2004

# Linear code implies publicf-key traitor tracing with revocation

Joseph Tonien
*University of Wollongong*, dong@uow.edu.au

Rei Safavi-Naini
*University of Calgary*, rei@uow.edu.au

## Publication Details

To, V. & Safavi-Naini, R. (2004). Linear code implies publicf-key traitor tracing with revocation. 9th Australasian Conference on Information Security and Privacy (pp. 24-35). Germany: Springer Berlin Heidelberg.

# Linear code implies publicf-key traitor tracing with revocation

**Abstract**
In this paper, we show that the linear-coded Kurosawa–Desmedt scheme can be modified to allow revocation of users, that is to show a revocation scheme can be derived from a linear code.

**Keywords**
publicf, key, traitor, tracing, implies, revocation, linear, code



**Publication Details**
To, V. & Safavi-Naini, R. (2004). Linear code implies publicf-key traitor tracing with revocation. 9th Australasian Conference on Information Security and Privacy (pp. 24-35). Germany: Springer Berlin Heidelberg.

# Linear Code Implies Public-Key Traitor Tracing with *Revocation*

Vu Dong Tô and Reihaneh Safavi-Naini

School of Information Technology & Computer Science,
University of Wollongong, NSW 2522, Australia

**Abstract.** It was shown in [K. Kurosawa et al., Proc. PKC'02, LNCS 2274, pp. 172–187, 2002] that a public-key $(k, n)$-traitor tracing scheme, called linear-coded Kurosawa–Desmedt scheme, can be derived from an $[n, u, d]$-linear code such that $d \geq 2k + 1$. In this paper, we show that the linear-coded Kurosawa–Desmedt scheme can be modified to allow revocation of users, that is to show a revocation scheme can be derived from a linear code. The overhead of the modified scheme is very efficient: there is no extra user secret key storage, the public encryption key size remains the same, and the ciphertext size is of length $O(k)$. We prove the modified scheme is semantically secure against a passive adversary. Since the Boneh–Franklin scheme is proved to be equivalent to a slight modification of the corrected Kurosawa-Desmedt scheme, we show that we can also modify the Boneh–Franklin scheme to provide user revocation capability for this scheme. We also look at the problem of permanent removing a traitor in the Boneh-Franklin and prove some negative results.

## 1 Introduction

Digital content distribution is an important application of global networking. In such an application, data suppliers want their digital content to be available to authorized users only. The number of authorized users is large enough so that broadcasting data is much more efficient than establishing a secure channel between the data supplier and each individual authorized user.

In a public-key $(k, n)$-traitor tracing scheme, there are $n$ users, each holds a secret decryption key (or a decoder device). The encryption key is made public and the data supplier can use this public key to encrypt the digital content and broadcast the corresponding ciphertext. Authorized users, using their secret decryption keys, should be able to decrypt the broadcast messages. If a coalition of up to $k$ users collude to form a pirate decryption device, upon capturing this pirate device, the system uses tracing algorithm to identify at least one of the colluders.

Kurosawa–Desmedt [12] and Boneh–Franklin [2] proposed public key traitor tracing schemes based on the difficulty of decision Diffie–Hellman problem. To avoid linear attack [18,2], the Kurosawa–Desmedt scheme is modified to become the corrected Kurosawa–Desmedt scheme. It is an important property that the

corrected Kurosawa–Desmedt scheme can be generalized to use any linear code. It is shown that a public-key $(k, n)$-traitor tracing scheme, called linear-coded Kurosawa–Desmedt scheme [13], can be derived from an $[n, u, d]$-linear code such that $d \geq 2k+1$ and the Boneh–Franklin scheme is equivalent to the linear-coded Kurosawa–Desmedt scheme where Reed–Solomon code is used.

In both Kurosawa–Desmedt and Boneh–Franklin schemes, broadcast data can be decrypted by all legitimate users and it is not possible to target the data to a subgroup of users. Trace and revoke schemes [1,7,16,22,21,17,20,5,6,19,11] have the extra property that users can be revoked and so the broadcast targeted to a subgroup of users. In schemes [1,7,16,22,21] the encryption key is secret so it only supports one data provider. The scheme [5] by Dodis et al is the first trace and revoke scheme that has CCA2 security. The scheme [11] by Kim et al is a modification of Dodis scheme, it has CCA2 security and the ciphertext size is half that of Dodis's scheme. Most of public-key revocation schemes are polynomial based that make use of the Shamir's secret sharing technique, in which there is a correspondence between the secret key held by each user with a value of the polynomial at a specific point; and when a user is revoked, the information of the polynomial value corresponds to the revoked user is broadcasted in the ciphertext.

In this paper, we show that the linear-coded Kurosawa–Desmedt and the Boneh–Franklin schemes can be modified to have user revocation capability. It is interesting that in modifying these schemes we do not introduce any more complexity in the key generation process. Users will keep exactly the same secret keys as the original schemes. The public encryption key in the linear-coded Kurosawa–Desmedt remains the same as it is in the modified scheme. For the Boneh–Franklin scheme, only one more group element is added into the public encryption key. The ciphertext size is as efficient as other revoke schemes [17, 20,11,19,6]. For a tracing threshold $k$, our modified schemes can afford up to $2k-1$ user revocation and the ciphertext consists of $2k$ field elements and $2k+2$ group elements. We also prove the semantic security for our proposed revocation schemes.

Dodis et al. [6] introduced the notion of *scalable* system. A broadcast system is *server-side scalable* if any party can broadcast messages, this can be accomplished by using public key approach. A broadcast system is *client-side scalable* if it supports increasing number *add-user* and *remove-user* operations. Our proposed revocation schemes are server-side scalable but not client-side scalable since they cannot remove more than $2k$ users. The only known scalable scheme is the Dodis et al. scheme [6]. In this paper, we look at the possibility of repeatedly permanent-removing users by modifying only the public key and public parts of users keys. If it is possible to do so then we would have a client-side scalable scheme. Unfortunately, we prove that it is not possible, at least, for the Boneh–Franklin scheme. We consider the first case when we remove a user by modifying only the public encryption key so that the decryption key of the removed user become invalid in the new encryption-decryption system, however, the remained non-removed users should have their decryption keys remain valid.

We show that it is impossible to do so. The second case we consider is to remove a user by modifying the public encryption key and the public parts of users keys (which are the rows of the public matrix). We show that it is also impossible to do so in this case.

The rest of the paper is organized as follows. In section 2, we give a model of traitor tracing scheme. The Boneh–Franklin scheme, the corrected Kurosawa–Desmedt and the linear-coded Kurosawa–Desmedt schemes are reviewed in section 3. In section 4, we present revocation functionality for these three schemes. In section 5, we look at the problem of permanently removing a user in the Boneh–Franklin scheme and prove some negative results.

## 2 Preliminaries

An $[n, u, d]$-linear code is a linear code contains $n$ codewords, dimension $u$ and the minimum Hamming distance $d$. The parity check matrix for an $[n, u, d]$-linear code is a matrix of size $(n - u) \times n$ and any $d - 1$ columns vectors are linear independent. The notation $\cdot$ denotes the inner product of two vectors.

Let $q > n$ be a prime number. Let $G_q$ be a group of prime order $q$. The Decision Diffie–Hellman problem in $G_q$ is to determine whether $w = uv$ given $g$, $g^u$, $g^v$, $g^w$ where $g$ is chosen random from $G_q$ and $u, v, w$ are chosen random from $Z_q$.

### 2.1 Model of Traitor Tracing

A $(k, n)$-traitor tracing scheme with revocation has four components.
**Key Generation:** given a security input $1^\ell$, the key generation procedure outputs an encryption key $PK$ and $n$ user decryption keys $SK_1, \ldots, SK_n$. The encryption key $PK$ is made public so any data supplier can use it to broadcast data. Decryption key $SK_i$ is given to the user $i$ to keep secretly.
**Encryption:** taken as input a message $M$, the encryption key $PK$ and a revoked set of users $R$, the encryption procedure $\mathcal{E}$ outputs the corresponding ciphertext $C = \mathcal{E}_{PK}(R, M)$. If the scheme do not support revocation (for instance, the linear-coded Kurosawa–Desmedt scheme and the Boneh–Franklin scheme) then $R$ is always an empty set.
**Decryption:** taken as input a ciphertext $C = \mathcal{E}_{PK}(R, M)$ and a decryption key $SK_i$, the decryption procedure $\mathcal{D}$ outputs the message $M$ if $i \notin R$.
**Traitor Tracing:** if up to $k$ users collude to form a pirate decryption box then upon capturing this pirate device, the traitor tracing procedure can identify at least one of the colluders. It is assumed that the pirate decryption box is resettable to the initial state.
There are two types of tracing: *open-box* tracing and *black-box* tracing. In *open-box* tracing, it is assumed that the pirate box can be opened and the pirate keys inside the box can be obtained. In *black-box* tracing, the tracing algorithm cannot open the decoder box and access the stored keys. However it can make *queries* and see the responses. That is, it can send encrypted contents to the box and see the outputs of the box.

## 3 Previous Public-Key Traitor Tracing Schemes

In this section, we look at three public-key traitor tracing schemes: the Boneh–Franklin (BF) scheme [2], the linear-coded Kurosawa–Desmedt (LC-KD') scheme and the corrected Kurosawa–Desmedt (corrected KD) scheme [13].

### 3.1 Boneh–Franklin Scheme

Below is the description of the Boneh–Franklin traitor tracing scheme for $n$ users and collusion threshold $k$.

**Key Generation:** Let $G_q$ be a group of order prime $q$ and $g$ be a group generator. It is assumed that the Decision Diffie-Hellman problem in $G_q$ is hard.

Let $A$ be the following $(n-2k) \times n$ matrix

$$A = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ 1^3 & 2^3 & 3^3 & \dots & n^3 \\ \vdots & & & & \vdots \\ 1^{n-2k-1} & 2^{n-2k-1} & 3^{n-2k-1} & \dots & n^{n-2k-1} \end{pmatrix} .$$

Since $A$ has full rank, the equation $Ax = 0$ has a nullspace of dimension $2k$. Let $\Gamma$ be an $n \times 2k$ matrix whose columns are $2k$ independent solutions $x_1, \dots, x_{2k}$ of $Ax = 0$. Let $\boldsymbol{\gamma}^{(1)}, \boldsymbol{\gamma}^{(2)}, \dots, \boldsymbol{\gamma}^{(n)}$ denote the $n$ row vectors of $\Gamma$, each of length $2k$. The matrix $\Gamma$ is made public.

Choose random $\boldsymbol{b} = (b_1, \dots, b_{2k}) \in \mathbf{Z}_q^{2k}$. Let $\Gamma \cdot \boldsymbol{b} = \boldsymbol{e} = (e_1, \dots, e_n)$. Let $h_1 = g^{b_1}, \dots, h_{2k} = g^{b_{2k}}$. Choose random $r_1, \dots, r_{2k}$ in $\mathbf{Z}_q$ and let $y = h_1^{r_1} \dots h_{2k}^{r_{2k}}$. We have $y = g^a$ with $a = b_1 r_1 + \dots + b_{2k} r_{2k}$. (It is commented in [13] that, it is redundant to store system secret values $r_1, \dots, r_{2k}$. Instead, we can just choose a random $a$ and let $y = g^a$.)

**Public encryption key** is $PK = (y, h_1, \dots, h_{2k})$.

Note that the matrix $\Gamma$ is made public.

**User secret decryption keys.** For each $1 \le i \le n$, let $v_i = a/e_i$. Decryption key for user $i$ is the vector $\boldsymbol{\theta}^{(i)} = v_i \boldsymbol{\gamma}^{(i)}$. This decryption key can be thought of as two-part key. The first part is the row vector $\boldsymbol{\gamma}^{(i)}$ of the public matrix $\Gamma$. User $i$ only needs to keep the second part $v_i$ secret.

*Discrete Log (DL) Representation.* A vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_{2k}) \in \mathbf{Z}_q^{2k}$ satisfying

$$y = h_1^{\theta_1} h_2^{\theta_2} \dots h_{2k}^{\theta_{2k}} \tag{1}$$

is called a DL-representation of the DL-element $y$ with respect to the DL-base $h_1, \dots, h_{2k}$. The condition (1) is equivalent to

$$a = \boldsymbol{\theta} \cdot \boldsymbol{b} = \theta_1 b_1 + \dots + \theta_{2k} b_{2k} \ . \tag{2}$$

We note that the decryption key for user $i$, $\boldsymbol{\theta}^{(i)} = v_i \boldsymbol{\gamma}^{(i)} \in \mathbf{Z}_q^{2k}$, is a scalar multiple of the $i^{\text{th}}$ row vector $\boldsymbol{\gamma}^{(i)}$ of the matrix $\Gamma$. It is also a DL-representation of $y$ with respect to $h_1, \dots, h_{2k}$ since, $\boldsymbol{\theta}^{(i)} \cdot \boldsymbol{b} = v_i (\boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{b}) = v_i e_i = a$.

**Encryption:** a message $M \in G_q$ is encrypted as

$$\langle My^r, h_1^r, h_2^r, \ldots, h_{2k}^r \rangle$$

where $r$ is randomly chosen in $\mathbf{Z}_q$.
**Decryption:** Any DL-representation $\boldsymbol{\theta} \in \mathbf{Z}_q^{2k}$ of $y$ with respect to $h_1, \ldots, h_{2k}$ can be used to decrypt

$$M = \frac{My^r}{(h_1^r)^{\theta_1}(h_2^r)^{\theta_2} \ldots (h_{2k}^r)^{\theta_{2k}}} \ .$$

Decryption key of each user is a DL-representation so they can use it to decrypt the ciphertext.
**Traitor tracing:** a collusion of $c$ users can generate a pirate key from their $c$ keys $\boldsymbol{\theta}^{(u_1)}, \ldots, \boldsymbol{\theta}^{(u_c)}$ as follows

$$\boldsymbol{\theta}_{pirate} = \mu_1 \boldsymbol{\theta}^{(u_1)} + \ldots + \mu_c \boldsymbol{\theta}^{(u_c)}, \quad \text{where } \mu_1 + \ldots + \mu_c = 1 \ .$$

The pirate key $\boldsymbol{\theta}_{pirate}$ is called a convex combination of the colluders' keys $\boldsymbol{\theta}^{(u_1)}, \ldots, \boldsymbol{\theta}^{(u_c)}$. It is easy to verify that $\boldsymbol{\theta}_{pirate}$ is a DL-representation of $y$ with respect to $h_1, \ldots, h_{2k}$. Since $\boldsymbol{\theta}^{(u_i)}$ is a scalar multiple of $\boldsymbol{\gamma}^{(u_i)}$, the pirate key $\boldsymbol{\theta}_{pirate}$ is a linear combination of $\boldsymbol{\gamma}^{(u_1)}, \ldots, \boldsymbol{\gamma}^{(u_c)}$. The BF tracing algorithm bases on this fact. It uses the Berlekamp's algorithm to identify *all* of the colluders $u_1, \ldots, u_c$. In Berlekamp's algorithm, given a linear combination of $\boldsymbol{\gamma}^{(u_1)}, \ldots, \boldsymbol{\gamma}^{(u_c)}$ as input, it outputs all the indices $u_1, \ldots, u_c$.

### 3.2 Linear-Coded Kurosawa–Desmedt (LC-KD') Scheme

Below is the description of the LC-KD' scheme for $n$ users and collusion threshold $k$.
**Key Generation:** Let $G_q$ be a group of order prime $q$ and $g$ be a group generator. It is assumed that the Decision Diffie-Hellman problem in $G_q$ is hard.

Let $\mathcal{C}$ be an $[n, u, d]$-linear code over $Z_q$ whose distance $d \geq 2k + 1$. Let $m = n - u$, we have $m \geq d - 1 \geq 2k$. (In the corrected KD scheme, it is chosen that $m = d - 1 = 2k$.) Let $H$ be the parity check matrix for $\mathcal{C}$. Let $\Gamma = H^T$ then $\Gamma$ is a matrix of size $n \times m$. Let $\boldsymbol{\gamma}^{(1)}, \boldsymbol{\gamma}^{(2)}, \ldots, \boldsymbol{\gamma}^{(n)}$ denote the $n$ row vectors of $\Gamma$, each of length $m \geq 2k$. Any $(d-1)$ rows of $\Gamma$ are linear independent.

Choose random $\boldsymbol{b} = (b_1, \ldots, b_m) \in \mathbf{Z}_q^m$ such that $\boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{b} \neq 0$ for $i = 1, \ldots, n$. Let $h_1 = g^{b_1}, \ldots, h_m = g^{b_m}$, and $\Gamma \cdot \boldsymbol{b} = \boldsymbol{e} = (e_1, \ldots, e_n)$. Then $e_i = \boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{b} \neq 0$.
**Public encryption key** is $PK = (g, h_1, \ldots, h_m)$.
Note that the matrix $\Gamma$ is made public.
**User secret decryption keys.** The decryption key for user $i$ is $e_i$.
**Encryption:** a message $M \in G_q$ is encrypted as

$$\langle Mg^r, h_1^r, h_2^r, \ldots, h_m^r \rangle$$

where $r$ is randomly chosen in $\mathbf{Z}_q$.

**Decryption:** Each user $i$ uses the $i^{\text{th}}$ row of $\Gamma$, $\boldsymbol{\gamma}^{(i)}$, and his secret $e_i$ to decrypt as follows

$$M = \frac{M\, g^r}{\left[(h_1^r)^{\gamma_1^{(i)}} \ldots (h_m^r)^{\gamma_m^{(i)}}\right]^{1/e_i}} \ .$$

### 3.3 The Corrected Kurosawa–Desmedt Scheme

The corrected KD scheme chooses an $[n, n-2k, 2k+1]$-Reed Solomon code. The matrix $\Gamma$ in this case has size $n \times 2k$ and the $i^{\text{th}}$ row vector of $\Gamma$ is $\boldsymbol{\gamma}^{(i)} = (1, i, i^2, \ldots, i^{2k-1})$. Let $f(x) = b_1 + b_2 x + \ldots + b_{2k} x^{2k-1}$ then in the matrix equation $\Gamma \cdot \boldsymbol{b} = \boldsymbol{e} = (e_1, \ldots, e_n)$ we have $e_i = \boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{b} = f(i)$. Thus, the secret key for user $i$ is the polynomial value $e_i = f(i)$.
**Encryption:** a message $M \in G_q$ is encrypted as

$$\langle g^r, M\, h_1^r, h_2^r, \ldots, h_{2k}^r \rangle$$

where $r$ is randomly chosen in $\mathbf{Z}_q$.
**Decryption:** Each user $i$ uses the secret key $e_i = f(i)$ to decrypt as follows

$$M = \frac{(M\, h_1^r)(h_2^r)^i (h_3^r)^{i^2} \ldots (h_{2k}^r)^{i^{2k-1}}}{(g^r)^{f(i)}} \ .$$

## 4 Modified Schemes with Revocation

In this section, we show that revocation schemes can be derived from linear codes. We propose revocation technique for the three schemes: the linear-coded Kurosawa–Desmedt scheme, the corrected Kurosawa–Desmedt scheme and the Boneh–Franklin scheme. The advantage of the proposed schemes is that no user secret keys needed to change. There is no changes in public encryption keys, except in the BF scheme, a single group element is added to the public key. The security is provable (semantic security against passive adversary). The proposed revocation schemes are threshold schemes, up to $2k-1$ users can be revoked where $k$ denotes the collusion threshold. Broadcast ciphertexts contain $2k$ field elements and $2k+2$ group elements which is as efficient as other revocation schemes such as Naor–Pinkas [17], Tzeng–Tzeng [20], Tô et al [19], and Kim et al [11].

### 4.1 LC-KD' with Revocation

**Revocation:** Let $R$ be a subset of $\{1, \ldots, n\}$ such that $1 \leq |R| < 2k$. $R$ represents the set of revoked users. Choose $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)$, such that in the equation $\Gamma \cdot \boldsymbol{\beta} = \boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$, we have $\epsilon_i = 0$ if and only if $i \in R$. This can be done because any $2k$ rows of $\Gamma$ are linear independent.

Let $\eta_1 = g^{\beta_1}, \ldots, \eta_m = g^{\beta_m}$.

A message $M \in G_q$ is encrypted as

$$\langle\, g^{r_1}, M\, g^{r_2}, \beta_1, \ldots, \beta_m, h_1^{r_1}\eta_1^{r_2}, \ldots, h_m^{r_1}\eta_m^{r_2}\,\rangle$$

where $r_1$, $r_2$ are random numbers in $Z_q$.
**Decryption.** User $i$ first calculates $\epsilon_i = \boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{\beta}$. If $\epsilon_i = 0$ then $i$ is revoked. Otherwise, $i \notin R$, and user $i$ can use secret value $e_i$ and the vector $\boldsymbol{\gamma}^{(i)}$ of the public matrix $\Gamma$ to decrypt

$$(M\, g_1^{r_2}) \left( \frac{(g^{r_1})^{e_i}}{(h_1^{r_1}\eta_1^{r_2})^{\gamma_1^{(i)}} \ldots (h_{2k}^{r_1}\eta_{2k}^{r_2})^{\gamma_{2k}^{(i)}}} \right)^{1/\epsilon_i} = M \ .$$

### 4.2 The Corrected Kurosawa–Desmedt Scheme

If $R = \{i_1, \ldots, i_c\}$, $1 \le c < 2k$, is the revoked user set then in the revocation procedure we need to find a vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{2k})$ such that in the equation $\Gamma \cdot \boldsymbol{\beta} = \boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ we have $\epsilon_i = 0$ if and only if $i \in R$. Consider the polynomial $g(x) = \beta_1 + \beta_2\, x + \ldots + \beta_{2k}\, x^{2k-1}$ formed by the vector $\boldsymbol{\beta}$. We have $\epsilon_i = \boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{\beta} = g(i)$. Thus $g(i) = 0$ if and only if $i \in R$. That means $g(x)$ can be written as $g(x) = (x - i_1) \ldots (x - i_c) z(x)$ where $z$ is a polynomial of degree up to $2k - c$ whose roots are not in the set $U = \{1, \ldots, n\}$. In particular, if the number of revoked user is $2k$ then $z(x)$ is a non-zero number in $\mathbf{Z}_q$.

In summary, for the corrected KD scheme, the revocation procedure is as follows.
**Revocation.** Let $R = \{i_1, \ldots, i_c\}$, $1 \le c < 2k$, be the revoked user set. Choose a random polynomial $z(x)$ of degree up to $2k - c$ such that $z(i) \neq 0$ for all $i = 1, \ldots, n$. Let $g(x) = (x - i_1) \ldots (x - i_c) z(x) = \beta_1 + \beta_2\, x + \ldots + \beta_{2k}\, x^{2k-1}$. Let $\eta_1 = g^{\beta_1}$, $\ldots$, $\eta_{2k} = g^{\beta_{2k}}$.

A message $M \in G_q$ is encrypted as

$$\langle\, g^{r_1}, M\, g^{r_2}, \beta_1, \ldots, \beta_{2k}, h_1^{r_1}\eta_1^{r_2}, \ldots, h_{2k}^{r_1}\eta_{2k}^{r_2}\,\rangle$$

where $r_1$, $r_2$ are random numbers in $Z_q$.
**Decryption.** User $i$ first calculates $\epsilon_i = g(i) = \beta_1 + \beta_2\, i + \ldots + \beta_{2k}\, i^{2k-1}$. If $g(i) = 0$ then $i$ is revoked. For $i \notin R$, $\epsilon_i = g(i) = (i - i_1) \ldots (i - i_c) z(i) \neq 0$. User $i$ then uses secret value $e_i = f(i)$ to decrypt

$$(M\, g_1^{r_2}) \left( \frac{(g^{r_1})^{f(i)}}{(h_1^{r_1}\eta_1^{r_2})(h_2^{r_1}\eta_2^{r_2})^{i}(h_3^{r_1}\eta_3^{r_2})^{i^2} \ldots (h_{2k}^{r_1}\eta_{2k}^{r_2})^{i^{2k-1}}} \right)^{1/g(i)} = M \ .$$

### 4.3 BF with Revocation

For the BF scheme, the public encryption key is slightly changed. A single group element is added to the encryption key.
**The new encryption key** is $PK' = (g_1, y, h_1, \ldots, h_{2k})$.
The added element $g_1$ is an arbitrary generator of $G_q$, indeed, we can choose $g_1 = g$.

**Revocation.** Let $R$ be a subset of $\{1, \ldots, n\}$ such that $1 \leq |R| < 2k$. $R$ represents the set of revoked users. Choose $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{2k})$, such that in the equation $\Gamma \cdot \boldsymbol{\beta} = \boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$, we have $\epsilon_i = 0$ if and only if $i \in R$. This can be done because any $2k$ rows of $\Gamma$ are linear independent.

Let $\eta_1 = g_1^{\beta_1}, \ldots, \eta_{2k} = g_1^{\beta_{2k}}$.

A message $M \in G_q$ is encrypted as

$$\langle\, y^{r_1}, M\, g_1^{r_2}, \beta_1, \ldots, \beta_{2k}, h_1^{r_1}\eta_1^{r_2}, \ldots, h_{2k}^{r_1}\eta_{2k}^{r_2} \,\rangle$$

where $r_1$, $r_2$ are random numbers in $\mathbf{Z}_q$.

**Decryption.** User $i$ first calculates $\epsilon_i = \boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{\beta}$. If $\epsilon_i = 0$ then $i$ is revoked. Otherwise, $i \notin R$, user $i$ can use his decryption key $\boldsymbol{\theta}^{(i)} = v_i\, \boldsymbol{\gamma}^{(i)}$ to decrypt

$$(M\, g_1^{r_2}) \left( \frac{y^{r_1}}{(h_1^{r_1}\eta_1^{r_2})^{\theta_1^{(i)}} \ldots (h_{2k}^{r_1}\eta_{2k}^{r_2})^{\theta_{2k}^{(i)}}} \right)^{1/(v_i \epsilon_i)} = M \ .$$

### 4.4 Semantic Security for Revocation

We show that the proposed revocation schemes are semantically secure against a passive adversary who controls up to $2k-1$ users assuming the difficulty of the DDH problem in the group $G_q$. We give a security proof for the linear-coded Kurosawa–Desmedt (LC-KD') scheme. The proof can be easily adjusted for other schemes.

**Model of Adversary.** The following game models an *Adversary* $\mathcal{A}$ who controls up to $2k-1$ users and an *Oracle* who represents the revocation scheme.

1. Adversary adaptively chooses a set $\mathcal{A}_{\text{users}}$ of up to $2k-1$ users that it controls.
2. Given $\mathcal{A}_{\text{users}}$, for a given security parameter $\lambda$, the Oracle runs the key generation procedure and gives the Adversary the public encryption key together with all secret keys of the users in $\mathcal{A}_{\text{users}}$ under the control of the Adversary.
3. The Adversary then produces two challenge messages $M_0$ and $M_1$ and gives them to the Oracle.
4. The Oracle selects a random bit $r \in \{0, 1\}$ and gives the Adversary back the ciphertext of $M_r$ encrypted with the revoked set $R = \mathcal{A}_{\text{users}}$.
5. The Adversary output a bit $r'$.

The advantage of the adversary $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}(\lambda) = |Pr(r = r') - 1/2|$. We say that the revocation scheme is semantically secure if $Adv_{\mathcal{A}}(\lambda)$ is negligible.

Theorem 1 states that the linear-coded Kurosawa–Desmedt revocation scheme is semantically secure, the proof is given in the full version of the paper.

**Theorem 1.** *The LC-KD' revocation scheme is semantically secure against a collusion of up to $2k-1$ revoked users assuming the difficulty of the DDH problem.*

## 5 Permanent User Removal

In a revocation scheme, we can remove a traitor permanently by always including the traitor in the revoked user set. However, in a threshold revocation scheme such as our proposed schemes, the number of revoked users is limited so we cannot use it to remove many traitors. Dodis et al. [6] define a scheme to be *scalable* if any party can broadcast messages using public key (*server-side scalable*) and if it supports increasing number of *add-user* and *remove-user* operations (*client-side scalable*).

Our proposed revocation schemes are server-side scalable but not client-side scalable since they cannot remove more than $2k - 1$ users. The only known scalable scheme is the Dodis et al. scheme [6]. In this scheme, everytime after removing $v$ users, it allows legitimate users update their secret keys. So the time line is divided into many "windows", and in each window, $v$ users are removed. To calculate new secret key, legitimate users need to use their old secret key together with a single update information broadcasted by the system administrator. The only problem with this scheme is, in each key update time, the same update information is used for all users. Even a revoked user, if by any chance he has this update information, he can use it to update his key to a valid key in the new session. Therefore, as emphasized in their paper, the Dodis et al. scheme is only secure against *window adversary*. That is, it is secure against up to a threshold of $v$ revoked users who are *subsequently revoked in the same window*. This make the scheme vulnerable under the collusion of as small as two revoked users who are revoked in two different windows. It remains as an open problem to design a scalable scheme that is secure against a collusion of a threshold number of arbitrary revoked users.

In this section, we look at the possibility of repeatedly removing permanently users by modifying only the public key and public parts of users keys. If it is possible to do so then we would have a scalable scheme. Unfortunately, we prove that it is not possible, at least, for the BF scheme. Section 5.1 considers the case when we remove a user by modifying only the public encryption key. Section 5.2 considers the case when we remove a user by modifying the public encryption key and the public parts of users keys (which are the rows of the public matrix).

### 5.1 Modifying Public Key, Keeping User Keys Unchanged

Consider the Boneh–Franklin scheme. Let $y = g^a$, $h_1 = g^{b_1}$, $h_2 = g^{b_2}$, ..., $h_{2k} = g^{b_{2k}}$ be the current public encryption key. The user decryption key is $\boldsymbol{\theta}^{(i)} = v_i\, \boldsymbol{\gamma}^{(i)}$.

For simplicity, assume now we want to remove user $n$. We want to change the public encryption key become $y' = g^{a'}$, $h'_1 = g^{b'_1}$, $h'_2 = g^{b'_2}$, ..., $h'_{2k} = g^{b'_{2k}}$.

For each $i = 1, \ldots, n-1$, in order to have user $i$ remained valid, the decryption key $\boldsymbol{\theta}^{(i)} = v_i\, \boldsymbol{\gamma}^{(i)}$ must be a DL-representation of the new DL-element $y'$ with respect to the new DL-base $h'_1$, ..., $h'_{2k}$. Therefore,

$$a' = \boldsymbol{\theta}^{(i)} \cdot \boldsymbol{b'} = v_i(\boldsymbol{\gamma}^{(i)} \cdot \boldsymbol{b'}), \quad \forall i = 1, \ldots, n-1 \ .$$

We want the decryption key of the removed user $n$, $\boldsymbol{\theta}^{(n)} = v_n \boldsymbol{\gamma}^{(n)}$, to become invalid. That is, the vector $\boldsymbol{\theta}^{(n)}$ is not a DL-representation of the new DL-element $y'$ with respect to the new DL-base $h'_1, h'_2, \ldots, h'_{2k}$:

$$\boldsymbol{\theta}^{(n)} \cdot \boldsymbol{b'} = v_n(\boldsymbol{\gamma}^{(n)} \cdot \boldsymbol{b'}) \neq a' \ .$$

We will prove that this cannot be achieved.

**Theorem 2.** *Let $A$ be a full rank matrix of size $(n - 2k) \times n$ and let $\Gamma$ be an $n \times 2k$ matrix whose columns are $2k$ linear independent solutions of the equation $Ax = 0$. Let $\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(n)}$ denote the $n$ row vectors of $\Gamma$.*

*If for some vectors $\boldsymbol{b}$, $\boldsymbol{b'}$ of length $2k$, and some non-zero numbers $a$, $a'$, $v_1$, $v_2, \ldots, v_n$ we have $v_1(\boldsymbol{\gamma}^{(1)} \cdot \boldsymbol{b}) = v_2(\boldsymbol{\gamma}^{(2)} \cdot \boldsymbol{b}) = \ldots = v_n(\boldsymbol{\gamma}^{(n)} \cdot \boldsymbol{b}) = a$ and $v_1(\boldsymbol{\gamma}^{(1)} \cdot \boldsymbol{b'}) = v_2(\boldsymbol{\gamma}^{(2)} \cdot \boldsymbol{b'}) = \ldots = v_{n-1}(\boldsymbol{\gamma}^{(n-1)} \cdot \boldsymbol{b'}) = a'$, then it holds that $v_n(\boldsymbol{\gamma}^{(n)} \cdot \boldsymbol{b'}) = a'$.*

From Theorem 2, we can see that the secret key of user $n$, $\boldsymbol{\theta}^{(n)}$, remains valid for the new decryption with the new public encryption key.

We have proved in this section that it is impossible to remove a user by changing only the public encryption key and keeping all user secret keys fixed. In the next section, we will see a method by Silja Mäki to remove an user by changing public encryption key together with the matrix $\Gamma$.

### 5.2 Silja Mäki's Attempt: Changing Public Key Together with Public Parts of User Keys

In BF scheme, the user decryption key $\boldsymbol{\theta}^{(i)} = v_i \boldsymbol{\gamma}^{(i)}$ can be viewed as two parts. The secret part $v_i$ is kept by the user and the public part $\boldsymbol{\gamma}^{(i)}$ is kept by the system. Mäki's [14] idea is to keep the secret part $v_i$ unchanged while the public part of the key, $\boldsymbol{\gamma}^{(i)}$, is changing whenever the system wants to remove a colluder.

Below is a summary of Mäki's modification to BF

1. Replace $\boldsymbol{b} = (b_1, b_2, \ldots, b_{2k})$ with $\boldsymbol{b'} = (b'_1, b'_2, \ldots, b'_{2k})$, that is, replace $h_1 = g^{b_1}, \ldots, h_{2k} = g^{b_{2k}}$ by $h'_1 = g^{b'_1}, \ldots, h'_{2k} = g^{b'_{2k}}$.
2. Replace the matrix $\Gamma$ with the new matrix $\Gamma'$ whose $2k$ columns vectors also are $2k$ linear independent solutions of the equation $Ax = 0$. Denote the $n$ row vectors of $\Gamma'$ by $\boldsymbol{\gamma}^{(1)\prime}, \boldsymbol{\gamma}^{(2)\prime}, \ldots, \boldsymbol{\gamma}^{(n)\prime}$. Matrix $\Gamma'$ is chosen so that the vector $\boldsymbol{\theta}^{(i)\prime} = v_i \boldsymbol{\gamma}^{(i)\prime}$ is a DL-representation of $y$ with respect to the new DL-base $h'_1, h'_2, \ldots, h'_{2k}$. In the other words, the row vectors $\boldsymbol{\gamma}^{(i)\prime}$ must satisfy $\boldsymbol{\theta}^{(i)\prime} \cdot \boldsymbol{b'} = v_i(\boldsymbol{\gamma}^{(i)\prime} \cdot \boldsymbol{b'}) = a$.
3. Make the first $n - 1$ row vectors $\boldsymbol{\gamma}^{(1)\prime}, \boldsymbol{\gamma}^{(2)\prime}, \ldots, \boldsymbol{\gamma}^{(n-1)\prime}$ of the matrix $\Gamma'$ public so that each user $i$, $1 \leq i \leq n - 1$, can use the corresponding new vector with his secret key $v_i$ to form the new decryption key $\boldsymbol{\theta}^{(i)\prime} = v_i \boldsymbol{\gamma}^{(i)\prime}$ for the future decryption.
4. The last row vector $\boldsymbol{\gamma}^{(n)\prime}$ of the matrix $\Gamma'$ is kept secret so that the removed user $n$ cannot form the decryption key.
5. Since $A\Gamma' = 0$, the tracing remains the same.

However, Mäki also presented a successful attack, in which the removed user $n$ by looking at the matrix $\Gamma$ and the first $n-1$ rows of matrix $\Gamma'$ can calculate the last row $\boldsymbol{\gamma}^{(n)\prime}$ of $\Gamma'$. Therefore, the removed user can obtain the new decryption key for himself.

We can eliminate this attack by choosing the matrix $\Gamma'$ so that even the user $n$ can calculate the last row vector $\boldsymbol{\gamma}^{(n)\prime}$ but he cannot use it to form the decryption key. That is $\boldsymbol{\gamma}^{(n)\prime}$ must be chosen so that the vector $\boldsymbol{\theta}^{(n)\prime} = v_n\,\boldsymbol{\gamma}^{(n)\prime}$ is not a DL-representation of $y$ with respect to the new DL-base $h'_1, h'_2, \ldots, h'_{2k}$. Or equivalently, $\boldsymbol{\theta}^{(n)\prime} \cdot \boldsymbol{b'} = v_i(\cdot\boldsymbol{\gamma}^{(n)\prime} \cdot \boldsymbol{b'}) \neq a$. However, we will prove that this cannot be done.

**Theorem 3.** *Let $A$ be a full rank matrix of size $(n-2k)\times n$ and let $\Gamma$, $\Gamma'$ be two $n\times 2k$ matrices both of whose $2k$ columns are linear independent solutions of the equation $Ax = 0$. Let $\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(n)}$ and $\boldsymbol{\gamma}^{(1)\prime}, \ldots, \boldsymbol{\gamma}^{(n)\prime}$ respectively denote row vectors of $\Gamma$ and $\Gamma'$.*

*If for some vectors $\boldsymbol{b}$, $\boldsymbol{b'}$ of length $2k$, and some non-zero numbers $a$, $a'$, $v_1$, $v_2, \ldots, v_n$ we have $v_1(\boldsymbol{\gamma}^{(1)} \cdot \boldsymbol{b}) = v_2(\boldsymbol{\gamma}^{(2)} \cdot \boldsymbol{b}) = \ldots = v_n(\boldsymbol{\gamma}^{(n)} \cdot \boldsymbol{b}) = a$ and $v_1(\boldsymbol{\gamma}^{(1)\prime} \cdot \boldsymbol{b'}) = v_2(\boldsymbol{\gamma}^{(2)\prime} \cdot \boldsymbol{b'}) = \ldots = v_{n-1}(\boldsymbol{\gamma}^{(n-1)\prime} \cdot \boldsymbol{b'}) = a'$, then it holds that $v_n(\boldsymbol{\gamma}^{(n)\prime} \cdot \boldsymbol{b'}) = a'$.*

From Theorem 3, if for all $i$, $1 \leq i \leq n-1$, user $i$ can combine the new vector $\boldsymbol{\gamma}^{(i)\prime}$ with $v_i$ to get the new decryption key $\boldsymbol{\theta}^{(i)\prime} = v_i\,\boldsymbol{\gamma}^{(i)\prime}$ then the last vector $\boldsymbol{\gamma}^{(n)\prime}$ also makes a valid decryption key $\boldsymbol{\theta}^{(n)\prime} = v_n\,\boldsymbol{\gamma}^{(n)\prime}$. Since $A\Gamma' = 0$ and $n-1$ row vectors of $\Gamma'$ is publicly known, the removed user $n$ can calculate the last row $\boldsymbol{\gamma}^{(n)\prime}$ and hence obtain the valid decryption key $\boldsymbol{\theta}^{(n)\prime}$.

## 6 Conclusion

In this paper, we have shown that from an $[n, u, d]$-linear code such that $d \geq 2k+1$, it is possible to construct a public-key $(k, n)$-traitor tracing scheme with *revocation*. We demonstrate this technique for three schemes: the Boneh–Franklin scheme, the corrected Kurosawa–Desmedt scheme and the linear-coded Kurosawa–Desmedt scheme. The security of our proposed revocation schemes is provable (semantic security against passive adversary). It seems that CCA2 security can be also achieved if we use Cramer and Shoup approach [4] to modify our schemes.

We also look at the problem of permanently removing a user to make the system scalable. We prove that it is impossible to obtain a *remove-user* procedure for the Boneh–Franklin scheme where all the secret part of user keys are kept unchanged and only the public encryption key and the public part of user keys are allowed to modified. Dodis et al [6] is the only known scalable scheme, however it is only secure against "window adversary", which is not a strong model. It remains as an open problem to design a public key traitor tracing with revocation which is fully scalable.

## References


1. T. Asano, *A revocation scheme with minimal storage at receivers,* AsiaCrypt'02, LNCS 2501, 433–450, 2002.
2. D. Boneh and M. Franklin, *An Efficient Public Key Traitor Tracing Scheme,* Crypto'99, LNCS 1666, 338–353, 1999.
3. D. Boneh and J. Shaw, *Collusion-Secure Fingerprinting for Digital Data,* IEEE Transactions on Information Theory, **44** (1998), 1897–1905.
4. R. Cramer and V. Shoup, *A Practical Public Key Cryptosystem Provable Secure Against Adaptive Chosen Ciphertext Attack,* Crypto'98, LNCS 1462, 13–25, 1998.
5. Y. Dodis and N. Fazio, *Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack,* PKC'03, LNCS 2567, 100–115, 2003.
6. Y. Dodis, N. Fazio, A. Kiayias and M. Yung, *Scalable Public-Key Tracing and Revoking,* Principles of Distributed Computing (PODC), July 2003.
7. D. Halevy and A. Shamir, *The LSD broadcast encryption scheme,* Crypto'02, LNCS 2442, 47–60, 2002.
8. A. Kiayias and M. Yung, *Self Protecting Pirates and Black-Box Traitor Tracing,* Crypto'01, LNCS 2139, 63–79, 2001.
9. A. Kiayias and M. Yung, *On Crafty Pirates and Foxy Tracers,* SP-DRM'01, LNCS 2320, 22–39, 2001.
10. A. Kiayias and M. Yung, *Traitor Tracing with Constant Transmission Rate,* EuroCrypt'02, LNCS 2332, 450–465, 2002.
11. C.H. Kim, Y.H. Hwang and P.J. Lee, *An Efficient Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack,* AsiaCrypt'03, LNCS 2894, 359–373, 2003.
12. K. Kurosawa and Y. Desmedt, *Optimum Traitor Tracing and Asymmetric Schemes with Arbiter,* EuroCrypt'98, LNCS 1403, 145–157, 1998.
13. K. Kurosawa and T. Yoshida, *Linear Code Implies Public-Key Traitor Tracing,* PKC'02, LNCS 2274, 172–187, 2002.
14. S. Mäki, *On Long-Lived Public-Key Traitor Tracing: First Steps,* Tik-110.501 Seminar on Network Security (2000), Helsinki University of Technology.
15. S. Mitsunari, R. Sakai and M. Kasahara, *A New Traitor Tracing,* IEICE Trans. Fundamentals, Vol. E85-A, No. 2, Feb 2002.
16. D. Naor, M. Naor and J. Lotspiech, *Revocation and Tracing Schemes for Stateless Receivers,* Crypto'01, LNCS 2139, 41–62, 2001.
17. M. Naor and B. Pinkas, *Efficient Trace and Revoke Schemes,* Financial Cryptography'00, LNCS 1962, 1–20, 2001.
18. D.R. Stinson and R. Wei, *Key Preassigned Traceability Schemes for Broadcast Encryption,* SAC'98, LNCS 1556, 144–156, 1998.
19. V.D. Tô, R. Safavi-Naini and F. Zhang, *New Traitor Tracing Schemes Using Bilinear Map,* ACM DRM'03, 67–76, 2003.
20. W. Tzeng and Z. Tzeng, *A Public-Key Traitor Tracing Scheme with Revocation Using Dynamic Shares,* PKC'01, LNCS 1992, 207–224, 2001.
21. D.M. Wallner, E.J. Harder, and R.C. Agee, *Key Management for Multicast: Issues and Architectures,* IETF Network Working Group, RFC 2627, 1999.
22. C.K. Wong, M. Gouda and S. Lam, *Secure Group Communications Using Key Graphs,* ACM SIGCOMM'98, 68–79, 1998.