \ldots, p_s)$ and $Q = (q_1, q_2, \ldots, q_s)$ and impose that $p_1 = q_1 = 1$. For any function $h : \mathbb{F}_p \to \Omega$ and vector $(x_1, \ldots, x_n) \in \mathbb{F}_p^n$, $h(P(x_1, \ldots, x_n))$ stands for $(h(p_1(x_1, \ldots, x_n)), \ldots, h(p_s(x_1, \ldots, x_n))) \in \Omega^s$. We use a similar notation for the $s$-tuple $Q$. Let $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. It is said that $f$ depends on $(P, Q)$, which denotes $f \in \langle P, Q \rangle$, when there exists a linear decomposition

$$f = \sum_{1 \leq i,j \leq s} a_{i,j} \cdot p_i \cdot p_j + \sum_{1 \leq i \leq s} b_i \cdot q_i, \qquad a_{i,j}, b_i \in \mathbb{Z}_p$$

Let $P, Q$ be as above and $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. The $(P, Q, f) - \mathsf{GDDHE}$ problem is defined as follows.

**Definition 1.** $((P, Q, f) - \mathsf{GDDHE})$ *[5].*
*Given* $H(x_1, \ldots, x_n) \in \mathbb{G}^s \times \mathbb{G}_T^s$ *as above and* $T \in \mathbb{G}_T$ *decide whether* $T = g_T^{f(x_1, \ldots, x_n)}$.

The $(P, Q, f) - \mathsf{GDDHE}$ assumption says that it is hard to solve the $(P, Q, f) - \mathsf{GDDHE}$ problem if $f$ is independent of $(P, Q)$. In this paper, we will prove our scheme is semantically secure under this assumption.

## 3 Construction

Let $(p, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot))$ a bilinear map group system and $g \in \mathbb{G}$ be a generator of $\mathbb{G}$, our scheme is constructed as follows:

**Setup**$(\lambda)$. The algorithm chooses $e_1, e_2, v \xleftarrow{\$} \mathbb{Z}_p$ then sets $d_1 = e_1^{-1}, d_2 = e_2^{-1}$
The master key $\mathsf{msk}$ is $(e_1, e_2, v)$. The system public keys $\mathsf{mpk}$ is:

$$(g^{d_1}, e(g, g)^{d_2}, g^{d_1 \cdot d_2}, e(g, g), e(g, g)^v, e(g, g)^{d_2 \cdot v})$$

**Joint**$(i, \mathsf{msk})$. For each user $i$ chooses $a_i \xleftarrow{\$} \mathbb{Z}_p$ such that $a_i \neq -1, -v, d_2 - 1$. The secret key for user $i$ is set as: $A_i = g^{e_1(a_i + v)}, B_i = \frac{1}{(a_i + 1)} - e_2$. We call the secret keys in the case $a_i = -v$ or $a_i = d_2 - 1$ are special keys. The users in the system can be assigned to all secret keys in the secret key space except these special keys. Note that the special key, in the case $a_i = -v$, is not useful for decryption.

**Encrypt**$(M, \mathsf{mpk})$. Encryptor picks a random $k$ in $\mathbb{Z}_p$, then computes:

$$C_1 = g^{d_1 \cdot k}, C_2 = e(g, g)^{d_2 \cdot k}, C_3 = g^{d_1 \cdot d_2 \cdot k},$$

$$C_4 = e(g, g)^{k \cdot v}, C_5 = e(g, g)^{d_2 \cdot k \cdot v}, C_6 = e(g, g)^{-k} \cdot M$$

Finally, outputs $C = (C_1, C_2, C_3, C_4, C_5, C_6)$.

**Decrypt**$(A_i, B_i, C)$. User $i'th$ first computes:

$$\frac{e(A_i, C_1)}{C_4} \cdot C_2^{B_i - 1} \cdot (\frac{e(A_i, C_3)}{C_5})^{B_i} = \frac{e(g^{e_1(a_i + v)}, g^{d_1 \cdot k})}{e(g, g)^{k \cdot v}} \cdot e(g, g)^{d_2 \cdot k \cdot (\frac{1}{(a_i + 1)} - e_2 - 1)}.$$

$$\cdot\left(\frac{e(g^{e_1(a_i+v)}, g^{d_1\cdot d_2\cdot k})}{e(g,g)^{d_2\cdot k\cdot v}}\right)^{\frac{1}{(a_i+1)}-e_2} =$$

$$= e(g,g)^{k\cdot a_i}\cdot e(g,g)^{\frac{d_2\cdot k}{(a_i+1)}}\cdot e(g,g)^{-k}\cdot e(g,g)^{-d_2\cdot k}\cdot e(g,g)^{k\cdot a_i\cdot(\frac{d_2}{(a_i+1)}-1)} = e(g,g)^{-k}.$$

then outputs $M = C_6/e(g,g)^{-k}$.

*Intuition about our construction* In the decryption, we emphasize that the crucial element is $(\frac{e(A_i,C_3)}{C_5})^{B_i}$. We remark that, though a pirate can perform a linear combination on the elements $A_i$ in his collected keys, there is no way for the pirate to exploit the combination of his keys to do a linear combination for the elements $(\frac{e(A_i,C_3)}{C_5})$ because $C_5$ is changed for each encryption. Therefore the well-known pirate's strategy of making a linear combination on the collected keys do not work for our scheme. The next section is devoted for formal analysis of security.

## 4 Security

**Definition 2 ($\mathsf{GDDHE}_1$ Assumption).** *The $(t,\varepsilon)-\mathsf{GDDHE}_1$ assumption says that for any t-time adversary $\mathcal{A}$ that is given* $\mathsf{input} = (g, g^x, g^y, g^{xy}, g^{kx}, g^{ky}, g^{kxy})$ *cannot distinguish between a value $e(g,g)^k \in \mathbb{G}_T$ or a random value $T \in \mathbb{G}_T$, where $x, y, k \in \mathbb{Z}_p, g \in \mathbb{G}$, with advantage greater than $\varepsilon$:*

$$\mathbf{Adv}^{\mathsf{GDDHE}_1}(\mathcal{A}) = \left|\begin{array}{l}\Pr[\mathcal{A}(\mathsf{input}, e(g,g)^k) = 1] \\ -\Pr[\mathcal{A}(\mathsf{input}, T) = 1]\end{array}\right| \leq \epsilon.$$

It is not hard to see that $\mathsf{GDDHE}_1$ assumption is a special case of $(P,Q,f)-$ $\mathsf{GDDHE}$ assumption. Indeed, we set $P = (p_1 = 1, p_2 = X, p_3 = Y, p_4 = XY,$ $p_5 = KX, p_6 = KY, p_7 = KXY), Q = (q_1 = 1), f = K$. Suppose that $f$ is not independent to $\langle P, Q\rangle$, *i.e.,* one can find $a_8 \neq 0$ such that the following equation holds for all $X, Y, K \in \mathbb{Z}_p$

$$a_8 f = \sum_{1\leq i,j\leq 7} a_{i,j}\cdot p_i\cdot p_j + b_1\cdot q_1$$

$$\iff a_8 K = (KX+KY+KXY)(a_1+a_2X+a_3Y+a_4XY+a_5KX+a_6KY+a_7KXY)$$

$$\iff a_8 = (X+Y+XY)(a_1+a_2X+a_3Y+a_4XY+a_5KX+a_6KY+a_7KXY)$$

$$\iff (X+Y+XY)(a_1+a_2X+a_3Y+a_4XY+a_5KX+a_6KY+a_7KXY) - a_8 = 0$$

This implies that the constant term $a_8 = 0$ which is a contradiction with the requirement that $a_8 \neq 0$. Therefore, $f$ is independent to $\langle P, Q\rangle$.

**Theorem 1.** *Under the $\mathsf{GDDHE}_1$ assumption, our scheme is semantically secure.*

*Proof.* Assume that there exists an adversary $\mathcal{B}$ who is successful in breaking the semantic security of our scheme, we prove that there also exists an adversary $\mathcal{A}$ which attacks the $\mathsf{GDDHE}_1$ assumption with the same advantage.
We show that $\mathcal{A}$ can simulate the interaction with $\mathcal{B}$ and then use the output of $\mathcal{B}$ to break the $\mathsf{GDDHE}_1$ assumption as follow:
In the setup, $\mathcal{A}$ receives the inputs from his challenger:

$$(g, g^x, g^y, g^{xy}, g^{kx}, g^{ky}, g^{kxy}, T)$$

and needs to distinguish $T$ is either $e(g, g)^k$ or a random value in $\mathbb{G}_T$.
In the next step, $\mathcal{A}$ provides the inputs for $\mathcal{B}$ as follow:
He chooses randomly $z \in \mathbb{Z}_p$, implicitly sets $d_1 = zy, d_2 = x, v = y$, then computes the public key:

$$g^{d_1} = (g^y)^z, e(g, g)^{d_2} = e(g, g^x), g^{d_1 \cdot d_2} = (g^{xy})^z, e(g, g), e(g, g)^v = e(g, g^y),$$
$$e(g, g)^{d_2 \cdot v} = e(g, g^{xy})$$

In the challenge phase, $\mathcal{B}$ outputs two messages $M_0$ and $M_1$. $\mathcal{A}$ chooses randomly a bit $b \in \{0, 1\}$ then computes the challenge ciphertext as follow:

$$C_1 = (g^{ky})^z = g^{d_1 \cdot k}, C_2 = e(g, g^{kx}) = e(g, g)^{d_2 \cdot k}, C_3 = (g^{kxy})^z = g^{d_1 \cdot d_2 \cdot k},$$
$$C_4 = e(g, g^{ky}) = e(g, g)^{k \cdot v}, C_5 = e(g, g^{kxy}) = e(g, g)^{d_2 \cdot k \cdot v}, C_6 = \frac{1}{T} \cdot M_b$$

then gives it to $\mathcal{B}$.
$\mathcal{B}$ outputs its guess $b'$ for $b$. If $b' = b$ the algorithm $\mathcal{A}$ outputs 0 (indicating that $T = e(g, g)^k$). Otherwise, it outputs 1 (indicating that $T$ is random in $\mathbb{G}_T$).
As the simulation of $\mathcal{A}$ is perfect, $\mathcal{A}$ can thus break $\mathsf{GDDHE}_1$ assumption with the same advantage that $\mathcal{B}$ can break the semantic security.

# 5 Traitor Tracing

## 5.1 Non-Black-Box Tracing

**Definition 3** ($\mathsf{GDDHE}_2$ **Assumption).** *The* $(t, \varepsilon) - \mathsf{GDDHE}_2$ *assumption says that for any* $t$*-time adversary* $\mathcal{A}$ *that is given* $(b_1, \ldots, b_l, \mathsf{input})$ *in which* $b_1, \ldots, b_l$ *are random in* $\mathbb{Z}_p$ *and* $\neq 0$,

$$\mathsf{input} = \left( g^{d_1}, g^{d_1 d_2}, g^{\frac{1}{d_1}}, g^{\frac{d_2 - b_1 d_2 - 1}{d_1(b_1 d_2 + 1)}}, \ldots, g^{\frac{d_2 - b_l d_2 - 1}{d_1(b_l d_2 + 1)}} \right)$$

*its probability to output a value* $g^{\frac{d_2}{d_1}} \in \mathbb{G}$*, where* $d_1, d_2 \in \mathbb{Z}_p, g \in \mathbb{G}$*, is bounded by* $\varepsilon$*:*

$$\mathbf{Succ}^{\mathsf{GDDHE}_2}(\mathcal{A}) = \Pr[\mathcal{A}(b_1, \ldots, b_l, \mathsf{input}) = g^{\frac{d_2}{d_1}}] \leq \varepsilon.$$

We show that this assumption holds in the generic group, the details can be found in the full version of this paper [18]. Next, we recall the definition of $\mathsf{Modified}{-}l-\mathsf{SDH}$ assumption from [11].

**Definition 4** (Modified$-l-$SDH **Assumption).**
*Given $g, g^{\alpha} \in \mathbb{G}$ and $l-1$ pairs $\langle w_j, g^{1/(\alpha+w_j)}\rangle \in \mathbb{Z}_p \times \mathbb{G}$ for a fixed parameter $l \in \mathbb{N}$.*
*Output another pair $\langle w, g^{1/(\alpha+w)}\rangle \in \mathbb{Z}_p \times \mathbb{G}$.*

**Theorem 2.** *Under the* $\mathsf{GDDHE}_2$ *assumption and* Modified$-l-$SDH *assumption, our scheme is secure in the non-black-box tracing model.*

*Proof.* It is sufficient for us to show that the collusion of any number of traitors cannot derive a new valid secret key. Then, the proof is automatically followed since at least a traitor's key must be embedded in the pirate decoder and when the tracer reverse this key, the identity of the corresponding traitor is revealed. To prove that the collusion of any number of traitors cannot derive a new valid secret key, we first prove that they cannot derive a special key $A, B$ in which $a = d_2 - 1$, we then prove that they also cannot derive any new valid secret key that differs from this special key.

**Lemma 1.** *Under the* $\mathsf{GDDHE}_2$ *assumption, the collusion of any number of traitors cannot derive a special key $A, B$ in which $a = d_2 - 1$.*

*Proof.* Assume that there is an adversary $\mathcal{B}$ which takes as inputs $l$ traitors' keys, for any number $l$, the system public key, and successfully derive a special key $A, B$ in which $a = d_2 - 1$. We construct an algorithm $\mathcal{A}$ which can simulate the interaction with $\mathcal{B}$ and then use the output of $\mathcal{B}$ to break the $\mathsf{GDDHE}_2$ assumption as follow:
In the setup, $\mathcal{A}$ receives the inputs from his challenger:

$$b_1, \ldots, b_l, g^{d_1}, g^{d_1 d_2}, g^{\frac{1}{d_1}}, g^{\frac{d_2 - b_1 d_2 - 1}{d_1(b_1 d_2+1)}}, \ldots, g^{\frac{d_2 - b_l d_2 - 1}{d_1(b_l d_2+1)}}$$

And needs to output the value $g^{\frac{d_2}{d_1}}$.
In the next step, $\mathcal{A}$ first chooses randomly $v \in \mathbb{Z}_p$, then provides the inputs for $\mathcal{B}$ as follow:

- $\mathcal{A}$ provides a secret key $A_i, B_i, i = 1, \ldots, l$ for $\mathcal{B}$ by setting $B_i = b_i = \frac{1}{a_i+1} - e_2$, therefore implicitly $a_i = \frac{d_2 - b_i d_2 - 1}{(b_i d_2+1)}$, then computes

  $$A_i = g^{\frac{d_2 - b_i d_2 - 1}{d_1(b_i d_2+1)}} \cdot g^{\frac{v}{d_1}} = g^{\frac{a_i}{d_1}} \cdot g^{\frac{v}{d_1}} = g^{e_1(a_i+v)}$$

  where $e_1 = d_1^{-1}, e_2 = d_2^{-1}$. Note that because $b_i, d_1, d_2, v$ are randomly chosen in $\mathbb{Z}_p$, the resulted secret key is also chosen in the same distribution as in the joint algorithm.
- For the public key, $\mathcal{A}$ computes:

  $$g^{d_1}, e(g,g)^{d_2} = e(g^{d_1 d_2}, g^{\frac{1}{d_1}}), g^{d_1 d_2}, e(g,g) = e(g^{d_1}, g^{\frac{1}{d_1}}), e(g,g)^{v},$$

  $$e(g,g)^{v \cdot d_2} = e(g^{d_1 d_2}, g^{\frac{v}{d_1}})$$

When $\mathcal{B}$ outputs the special secret key $A, B$ in which $a = d_2 - 1$

$$A = g^{e_1(d_2+v-1)}, B = 0$$

then $\mathcal{A}$ outputs

$$\frac{A \cdot g^{\frac{1}{d_1}}}{g^{\frac{v}{d_1}}} = g^{\frac{d_2}{d_1}}$$

As a result, the probability that the collusion of any number of traitors can derive a special key $A, B$ in which $a = d_2 - 1$ is the same as the probability that a $t$-time adversary $\mathcal{A}$ who breaks the security of the $\mathsf{GDDHE}_2$ assumption.

**Lemma 2.** *Under the* $\mathsf{Modified}{-}l{-}\mathsf{SDH}$ *assumption, the collusion of any number of traitors cannot derive any new valid secret key that differs from the special key above.*

*Proof.* Assume that there is an adversary $\mathcal{B}$ which takes as inputs $l-2$ traitors' keys, for any number $l$, the system public key, and successfully derive a new valid secret key which is different from these $l-2$ traitors' keys and the special key above. We construct an algorithm $\mathcal{A}$ which can simulate the interaction with $\mathcal{B}$ and then use the output of $\mathcal{B}$ to break the $\mathsf{Modified}{-}l-\mathsf{SDH}$ assumption as follow:
In the setup, $\mathcal{A}$ receives the inputs from his challenger:

$$(w_1, \ldots, w_{l-1}, g, g^{\alpha}, g^{\frac{1}{\alpha+w_1}}, \ldots, g^{\frac{1}{\alpha+w_{l-1}}})$$

In the next step, $\mathcal{A}$ provides the inputs for $\mathcal{B}$ as follow:
He first chooses randomly $e_1, v \in \mathbb{Z}_p$, then implicitly sets $e_2 = \alpha + w_1$ thus $g^{\frac{1}{\alpha+w_1}} = g^{\frac{1}{e_2}} = g^{d_2}$. $\mathcal{A}$ can easily compute the system public keys and gives them to $\mathcal{B}$.
To compute $A_i, B_i, i = 2, \ldots, l-1$, $\mathcal{A}$ sets $B_i = \frac{1}{a_i+1} - e_2 = w_i - w_1$ thus $a_i = \frac{1}{e_2+w_i-w_1} - 1$ and

$$A_i = (g^{\frac{1}{\alpha+w_i}})^{e_1} \cdot g^{e_1(v-1)} = g^{e_1(\frac{1}{e_2+w_i-w_1}+v-1)} = g^{e_1(a_i+v)}$$

Note that $\alpha = e_2 - w_1$.
When $\mathcal{B}$ outputs a new secret key

$$A = g^{e_1(a+v)}, B = \frac{1}{(a+1)} - e_2$$

where $a \neq -1, d_2-1, a_2, \ldots, a_{l-1}$, then $\mathcal{A}$ outputs $w = B + w_1 = \frac{1}{(a+1)} - e_2 + w_1$ thus $a = \frac{1}{e_2+w-w_1} - 1$, and

$$g^{\frac{1}{\alpha+w}} = (\frac{A}{g^{e_1(v-1)}})^{\frac{1}{e_1}} = \frac{g^{a+v}}{g^{v-1}} = g^{a+1} = g^{\frac{1}{e_2+w-w_1}-1+1} = g^{\frac{1}{e_2+w-w_1}} = g^{\frac{1}{\alpha+w}}$$

Note that $a \neq -1, d_2-1, a_2, \ldots, a_{l-1}$ thus $w \neq w_1, \ldots, w_{l-1}$.
As the simulation of $\mathcal{A}$ is perfect, $\mathcal{A}$ can thus break $\mathsf{Modified}{-}l{-}\mathsf{SDH}$ assumption with the same advantage that $\mathcal{B}$ can successfully derive a new valid secret key.

### 5.2 Single-key Black Box Tracing

**Definition 5** ($\mathsf{GDDHE}_3$ **Assumption).** *The* $(t, \varepsilon) - \mathsf{GDDHE}_3$ *assumption says that for any* $t$*-time adversary* $\mathcal{A}$ *that is given a pair* $(b, \mathsf{input})$ *in which* $b \neq 0$ *is random in* $\mathbb{Z}_p$ *and*

$$\mathsf{input} = \left( g, g^{d_1}, g^{d_1 d_2}, g^{\frac{v}{d_1}}, g^{\frac{d_2 - bd_2 - 1}{d_1(bd_2+1)}}, g^{kd_1}, g^{kd_1 d_2}, e(g,g)^{d_2}, e(g,g)^{k} \right)$$

*cannot distinguish between a value* $e(g,g)^{kd_2} \in \mathbb{G}_T$ *and a random value* $T \in \mathbb{G}_T$*, where* $d_1, d_2, v, k \in \mathbb{Z}_p, g \in \mathbb{G}$*, with an advantage greater than* $\varepsilon$*:*

$$\mathbf{Adv}^{\mathsf{GDDHE}_3}(\mathcal{A}) = \left| \begin{array}{l} \Pr[\mathcal{A}(b, \mathsf{input}, e(g,g)^{kd_2}) = 1] \\ - \Pr[\mathcal{A}(b, \mathsf{input}, T) = 1] \end{array} \right| \leq \epsilon$$

We notice that, unlike the $\mathsf{Modified}{-}l{-}\mathsf{SDH}$ assumption, this is a static assumption. We show that this assumption holds in the generic group, the details can be found in the full version of this paper [18].

**Theorem 3.** *Under the* $\mathsf{GDDHE}_3$ *assumption, our scheme is secure in the single-key black box tracing model.*

*Proof.* We note that in the single-key black box tracing model, there are two separate functions which are called the *key-builder* and the *box-builder*. In the first one, the traitors will collude to derive a new valid secret key. In the second one, one receives this new secret key and build a pirate decoder based on it.
In our proof we first prove that the pirate decoder takes as inputs a secret key and the public key, cannot distinguish a probe ciphertext and a well-form ciphertext, therefore it will run the decryption algorithm normally. Finally, we present a tracing algorithm in which the tracer creates a probe ciphertext and then queries the pirate decoder on this probe ciphertext. After the pirate decoder outputs the answer, the tracer can identify the secret key that pirate decoder is using to decrypt.

Assume that there is a pirates decoder $\mathcal{B}$, on inputs a secret key and the public key, can successfully distinguish a probe ciphertext and a well-form ciphertext. We show that $\mathcal{A}$ can simulate the interaction with $\mathcal{B}$ and then use the output of $\mathcal{B}$ to break the $\mathsf{GDDHE}_3$ assumption:
In the setup, $\mathcal{A}$ receives the inputs from his challenger:

$$b, g, g^{d_1}, g^{d_1 d_2}, g^{\frac{v}{d_1}}, g^{\frac{d_2 - bd_2 - 1}{d_1(bd_2+1)}}, g^{kd_1}, g^{kd_1 d_2}, e(g,g)^{d_2}, e(g,g)^{k}, T$$

with $b, d_1, d_2, v, k$ are randomly chosen in $\mathbb{Z}_p$, and needs to distinguish $T$ is $e(g,g)^{kd_2}$ or not.
In the next step, $\mathcal{A}$ provides the inputs for $\mathcal{B}$ as follow:

– $\mathcal{A}$ provides a secret key for $\mathcal{B}$ by setting $B = b = \frac{1}{a+1} - e_2$, therefore implicitly $a = \frac{d_2 - bd_2 - 1}{(bd_2+1)}$, then computes

$$A = g^{\frac{d_2 - bd_2 - 1}{d_1(bd_2+1)}} \cdot g^{\frac{v}{d_1}} = g^{\frac{a}{d_1}} \cdot g^{\frac{v}{d_1}} = g^{e_1(a+v)}$$

where $e_1 = d_1^{-1}, e_2 = d_2^{-1}$. Note that because $b, d_1, d_2, v$ are randomly chosen in $\mathbb{Z}_p$, the resulted secret key is also chosen in the same distribution as in the joint algorithm.

– For the public key, $\mathcal{A}$ computes:

$$g^{d_1}, e(g,g)^{d_2}, g^{d_1 \cdot d_2}, e(g,g), e(g,g)^v = e(g^{d_1}, g^{\frac{v}{d_1}}), e(g,g)^{v \cdot d_2} = e(g^{d_1 d_2}, g^{\frac{v}{d_1}})$$

$\mathcal{A}$ next chooses a random message $M$ and uses $T$ to compute the challenge ciphertext and passes it to $\mathcal{B}$:

$$g^{kd_1}, T, g^{kd_1 d_2}, e(g^{\frac{v}{d_1}}, g^{kd_1}), e(g^{\frac{v}{d_1}}, g^{kd_1 d_2}), e(g,g)^{-k} \cdot M$$

In the guess phase, if $\mathcal{B}$ outputs 0 (indicating that this is well-form ciphertext) then $\mathcal{A}$ outputs 0 (indicating that $T$ is $e(g,g)^{kd_2}$), and otherwise if $\mathcal{B}$ outputs 1 (indicating that this is probe ciphertext) then $\mathcal{A}$ also outputs 1 (indicating that $T$ is a random element).
We also note that $\mathcal{B}$ can maliciously output a random message $M'$ in the case he knows the challenge ciphertext is a probe ciphertext, however $\mathcal{A}$ still knows the right answer of $\mathcal{B}$ because he knows the real message $M$.
As the simulation of $\mathcal{A}$ is perfect, $\mathcal{A}$ can thus break $\mathsf{GDDHE}_3$ assumption with the same advantage that $\mathcal{B}$ can successfully distinguish a probe ciphertext and a well-form ciphertext. We can thus construct a single-key black box tracing algorithm as follow:

**Full Access Single-key Black Box Tracing Algorithm:** When a user $j$ joins the system, the tracer computes and stores the pair $(j, e(g,g)^{B_j})$ in a sorted table $Tab$. The tracing algorithm then works as follow:

1. The tracer picks random $k, r \in \mathbb{Z}_p$ then creates a probe ciphertext:

$$C_1 = g^{kd_1}, C_2 = e(g,g)^{kd_2 + r}, C_3 = g^{kd_1 d_2}, C_4 = e(g,g)^{kv},$$

$$C_5 = e(g,g)^{kvd_2}, C_6 = M'$$

2. Assume the decryption key $A_i, B_i$ is embedded in the pirate decoder. Then the tracer queries the pirate decoder on this probe ciphertext. The pirate decoder will compute:

$$K = \frac{e(A_i, C_1)}{C_4} \cdot C_2^{B_i - 1} \cdot (\frac{e(A_i, C_3)}{C_5})^{B_i} = e(g,g)^{\frac{kd_2 a_i}{a_i+1}} \cdot e(g,g)^{(kd_2+r)(\frac{1}{a_i+1} - \frac{1}{d_2} - 1)}$$

$$= e(g,g)^{-k} \cdot e(g,g)^{r(B_i - 1)}$$

Then outputs:

$$C_6/K$$

3. The tracer first recovers $K$ then computes $e(g,g)^{B_i}$ since it knows $k, r$. Then the tracer simply verifies if the element $e(g,g)^{B_i}$ is in the table $Tab$ and eventually outputs the traitor. It is easy to see that our tracing algorithm never accuses any innocent user and the time complexity of our tracing security is $O(\log N)$. We also notice that, in our system, $N$ is the effective number of the actual users in the system.

**Minimal Access Single-key Black Box Tracing Algorithm:** In the setup phase, the tracer picks random $k, r \in \mathbb{Z}_p$ and a message $M$, then creates:

$$C_1 = g^{kd_1}, C_2 = e(g,g)^{kd_2+r}, C_3 = g^{kd_1d_2}, C_4 = e(g,g)^{kv}, C_5 = e(g,g)^{kvd_2}$$

and store these values in a table $Tab$.
When a user $j$ joins the system, the tracer computes

$$C_{6,j} = e(g,g)^{-k} \cdot e(g,g)^{r(B_j-1)} \cdot M$$

and stores the pair $(j, C_{6,j})$ in the table $Tab$.
The tracing algorithm then works as follow:

1. For each user's indices $j$, the tracer queries the pirate decoder on a pair

$$(C = (C_1, C_2, C_3, C_4, C_5, C_{6,j}), M)$$

2. Assume the decryption key $A_i, B_i$ is embedded in the pirate decoder. The pirate decoder will compute:

$$K = \frac{e(A_i, C_1)}{C_4} \cdot C_2^{B_i-1} \cdot (\frac{e(A_i, C_3)}{C_5})^{B_i} = e(g,g)^{\frac{kd_2a_i}{a_i+1}} \cdot e(g,g)^{(kd_2+r)(\frac{1}{a_i+1} - \frac{1}{d_2} - 1)}$$

$$= e(g,g)^{-k} \cdot e(g,g)^{r(B_i-1)}$$

Then computes:

$$M' = C_{6,j}/K$$

3. At user's indices $j$, if the tracer receives a signal *valid* which indicates that $C$ is a valid encryption of $M$, then the tracer outputs user's indices $j$ is a traitor. It is easy to see that our tracing algorithm never accuses any innocent user and the time complexity of our tracing security is $O(N)$. We also notice that, in our system, $N$ is the effective number of the actual users in the system.

## 6 Conclusion

In this paper, we restrict ourselves to the non-black-box tracing and the single-key black box tracing models and proposed an optimal and practical scheme in these models. As far as we know, this is the first practical fully collusion resistant traitor tracing scheme. However the most important open problem in traitor tracing remains the construction of a practical fully collusion resistant traitor tracing scheme in the general black box tracing model. The schemes in [2,8] has constant ciphertext size but when considering the full collusion, the secret key size of user is $O(N^2)$ which is impractical. The most relevant schemes in [9] and in [10] still have large ciphertext size of $O(\sqrt{N})$ and require the use of bilinear maps in groups of composite order. We also recall that, non-black-box tracing and the single- key black box tracing models deal with pirates who are required to implement a key that has the form of the keys distributed to the users (this consideration is justified and discussed in the introduction) and do not consider pirates who can produce new form of key that can help to decrypt ciphertexts. One of the promising direction is to consider a model between the single-key black box tracing and the general black box tracing model in which one can still achieve a practical scheme.

## Acknowledgments

This work was partially supported by the French ANR-09-VERSO-016 BEST Project and partially conducted within the context of the International Associated Laboratory Formath Vietnam (LIAFV).

## References

1. S. Agrawal, Y. Dodis, V. Vaikuntanathan, and D. Wichs. On continual leakage of discrete log representations. Cryptology ePrint Archive, Report 2012/367, 2012. `http://eprint.iacr.org/2012/367`.
2. O. Billet and D. H. Phan. Efficient Traitor Tracing from Collusion Secure Codes. In R. Safavi-Naini, editor, *Information Theoretic Security—ICITS 2008*, volume 5155 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 2008.
3. O. Billet and D. H. Phan. Traitors collaborating in public: Pirates 2.0. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 189–205. Springer, Apr. 2009.
4. D. Boneh and X. Boyen. Short signatures without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, May 2004.
5. D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, May 2005.
6. D. Boneh and M. K. Franklin. An efficient public key traitor tracing scheme. In M. J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 338–353. Springer, Aug. 1999.
7. D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In V. Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 258–275. Springer, Aug. 2005.
8. D. Boneh and M. Naor. Traitor tracing with constant size ciphertext. In P. Ning, P. F. Syverson, and S. Jha, editors, *ACM CCS 08*, pages 501–510. ACM Press, Oct. 2008.
9. D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 573–592. Springer, May / June 2006.

10. D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06*, pages 211–220. ACM Press, Oct. / Nov. 2006.
11. X. Boyen. Mesh signatures. In M. Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 210–227. Springer, May 2007.
12. B. Chor, A. Fiat, and M. Naor. Tracing traitors. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 257–270. Springer, Aug. 1994.
13. C. Delerablée, P. Paillier, and D. Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, editors, *PAIRING 2007*, volume 4575 of *LNCS*, pages 39–59. Springer, July 2007.
14. P. Junod, A. Karlov, and A. K. Lenstra. Improving the Boneh-Franklin traitor tracing scheme. In S. Jarecki and G. Tsudik, editors, *PKC 2009*, volume 5443 of *LNCS*, pages 88–104. Springer, Mar. 2009.
15. A. Kiayias and M. Yung. Traitor tracing with constant transmission rate. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 450–465. Springer, Apr. / May 2002.
16. M. Naor and B. Pinkas. Efficient trace and revoke schemes. In Y. Frankel, editor, *FC 2000*, volume 1962 of *LNCS*, pages 1–20. Springer, Feb. 2000.
17. K. Nuida. A general conversion method of fingerprint codes to (more) robust fingerprint codes against bit erasure. In *Information Theoretic Security—ICITS 2009*, volume 5973 of *Lecture Notes in Computer Science*, pages 194–212. Springer, 2009.
18. D. H. Phan and V. C. Trinh. Optimal Public Key Traitor Tracing Scheme in Non-Black Box Model. In A. Youssef and A. Nitaj, editors, *Africacrypt 2013 Conference, Lecture Notes in Computer Science* Springer, June. 2013. Full version available at http://www.di.ens.fr/users/phan/2013-africa-a.pdf
19. D. Tonien and R. Safavi-Naini. An efficient single-key pirates tracing scheme using cover-free families. In J. Zhou, M. Yung, and F. Bao, editors, *ACNS 06*, volume 3989 of *LNCS*, pages 82–97. Springer, June 2006.