A Public-Key Traitor Tracing Scheme with Revocation Using Dynamic Shares*

Wen-Guey Tzeng and Zhi-Jia Tzeng

Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 30050
{tzeng,zjtzeng}@cis.nctu.edu.tw

Abstract. We proposed a new public-key traitor tracing scheme with revocation capability using the dynamic share and entity revocation techniques. The enabling block of our scheme is independent of the number of subscribers, but dependent on the collusion and revocation thresholds. Each receiver holds one decryption key only. Our traitor tracing algorithm works in a black-box way and is conceptually simple. The *distinct feature* of our scheme is that when the traitors are found, we can revoke their private keys (up to some threshold z) without updating any private key of the remaining subscribers. Furthermore, we can restore the decryption privilege of a revoked private key later. We can actually increase the revocation capability beyond z with dynamic assignment of shares into the enabling block. This property makes our scheme highly practical. Previously proposed public-key traitor tracing schemes have to update all existing private keys even when revoking one private key only. Our scheme is as efficient as Boneh and Franklin's scheme in many aspects. Our traitor tracing scheme is fully k-resilient such that our traitor tracing algorithm can find all traitors if the number of them is k or less. The encryption algorithm of our scheme is semantically secure assuming that the decisional Diffie-Hellman problem is hard. We also proposed a variant traitor tracing scheme whose encryption algorithm is semantically secure against the adaptive chosen ciphertext attack assuming hardness of the decisional Diffie-Hellman problem.

Keywords: broadcast encryption, traitor tracing, revocation.

1 Introduction

A broadcast encryption scheme [10] involves a sender and multiple (authorized) receivers. The sender has an encryption key and each receiver has a decryption (private) key such that the sender can encrypt a message and broadcast the ciphertext so that only the authorized receivers can decrypt the ciphertext.

Consider a situation that a content supplier distributes some digital data to its subscribers by a broadcast channel. To protect the data from eavesdropping,

* Research supported in part by the National Science Council grant NSC-89-2213-E-009-180 and by the Ministry of Education grant 89-E-FA04-1-4, Taiwan, ROC.

the content supplier encrypts the data and broadcast the ciphertext such that only its subscribers can decrypt the ciphertext. The content supplier gives each subscriber a decoder (decoding box) for decrypting the broadcast ciphertext. Each decoder consists of a tailored private key and a decryption program. However, a *traitor* (malicious subscriber) may clone his decoder (and the private key in it) and sell the pirate decoders for profits. The traitor may modify the private key and the decryption program in the pirate decoder to avoid leaking his identity. Furthermore, some traitors may together create a new and legal private key that cannot be traced to their creators. To deter the attack, when a pirate decoder is confiscated, the content supplier wants to reveal the private key in it and trace back to its original owners. A traitor tracing scheme is a broadcast encryption scheme with capability of dealing with the above scenario [6]. Furthermore, the content supplier may want to revoke the keys of traitors without too much work, such as, updating each subscriber's key. We focus on providing revocation capability to public-key traitor tracing schemes.

The basic technique of broadcast encryption is: first, to select a (or a set of) session key s for encrypting the broadcast data as the *cipher block* and embed the session key in the *enabling block*; then, to broadcast $\langle$*enabling block, cipher block*$\rangle$ to all subscribers. Any decoder with a legal private key can extract the session key from the enabling block and then uses the session key to decrypt the cipher block. A traitor tracing scheme tries to identify traitors by finding out the keys embedded in the confiscated pirate decoder. There are two measures for efficiency: the storage for the private key (or keys) in a decoder and the size of enabling block. Sometimes, encryption and decryption time is also considered.

The secret-key and coding approach has each decoder holding a set of keys (or codewords) such that the keys in the pirate decoder can be identified by the combinatorial methods [4,6,11,15,17,18,20]. There is a trade-off between the size of enabling block and the number of keys held by each decoder [5,14]. Generally speaking, if the number of subscribers is large, say millions, the schemes become impractical as one of the measures grows proportionally with the number of subscribers.

The public-key approach tries to have the size of enabling block independent of the number of subscribers and each decoder holding one key only [3,13]. This is achieved at the expense of tracing capability and computation time, for example only the collusion of k or less traitors can be dealt with for some threshold k. Boneh and Franklin's traitor tracing scheme is algebraic and deterministic such that k or less traitors who create a single-key pirate decoder can be traced efficiently. However, they have to embed a hidden trapdoor in the modulus so that the discrete logarithm problem over $Z^*_{n^2}$ can be solved in polynomial time.

As to other directions, Naor and Pinkas [15] proposes a threshold traitor tracing scheme that can trace the private keys in a pirate decoder if the decoder's decrypting probability is over some threshold. Fiat and Tassa's dynamic traitor tracing scheme [9] uses the watermarking technique. By observing the watermarks output by a pirate decoder on the fly, they can trace the traitors who created the pirate decoder.

Our results. We propose a new public-key traitor tracing scheme with revocation capability using the dynamic share and entity revocation techniques [2]. The enabling block of our scheme is independent of the number of subscribers, but dependent on the collusion and revocation thresholds, which are k and z, respectively. Each decoder stores only one private key. Our traitor tracing algorithm works in a black-box way and is conceptually simple. Our traitor tracing scheme is fully k-resilient, that is, our traitor tracing algorithm can find all traitors if the number of them is k or less. The encryption algorithm of our scheme is semantically secure against the passive adversary assuming hardness of the decisional Diffie-Hellman problem.

The *distinct feature* of our scheme is that when the traitors are found, we can revoke their private keys (up to z keys totally) without updating any private key of the remaining subscribers. Furthermore, we can restore a revoked private key later. We can actually increase the revocation capability beyond the threshold z with dynamic assignment of shares into the enabling block. This property makes our scheme highly practical. Previously proposed public-key traitor tracing schemes have to update all existing private keys even if revoking one private key only.

Our scheme is as efficient as Boneh and Franklin's scheme in many aspects. For example, the encryption and decryption algorithms of our scheme take $O(z)$ modular exponentiations. Our black-box tracing algorithm takes $O(n^k)$ time when $k \ll n$. Note that the encryption key of our scheme is dynamically dependent on the revoked traitors, while that Boneh and Franklin's scheme is fixed.

We also propose a variant traitor tracing scheme whose encryption algorithm is semantically secure against the adaptive chosen ciphertext attack assuming that computing the decisional Diffie-Hellman problem is hard.

Note: some of our results in this paper are independently discovered in [16] by Naor and Pinkas. Our scheme possesses traceability in addition.

2 Preliminaries

In this section we review the polynomial interpolation, the decisional Diffie-Hellman (DDH) problem and the chosen ciphertext attack and provide the definition for a traitor tracing scheme.

Polynomial interpolation. Let $f(x) = \sum_{i=0}^{z} a_i x^i$ be a polynomial of degree $z \geq 1$. Assume that each user i is given a share $(x_i, f(x_i))$. Then, a group of $z+1$ users, say users $0, 1, \ldots, z$, can compute the polynomial $f(x)$ by the Lagrange interpolation method, or equivalently solving the system of equations:

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^z \\ 1 & x_1 & \cdots & x_1^z \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_z & \cdots & x_z^z \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_z \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_z) \end{pmatrix}$$

Let $XA = F$ denote the above system of equations. If $\det(X) \neq 0$, we can solve all coefficients of $f(x)$ by $A = X^{-1}F$. The constant term a_0 is equal to the first row vector of X^{-1} multiplying F, which is

$$\sum_{t=0}^{z}(f(x_t) \cdot \prod_{0 \le j \ne t \le z} \frac{x_j}{x_j - x_t}).$$

where $\lambda_t = \prod_{0 \le j \ne t \le z} \frac{x_j}{x_j - x_t}, 0 \le t \le z$, are the Lagrange coefficients. Furthermore, in the exponent case, if we are given $(x_0, g^{rf(x_0)})$, $(x_1, g^{rf(x_1)})$, $\ldots$, $(x_z, g^{rf(x_z)})$, we can compute

$$g^{ra_0} = \prod_{t=0}^{z}(g^{rf(x_t)})^{\lambda_t}.$$

for arbitrary r. On the other hand, if $\det(X) = 0$, we cannot get any information about a_0 or g^{ra_0}.

In traitor tracing, a set of legal users may combine their shares linearly to form a new "share", which is the main threat that haunts some public-key based traitor tracing schemes [13]. For example, the legal users $z+i$ and $z+j$, $i, j \ge 1$ and $i \ne j$, can combine their shares to form a new "share"

$$(a+b, ax_{z+i} + bx_{z+j}, \ldots, ax_{z+i}^z + bx_{z+j}^z, af(x_{z+i}) + bf(x_{z+j})).$$

Together with the shares $(x_0, f(x_0)), (x_1, f(x_1)), \ldots, (x_{z-1}, f(x_{z-1}))$, one can compute a_0 by solving the system of equations:

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^z \\ 1 & x_1 & \cdots & x_1^z \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{z-1} & \cdots & x_{z-1}^z \\ a+b & ax_{z+i} + bx_{z+j} & \cdots & ax_{z+i}^z + bx_{z+j}^z \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_z \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_{z-1}) \\ af(x_{z+i}) + bf(x_{z+j}) \end{pmatrix}$$

We observe that if a pirate P gets a share by linear combination of m shares of traitors $j_1, j_2, \ldots, j_m, m \le z$, then P and the traitors together cannot compute a_0, or g^{ra_0} in the exponent case. We base our traitor tracing algorithm on this observation. In our system, we give each user i a share $(x_i, f(x_i))$. If we suspect that the users $j_1, j_2, \ldots, j_m, m \le z$, are traitors, we broadcast data encrypted with the session key s, which is embedded in sg^{ra_0}, together with the shares

$$(x_{j_1}, g^{rf(x_{j_1})}), \ldots, (x_{j_m}, g^{rf(x_{j_m})}), (l_1, g^{rf(l_1)}), \ldots, (l_{z-m}, g^{rf(l_{z-m})})$$

where $l_1, l_2, \ldots, l_{z-m}$ are arbitrarily chosen and different from $x_{j_1}, x_{j_2}, \ldots, x_{j_m}$. A user who is not a traitor can compute g^{ra_0} and thus s. We confirm the traitors if the pirate decoder cannot decrypt the data properly.

Decisional Diffie-Hellman problem. Let G be a group of a large prime order q. Consider the following two distribution ensembles R and D:

- $R = (g_1, g_2, u_1, u_2) \in G^4$, where g_1 and g_2 are generators of G_q;
- $D = (g_1, g_2, u_1, u_2)$, where g_1 and g_2 are generators of G_q and $u_1 = g_1^r$ and $u_2 = g_2^r$ for $r \in Z_q$.

The DDH problem is to distinguish the distribution ensembles R and D. That is, we would like to find a probabilistic polynomial-time algorithm A such that, for some positive constant c and all sufficiently large complexity parameter n,

$$|\Pr[A(R_n) = 1] - \Pr[A(D_n) = 1]| \geq 1/n^c,$$

where R_n and D_n are the size-n distributions of R and D, respectively.

Chosen ciphertext attack. The chosen ciphertext attack on an encryption scheme works as follows. Let PK be the public key of the scheme. The probabilistic polynomial-time adversary $\mathcal{A}$ of the attack has two algorithms A_1 and A_2. A_1 takes as input PK, makes some queries to the decryption oracle adaptively, and outputs two messages m_0 and m_1. Then, the encryption oracle randomly chooses a bit d and encrypts m_d as $C = E(PK, m_d)$. A_2 takes as input PK, m_0, m_1 and C, makes some queries to the decryption oracle in an adaptive way, and outputs d'. The decryption oracle takes as an input a ciphertext C', $C' \neq C$, and returns its corresponding plaintext m'. We say that A attacks the encryption scheme successfully if the probability of $d = d'$ is $1/2 + \varepsilon$ for some non-negligible function ε, where the probability is taken over all coin tossing of d, A_1 and A_2.

Traitor tracing scheme. A traitor tracing scheme consists of the following functions.

- **System setup.** The content supplier sets up the system algorithms and related parameters.
- **Registration.** After system setup, a user (subscriber) can register to the system and get a data decoder that contains a private key specific to the subscriber. A data decoder with a legal private key can decode the data broadcast by the content supplier.
- **Encryption.** When the content supplier would like to broadcast data M, it uses the encryption algorithm E with an appropriate key s to encrypt the data as the enabling block T and the cipher block $C = E(s, M)$ such that only legal subscribers who have decoders with appropriate keys can decrypt $\langle T, C \rangle$ to get the content M.
- **Decryption.** The data decoder consists of a decryption algorithm D such that with an appropriate private key the decoder can decrypt the broadcast $\langle T, C \rangle$ to get the message.
- **Traitor tracing.** If the content supplier gets a decoder, it wants to determine who is the original owner of the private key in the decoder. It may be that some legal subscribers conspire to compute some key that is not legal, but able to decrypt, maybe with a different decryption algorithm, the broadcast data. The traitor tracing algorithm need be able to reveal the identities of the conspirators. If we trace the owner of the private key in the decoder by observing the relation between input and output of the decoder, it is called *black box* tracing.

A tracing scheme is *k-resilient* if it can find at least one traitor among the k or less traitors. It is *fully k-resilient* if it can find all of them.

Note. In order to simplify presentation, we omit the security parameter (or complexity measure) n from the related parameters. For example, when we say a probability ϵ is negligible, we mean that for any positive constant c, $\epsilon = \epsilon(n) < 1/n^c$ for large enough n. A probability δ is overwhelming if $\delta = 1 - \epsilon$ for some negligible probability ϵ.

3 Our Traitor Tracing Scheme

In this section we present our traitor tracing scheme. Let k be the maximum number of colluded subscribers (traitors) and z be the revocation threshold, ie., at most z private keys of traitors can be revoked. We set $z \geq 2k - 1$.

System setup. Let G_q be a group of a large prime order q. The content supplier selects a degree-z polynomial $f(x) = \sum_{t=0}^{z} a_t x^t \pmod{q}$ with coefficients over Z_q. $f(x)$ is the content supplier's secret key. The content supplier publishes the public key

$$\langle g, g^{a_0}, g^{f(1)}, \ldots, g^{f(z)} \rangle$$

for a subscriber to verify his private key.

Registration. When a subscriber $i, i > z$, registers, the content supplier gives the subscriber i a decoder with the private key $(i, f(i))$. The subscriber i verifies whether the received key is correct by checking whether

$$g^{a_0} = \prod_{t=0}^{z} g^{f(x_t)\lambda_t}$$

where $x_0 = 1, x_1 = 2, \ldots, x_{z-1} = z, x_z = i$. If it is so, the subscriber i gets a decoder with the private key $(i, f(i))$.

Encryption. The content supplier randomly selects z unused shares, which are not assigned to any subscriber,

$$(j_1, f(j_1)), (j_2, f(j_2)), \ldots, (j_z, f(j_z))$$

and a one-time random number $r \in Z_q$, and computes the *enabling block*

$$T = \langle s g^{ra_0}, g^r, (j_1, g^{rf(j_1)}), (j_2, g^{rf(j_2)}), \ldots, (j_z, g^{rf(j_z)}) \rangle,$$

where s is the session key of encrypting broadcast data. To broadcast message M, the content supplier broadcasts

$$E(f(x), M) = \langle T, E'(s, M) \rangle$$

where E' is a secret-key cipher, such as DES.

For every possible m-coalition $\{c_1, c_2, \ldots, c_m\}$ of the subscribers, $m \leq k$,

1. Randomly selects $z - m$ unused shares, say, $\{j_1, \ldots, j_{z-m}\}$.
2. Construct a testing message $E(f(x), M) = \langle T, E'(s, M)\rangle$, where

$$T = \langle sg^{ra_0}, g^r, (c_1, g^{rf(c_1)}), (c_2, g^{rf(c_2)}), \ldots, (c_m, g^{rf(c_m)}), \\ (j_1, g^{rf(j_1)}), (j_2, g^{rf(j_2)}), \ldots, (j_{z-m}, g^{rf(j_{z-m})})\rangle.$$

3. Feed $\langle T, E'(s, M)\rangle$ to the decoder.
4. If the decoder does not output the correct data M, we set $\{c_1, c_2, \ldots, c_m\}$ as a possible set of traitors.

Output the smallest of all possible sets of traitors found in Step 4.

Fig. 1. A traitor tracing algorithm

Decryption. When receiving the broadcast data $\langle T, E'(s, M)\rangle$, the subscriber i uses T to compute s and then uses s to decrypt $E'(s, M)$ to get M. The subscriber i computes s with the equation:

$$s = sg^{ra_0} / [(g^r)^{f(i)\lambda_z} \cdot \prod_{t=0}^{z-1} (g^{rf(x_t)})^{\lambda_t}]$$

where $x_0 = j_1, x_1 = j_2, \ldots, x_{z-1} = j_z$ and $x_z = i$.

Traitor tracing. We present two black box traitor tracing algorithms. We can mix their use in tracing traitors.

Assume that m subscribers $\{c_1, c_2, \ldots, c_m\}, m \leq k$, use their shares to create a decoding key in any form. As long as the share indices $\{j_1, j_2, \ldots, j_z\}$ in the enabling block covers $\{c_1, c_2, \ldots, c_m\}$, ie.,$\{c_1, c_2, \ldots, c_m\} \subseteq \{j_1, j_2, \ldots, j_z\}$, there is no way that the decoder can use the conspired key and the enabling block to decode the data assuming that computing the discrete logarithm over G_q is hard. Our first traitor tracing algorithm is based on this idea. If we suspect the subscribers $\{c_1, c_2, \ldots, c_m\}, m \leq k$, are traitors, we put their shares in the enabling block

$$\langle sg^{ra_0}, g^r, (c_1, g^{rf(c_1)}), \ldots, (c_m, g^{rf(c_m)}), (j_1, g^{rf(j_1)}), \ldots, (j_{z-m}, g^{rf(j_{z-m})})\rangle,$$

where $j_1, j_2, \ldots, j_{z-m}$ are unused indices and different from $\{c_1, c_2, \ldots, c_m\}$. Therefore, our black box tracing algorithm is in Figure 1

Our second traitor tracing algorithm uses the opposite direction, that is, the traitors can decrypt the enabling block. If we suspect $\{c_1, c_2, \ldots, c_m\}$, $m \leq k$, are traitors, we find a degree-z polynomial $h(x) = \sum_{t=0}^{z} b_t x^t$ that passes points $(c_1, f(c_1))$, $(c_2, f(c_2))$, $\ldots$, $(c_m, f(c_m))$. $h(x)$ is significantly different from $f(x)$, ie., they share m common points only. We use $h(x)$ to create the enabling block. Let $\{j_1, j_2, \ldots, j_z\}$ be the indices other than $\{c_1, c_2, \ldots, c_m\}$. We feed $T = \langle sg^{rb_0}, g^r, (j_1, g^{rh(j_1)}), (j_2, g^{rh(j_2)}), \ldots, (j_z, g^{rh(j_z)})\rangle$ to the decoder. Note that this enabling block is computationally indistinguishable from the one created using $f(x)$, see Lemma 1. A share index x_i that is not in $\{c_1, c_2, \ldots, c_m\}$ cannot decode the enabling block correctly. If the decoder outputs correct data, we confirm that $\{c_1, c_2, \ldots, c_m\}$ are traitors.

Lemma 1. *For degree-z polynomials $f(x)$ and $h(x)$, the distributions of the enabling blocks constructed by $f(x)$ and $h(x)$ are computationally indistinguishable assuming that the DDH problem is hard.*

Proof. Note that the distinguisher does not know $f(x)$ and $h(x)$. Let g be a fixed generator of G_q and $a \in_R S$ denote that a is chosen from the set S uniformly and independently. Consider the following 3 distributions:

1. $T_1 = \langle S, g^r, (c_1, g_1^r), (c_2, g_2^r), \ldots, (c_z, g_z^r)\rangle$, where $r \in_R Z_q$, $S \in_R G_q$, $c_i \in_R G_q$, $g_i = g^{f(c_i)}$. This is the enabling block constructed by $f(x)$.
2. $R = \langle S, g^r, (c_1, u_1), (c_2, u_2), \ldots, (c_z, u_z)\rangle$, where $r \in_R Z_q$, $S \in_R G_q$, $c_i \in_R G_q$, $u_i \in_R G_q$, $1 \leq i \leq z$.
3. $T_2 = \langle S, g^r, (c_1, g_1^r), (c_2, g_2^r), \ldots, (c_z, g_z^r)\rangle$, where $r \in_R Z_q$, $S \in_R G_q$, $c_i \in_R G_q$, $g_i = g^{h(c_i)}$. This is the enabling block constructed by $h(x)$.

We can easily show that T_1 and R, and R and T_2 are computationally indistinguishable. Therefore, T_1 and T_2 are computationally indistinguishable. □

Framing. We now address the framing problem [3]. We show that it is not possible for two disjoint sets of k subscribers to construct the same "new" share by linear combination. Therefore, framing is not possible by linear combination of shares.

Lemma 2. *Let $C = \{c_1, c_2, \ldots, c_k\}$ and $D = \{d_1, d_2, \ldots, d_k\}$ be two disjoint subscriber sets. All linear combination of shares of C and those of D are different except the zero point.*

Proof. We can represent a share i as a $z+2$-dimensional vector

$$v_i = (1, i, i^2, \ldots, i^z, f(i)).$$

Since it is a point of a degree-z polynomial, any $z+1$ different shares are linearly independent. If one can use the shares of C and the shares of D to construct the same non-zero share by linear combination, we have

$$\sum_{i=1}^{k} a_i v_{c_i} = \sum_{i=1}^{k} b_i v_{d_i} \neq \mathbf{0}.$$

Therefore, we have

$$\sum_{i=1}^{k} a_i v_{c_i} - \sum_{i=1}^{k} b_i v_{d_i} = \mathbf{0}.$$

This is a contradiction since not all a_i's and b_i's are zero and $C \cup D$ is linearly independent. □

Complexity. To broadcast data, the computing time of creating an enabling block is O(z) modular exponentiations, which can be pre-computed. The runtime of the decryption algorithm for each subscriber is O(z) modular exponentiations also. The traitor tracing algorithm runs in O(C_k^n), where n is the total number of subscribers. When $n \gg k$, the runtime is about $O(n^k)$.

3.1 Revocation of Traitors

After a pirate decoder is confiscated and the traitors are revealed, we would like to revoke the private keys of the traitors since thousands of the pirate decoders may be sold.

Assume that $C = \{c_1, c_2, \ldots, c_m\}$, $m \le z$, is the set of found traitors or revoked subscribers. We can revoke their shares without updating the private keys of the remaining subscribers. To broadcast data to the remained subscribers, instead of randomly choosing unused shares for the enabling block, the content suppliers fixes the first m shares as

$$(c_1, g^{rf(c_1)}), (c_2, g^{rf(c_2)}), \ldots, (c_m, g^{rf(c_m)})$$

and randomly chooses the rest $z - m$ unused shares

$$(j_1, g^{rf(j_1)}), (j_2, g^{rf(j_2)}), \ldots, (j_{z-m}, g^{rf(j_{z-m})}).$$

We can see that the revoked shares or their combinations cannot be used to decrypt the broadcast data since their shares are in the enabling block. We can revoke at most z shares totally before updating the shares of the remaining subscribers.

3.2 Restoration of a Revoked Key

If for some reason we would like to restore the decryption privilege of a revoked key, we simply do not use it in forming the enabling block. The restored key can decrypt the broadcast ciphertext again.

3.3 Revocation beyond the Threshold

It is possible to revoke more than z traitors. The idea is that if a pirate decoder can get at most c percent of data M, say 95%, the partial part of M is useless [1]. For example, if a pirate decoder can only decrypt 95% of a movie, the traitor is revoked de facto.

Assume that $C = \{c_1, c_2, \ldots, c_m\}$, $m > z$, is the set of found traitors or revoked subscribers. Without loss of generality, let $m = tz$. To broadcast data M to the remained subscribers, we partition M as $M_1||M_2||\cdots||M_l$. For each M_i, $1 \le i \le l$, we construct an enabling block T_i with shares

$$(c_{i_1}, g^{rf(c_{i_1})}), (c_{i_2}, g^{rf(c_{i_2})}), \ldots, (c_{i_r}, g^{rf(c_{i_z})}),$$

where $c_{i_1}, c_{i_2}, \ldots, c_{i_z}$ are randomly chosen from C. With appropriately chosen r and l, each traitor in C can decrypt at most c percent of M in average.

Let $E_j^{(i)}$ be the probability that c_i is chosen into T_j. We can see that $E_j^{(i)} = z/m = 1/t$. The probability that c_i is not chosen into any T_j, $1 \leq j \leq l$, is $(1 - 1/t)^l$. With $l = 3t$, $(1 - 1/t)^{3t} \simeq 1/e^3 \simeq 0.05$. That is, to increase the revocation capability by 5 folds, we partition M into 15 parts. Furthermore, we can adjust these values dynamically.

3.4 Speedup of Tracing

Since the runtime of the tracing algorithm is $\mathrm{O}(C_k^n)$, when n or k is large, the algorithm is not efficient. In practice, we would like to have a more efficient tracing algorithm.

A practical solution to this problem is to group subscribers into classes C_1, $C_2, \ldots, C_r$. Each class C_i consists of a reasonable number of subscribers by the subscribers' residence, etc. For each class C_i, the content supplier uses a different polynomial $f_i(x)$ as the secret key. A subscriber j in class C_i is given a share $(j, f_i(j))$. The data M broadcast to the subscribers of class C_i are encrypted as $E(f_i(x), M)$. The decryption and tracing algorithms are the same as the original ones except that the keys are different for different classes.

Grouping subscribers can make our revocation mechanism more practical. It will be less frequent to revoke private keys in a class since the size is smaller. Even if the content supplier wants to revoke the $(z+1)$th private key in a class, only the private keys in the class have to be updated.

4 Security Analysis

We consider both semantic security and security against the z-coalition attack, in which any coalition of z or less legal subscribers cannot compute a legal private key for decryption.

The encryption algorithm of our scheme is semantically secure against a passive adversary if the DDH problem in G_q is hard (or computationally infeasible). Recall that $D = \langle g_1, g_2, g_1^r, g_2^r \rangle$ and $R = \langle g_1, g_2, g_1^a, g_2^b \rangle$ where g_1, g_2 are generators and a, b and r are randomly chosen over Z_q.

Theorem 1 (Semantic security). *Assume that the DDH problem is hard. The encryption algorithm of our traitor tracing scheme is semantically secure against the passive adversary.*

Proof. Suppose that our encryption algorithm is not semantically secure against the passive adversary. We show that there is a probabilistic polynomial-time algorithm $\mathcal{B}$ that distinguishes D from R with a non-negligible advantage ε.

Assume that adversary $\mathcal{A}$ attacks our encryption algorithm successfully in terms of semantic security. $\mathcal{A}$ has two procedures A_1 and A_2. Given the public key $\langle g, g^{a_0}, g^{f(1)}, \ldots, g^{f(z)} \rangle$ of the content supplier, A_1 finds two session keys s_0 and s_1 in G_q such that A_2 can distinguish them by observing the enabling block.

Let $\langle g_1, g_2, u_1, u_2\rangle$ be an input of the DDH problem. The following algorithm $\mathcal{B}$ shall decide whether $\langle g_1, g_2, u_1, u_2\rangle$ is from D or R.

1. Randomly choose $a_i \in Z_q$, $1 \le i \le z$, and let $f'(x) = \sum_{t=1}^{z} a_t x^t$. Let $g = g_1$, $g^{a_0} = g_2$, $g^{f(1)} = g_2 g_1^{f'(1)}$, $\ldots$, $g^{f(z)} = g_2 g_1^{f'(z)}$, where $f(x) = f'(x) + a_0$. Note that we don't know a_0.
2. Feed the public key $\langle g, g^{a_0}, g^{f(1)}, \ldots, g^{f(z)}\rangle$ to A_1. A_1 returns s_0 and s_1 in G_q.
3. Randomly select $d \in \{0,1\}$ and encrypt s_d as
$$C = \langle s_d u_2, u_1, (j_1, u_2 u_1^{f'(j_1)}), \ldots, (j_z, u_2 u_1^{f'(j_z)})\rangle$$
where $j_1, j_2, \ldots, j_z$ are randomly chosen.
4. Feed C to A_2 and get a return d'. Then, the algorithm outputs 1 if and only if $d = d'$.

If $\langle g_1, g_2, u_1, u_2\rangle$ is from D, $g = g_1, g_2 = g^{a_0}, u_1 = g^r, u_2 = g_2^r = g^{ra_0}$ and $u_2 u_1^{f'(j_i)} = g^{rf(j_i)}$ for $1 \le i \le z$. Thus, C is the encryption of s_d and $\Pr[\mathcal{B}(g_1, g_2, u_1, u_2) = 1] = \Pr[A_2(C) = d] = 1/2 + \varepsilon$. Otherwise, since $u_1 = g_1^a$ and $u_2 = g_2^b$, the distribution of C is the same for $d = 0$ and $d = 1$. Thus, $\Pr[\mathcal{B}(g_1, g_2, u_1, u_2) = 1] = \Pr[A_2(C) = d] = 1/2$. Therefore, $\mathcal{B}$ distinguishes D from R with a non-negligible advantage ε. □

The encryption algorithm of our scheme is secure against z-coalition assuming that computing the discrete logarithm is hard.

Theorem 2. *Assume that computing the discrete logarithm over G_q is hard. No coalition of z or less legal subscribers can compute the private key of another legal subscriber with a non-negligible probability.*

Proof. Assume that the probabilistic polynomial-time algorithm $\mathcal{A}$ can compute a new share (private key) $(x_u, f(x_u))$ from the given public key $\langle g, g^{a_0}, g^{f(1)}, g^{f(2)}, \ldots, g^{f(z)}\rangle$ and z shares $(x_1, f(x_1)), \ldots, (x_z, f(x_z))$ with a non-negligible probability ε. We construct another probabilistic polynomial-time algorithm $\mathcal{B}$ to compute the discrete logarithm over G_q with an overwhelming probability.

Let (p, g, y) be the input of the discrete logarithm problem. The following algorithm $\mathcal{B}'$ computes $\log_g y \pmod p$ with a non-negligible probability. Let $y = g^{a_0}$ and $f(x)$ be the degree-z polynomial passing $(0, a_0)$ and $(x_i, f(x_i)), 1 \le i \le z$. By Lagrange interpolation, we can compute $g^{f(i)}, 1 \le i \le z$. We feed the public key $\langle g, g^{a_0}, g^{f(1)}, g^{f(2)}, \ldots, g^{f(z)}\rangle$ and z shares $(x_1, f(x_1)), \ldots, (x_z, f(x_z))$ to $\mathcal{A}$ and shall get a new share $(x_u, f(x_u))$ with a non-negligible probability. With the given z shares and $(x_u, f(x_u))$, we can compute $f(0) = a_0$.

By applying the randomized technique to $\mathcal{B}'$ for a polynomial number of times, we get $\mathcal{B}$. □

5 Chosen Ciphertext Security

It is often desirable to have a cryptosystem secure against the adaptive chosen ciphertext attack. By the technique of [7], we modify our scheme so that it becomes secure against the adaptive chosen ciphertext attack under the standard intractability assumptions. Recall that G_q is a group of order-q and g is its generator. Our variant traitor tracing scheme is as follows.

System setup. The content supplier randomly chooses a degree-z polynomial $f(x) = \sum_{t=0}^{z} a_t x^t$ with coefficients in G_q and $a, b, x_1, x_2, y_1, y_2 \in Z_q$. Its secret key is $\langle f(x), a, b\rangle$ and public key is

$$\langle g, g^{a_0}, g^{f(1)}, \ldots, g^{f(z)}, g^a, g^b, c, d, H\rangle$$

where $c = g^{ax_1} g^{bx_2}$, $d = g^{ay_1} g^{by_2}$, and H is a collision-resistant hash function. Let $f'(x) = a^{-1}(f(x) - b)$ or $f(x) = af'(x) + b$.

Registration. When a subscriber $i, i > z$, registers to the system, the content supplier gives him a private key

$$(i, f'(i), x_1, x_2, y_1, y_2)$$

The subscriber i can verify his share.

Encryption. To broadcast data M, the content supplier randomly selects a session key $s \in G_q$, a one-time number $r \in Z_q$ and z unused indices $j_1, j_2, \ldots, j_z$ and computes the enabling block

$$T = \langle sg^{ra_0}, (j_1, g^{rf(j_1)}), \ldots, (j_z, g^{rf(j_z)}), g^{ra}, g^{rb}, v\rangle$$

where $v = c^r d^{r\alpha}$ and $\alpha = H(sg^{ra_0}, (j_1, g^{rf(j_1)}), \ldots, (j_z, g^{rf(j_z)}), g^{ra}, g^{rb})$. We use $T = \langle S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b, v\rangle$ to denote the enabling block.

Decryption. When receiving the enabling block $T = \langle S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b, v\rangle$, the subscriber i with the private key $\langle i, f'(i), x_1, x_2, y_1, y_2\rangle$ can compute the session key s by the following steps.

- Compute $\alpha = H(S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b, v)$;
- Check whether $F_a^{x_1+y_1\alpha} F_b^{x_2+y_2\alpha} = v$;
- If the checking fails, reject the enabling block; otherwise, compute

$$S/[F_a^{f'(i)\lambda_z} \cdot F_b^{\lambda_z} \cdot \prod_{t=0}^{z-1} F_{j_t}^{\lambda_t}] = S/g^{ra_0} = s,$$

  where $x_0 = j_1, x_1 = j_2, \ldots, x_{z-1} = j_z$ and $x_z = i$.

Traitor tracing. The traitor tracing algorithm is the same as that in Section 3.

The encryption algorithm of the above traitor tracing scheme is semantically secure against the adaptive chosen ciphertext attack.

Theorem 3. *Assume that the DDH problem is hard. The encryption algorithm of the above traitor tracing scheme is semantically secure against the adaptive chosen ciphertext attack.*

Proof. We assume that there is a probabilistic polynomial-time adversary $\mathcal{A}$ that can break our encryption algorithm with a non-negligible probability ε. $\mathcal{A}$ consists of two algorithms A_1 and A_2. A_1 takes as input of the content supplier's public key and outputs two session keys s_0 and s_1. Let d be a random bit. A_2 takes as input of the enabling block for s_d, makes some chosen ciphertext queries, and outputs d'. The probability of $d = d'$ is $1/2 + \varepsilon$. By A_1 and A_2, we can construct a probabilistic polynomial-time algorithm $\mathcal{B}$ that distinguishes D from R with a non-negligible probability.

Given a quadruple (g_1, g_2, u_1, u_2) in G_q^4, we construct a simulator S that simulates $\mathcal{A}$'s view in its attack on the algorithm. The simulator S includes an encryption oracle and a decryption oracle. We will show that if the quadruple is from D, the simulation of $\mathcal{A}$'s view will be nearly perfect and if the quadruple is from R, $\mathcal{A}$'s advantage is negligible. The simulator S works as follows.

1. **Key setup.** The content supplier's public key is constructed as follows.
   (a) Select two degree-z polynomials $f'(x) = \sum_{t=0}^{z} a_t x^t$ and $f''(x) = \sum_{t=0}^{z} r_t x^t$ and $w, j \in Z_q$ randomly. Let $g = g_1$, $g_2 = g_1^a = g^a$, $f(x) = f'(x) + wa$ and $g^b = g^{f(j)}/g_2^{f''(j)}$. Note that we don't know the constant coefficient of $f(x)$ since a is unknown. We have $f(j) = f'(j) + wa = af''(j) + b$. We don't know b, either.
   (b) Randomly select x_1, x_2, y_1, y_2 and compute
   $$c = g^{ax_1} g^{bx_2}, d = g^{ay_1} g^{by_2}.$$
   (c) The public key of the content supplier is
   $$\langle g, g^{a_0} g_2^w, g^{f'(1)} g_2^w, g^{f'(2)} g_2^w, \dots, g^{f'(z)} g_2^w, g^a, g^b, c, d, H \rangle.$$

   The above key generation is a bit different from the actual cryptosystem, but the effect is the same. We essentially fix $w = 0$.
2. **Challenge.** Feed the public key of the content supplier to A_1 and get two session keys s_0 and s_1 in G_q.
3. **Encryption.** Randomly pick d and z indices $\{j_1, \dots, j_z\}$ and compute
   $$S = s_d u_1^{a_0} u_2^w, F_{j_1} = u_1^{f'(j_1)} u_2^w, \dots, F_{j_z} = u_1^{f'(j_z)} u_2^w, F_a = u_2,$$
   $$F_b = u_1^{f'(j)} u_2^w / u_2^{f''(j)}, \alpha = H(S, (j_1, F_{j_1}), \dots, (j_z, F_{j_z}), F_a, F_b),$$
   $$\upsilon = F_a^{x_1 + y_1 \cdot \alpha} \cdot F_b^{x_2 + y_2 \cdot \alpha}.$$

   The ciphertext of s_d is $T = \langle S, (j_1, F_{j_1}), \dots, (j_z, F_{j_z}), F_a, F_b, \upsilon \rangle$.
4. **Decryption.** Given the ciphertext T, the decryption oracle first checks validity of T by verifying $u_1^{x_1 + y_1 \alpha} u_2^{x_2 + y_2 \alpha} = \upsilon$. If it is not valid, the oracle rejects it; otherwise, the oracle returns
   $$s = S / [\prod_{t=0}^{z-1} F_{j_t}^{\lambda_{t-1}} \cdot F_a^{f''(j)\lambda_z} \cdot F_b^{\lambda_z}],$$
   where $x_0 = j_1, x_2 = j_1, \dots, x_{z-1} = j_z, x_z = j$.

The above completes description of S. The adversary $\mathcal{B}$ takes as input (g_1, g_2, u_1, u_2) and outputs 1 if and only if $d = A_2(T)$, where T is the enabling block of the challenge s_d .

If the input comes from D, the ciphertext T of encryption of s_d is legitimate. If the input comes from R, the distribution of T is almost the same for $d = 0$ and $d = 1$. To complete the proof, we show:

1. If $\langle g_1, g_2, u_1, u_2 \rangle$ is from D, the joint distribution of the simulator S's output (adversary's view) and the hidden bit d is statistically indistinguishable from that in the actual attack.
2. If $\langle g_1, g_2, u_1, u_2 \rangle$ is from R, the distribution of the hidden bit d is (essentially) independent of S's output.

Lemma 3. *If the simulator S's input $\langle g_1, g_2, u_1, u_2 \rangle$ is chosen from D, the joint distribution of the adversary's view and the hidden bit d is statistically indistinguishable from that in the actual attack.*

Lemma 4. *If the simulator S's input $\langle g_1, g_2, u_1, u_2 \rangle$ is chosen from R, the distribution of the hidden bit d is (essentially) independent of the adversary's view.*

Since the proofs of the above two lemmas are similar to those of Cramer and Shoup [7], we put them in Appendix.

This completes the proof of Theorem 3. □

6 Discussion

Actually, we can drop the content supplier's public key from our traitor tracing schemes if verification of private keys by subscribers is not necessary. This is indeed the case for practicality. Thus, only the content suppblier can do data encryption. Since the enabling blocks are computationally indistinguishable from each other due to the DDH assumption, our scheme should be more secure.

For practicality, we can set $z = k$. In this case, there may be framing problem. The probability that a set of k subscribers can frame a specific set of k subscribers is $1/q$. Assume that there are $m = 10,000,000$ subscribers and k is set as 20. Then, the probability that a set of k subscribers can frame some set of k subscribers is $\leq C_k^m/q \approx m^k/q \approx 1/(10)^{168}$, for q being 1024-bit long.

7 Conclusion

In this work we have proposed a new public-key traitor tracing scheme with revocation capability using dynamic shares. Its distinct feature of revoking private keys makes the protocol highly practical. The scheme's traitor tracing algorithm is fully k-resilient and conceptually simple. The size of the enabling block is independent of the number of subscribers.

Our scheme is semantically secure against the passive adversary assuming that the DDH problem is hard. We also present a variant scheme that is semantically secure against the adaptive chosen ciphertext attack assuming that the DDH problem is hard.

References

1. M. Abdalla, Y. Shavitt, A. Wool, "Key management for restricted multicast using broadcast encryption", *Proceedings of Financial Cryptology 99*, Lecture Notes in Computer Science 1648, Springer Verlag, 1999.
2. J. Anzai, N. Matsuzaki, T. Matsumoto, "A quick group key distribution scheme with "entity revocation"", *Proceedings of Advances in Cryptology - Asiacrypt 99*, Lecture Notes in Computer Science 1716, pp.333-347, Springer Verlag, 1999.
3. D. Boneh, M. Franklin, "An efficient public key traitor tracing scheme", *Proceedings of Advances in Cryptology - Crypto 99*, Lecture Notes in Computer Science 1666, pp.338-353, Springer Verlag, 1999.
4. D. Boneh, J. Shaw, "Collusion-secure fingerprinting for digital data", *IEEE Trasaction on Information Theory* 44(5), pp.1897-1905, 1998. (See also, *Proceedings of Advances in Cryptology - Crypto 95*, Lecture Notes in Computer Science 963, pp.452-465, Springer Verlag, 1995.)
5. R. Canetti, T. Malkin, K. Nissim, "Efficient communication-storage tradeoffs for multicast encryption", *Proceedings of Adnaces in Cryptology - Eurocrypt 99*, Lecture Notes in Computer Science 1592, pp.459-474, 1999.
6. B. Chor, A. Fiat, M. Naor, "Tracing traitor", *Proceedings of Advances in Cryptology - Crypto 94*, Lecture Notes in Computer Science 839, pp.257-270, Springer Verlag, 1994.
7. R. Cramer, V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack", *Proceedings of Advances in Cryptology - Crypto 98*, Lecture Notes in Computer Science 1462, pp.13-25, Springer Verlag, 1998.
8. T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms", *IEEE Transactions on Information Theory* 31(4), pp.469-472, 1985.
9. A. Fiat, T. Tassa, "Dynamic traitor tracing", *Proceedings of Advances in Cryptology - Crypto 99*, Lecture Notes in Computer Science 1666, pp.354-371, Springer Verlag, 1999.
10. A. Fiat, M. Naor, "Broadcast encryption", *Proceedings of Advances in Cryptology - Crypto 93*, Lecture Notes in Computer Science 773, pp.480-491, Springer Verlag, 1993.
11. E. Gafni, J. Staddon, Y.L. Yin, "Efficient methods for integrating traceability and broadcast encryption", *Proceedings of Advances in Cryptology - Crypto 99*, Lecture Notes in Computer Science 1666, pp.372-387, Springer Verlag, 1999.
12. R. Kumar, S. Rajagopalan, A. Sahai, "Coding constructions for blacklisting problems without computational assumptions", *Proceedings of Advances in Cryptology - Crypto 99*, Lecture Notes in Computer Science 1666, pp.609-623, Springer Verlag, 1999.
13. K. Kurosawa, Y. Desmedt, "Optimum traitor tracing and asymmetric schemes", *Proceedings of Advances in Cryptology - Eurocrypt 98*, Lecture Notes in Computer Science 1403, pp.145-157, Springer Verlage, 1998.
14. M. Luby, J. Staddon, "Combinatorial bounds for braodcast encryption", *Proceedings of Advances of Cryptology - Eurocrypt 98*, Lecture Notes in Compouter Science 1403, pp.512-526, Springer Verlag, 1998.

15. M. Naor, B. Pinkas, "Threshold traitor tracing", *Proceedings of Advances in Cryptology - Crypto 98*, Lecture Notes in Computer Science 1462, pp.502-517, Springer Verlag, 1998.
16. M. Naor, B. Pinkas, "Efficient trace and revoke schemes", *Proceedings of Financial Cryptography 00*, 2000.
17. B. Pfitzmann, "Trials of traced traitors", *Proceedings of Workshop on Information Hiding*, Lecture Notes in Computer Science 1174, pp.49-64, Springer Verlag, 1996.
18. B. Pfitzmann, M. Waidner, "Asymmetric fingerprinting for large collusions", *Proceedings of ACM Conference on Computer and Communication Security*, pp.151-160, 1997.
19. A. Shamir, "How to share a secret", *Communications of the ACM*, 22(11), pp.612-613, 1979.
20. D.R. Stinson, R. Wei, "Combinatorial properties and constructions of traceability schemes and frameproof codes", *SIAM J. on Discrete Math* 11(1), pp.41-53, 1998.

Appendix

Lemma 3. Proof. We need argue two things. One is that the output of the encryption oracle and the decryption oracle has the right distribution. The other is that the decryption oracle rejects all invalid ciphertexts except with a negligible probability.

If the input is from D, we have $u_1 = g_1^r$ and $u_2 = g_2^r$ for some r. Therefore, $F_{j_i} = u_1^{f'(j_i)} u_2^w = g^{f(j_i)}$ for $1 \le i \le z$, $F_a = g_2^r = g^{ar}$, $F_b = (u_1^{f'(j)} u_2^w / u_2^{f''(x)}) = g^{br}$, $F_a^{x_1} F_b^{x_2} = c^r$, $F_a^{y_1} F_b^{y_2} = d^r$. Thus, $S = s_d u_1^{a_0} u_2^w = s_d g^{a_0 r} g^{arw} = s_d g^{rf(0)}$ and $v = c^r d^{r\alpha}$, where α is in right form. Hence, the output of the encryption oracle has the right distribution. In addition, $\langle S$, (j_1, F_{j_1}), $\ldots$, (j_z, F_{j_z}), F_a, F_b, $v\rangle$ is a valid ciphertext. Therefore, the decryption oracle outputs $s_d = S / \prod_{t=0}^{z-1} F_{j_t}^{\lambda_{t-1}} \cdot F_a^{f''(j)\lambda_z} \cdot F_b^{\lambda_z}$.

Moreover, we should prove that the decryption oracle rejects all invalid ciphertexts, except with a negligible probability. Consider the distribution of the point $\mathbf{P} = (x_1, x_2, y_1, y_2) \in Z_q^4$. Recall that $g = g_1$ and $\log_g g_2 = a$. ¿From c and d of the public key, we get two equations:

$$\log_g c = ax_1 + bx_2, \tag{1}$$

$$\log_g d = ay_1 + by_2. \tag{2}$$

¿From the output of the encryption oracle, we get another equation:

$$\log_g v = rax_1 + brx_2 + \alpha ray_1 + \alpha rby_2. \tag{3}$$

If the adversary submits an invalid ciphertext $\langle S'$, (j_1, F'_{j_1}), $\ldots$, (j_z, F'_{j_z}), F'_a, $F'_b, v\rangle$ to the decryption oracle, ie., $r_1 = \log_g u'_1 \neq \log_{g_2} u'_2 = r_2$. The decryption oracle will reject, unless $\mathbf{P}$ lies on the hyperplane $\mathbf{H}$ defined by

$$\log_g v' = ar_1x_1 + br_2x_2 + \alpha' ar_1y_1 + \alpha' r_2by_2, \tag{4}$$

where $\alpha' = H(S', (j_1, F'_{j_1}), \ldots, (j_z, F'_{j_z}), F'_a, F'_b)$. Since Equations 1, 2 and 4 are linearly independent, the hyperplane **H** and the plane **P** intersect at a line.

The first time the adversary submits an invalid ciphertext, the decryption oracle rejects it with probability $1 - 1/q$. This rejection constrains the point P with q less points. Furthermore, the ith invalid ciphertext will be rejected with probability at least $1-q/(q^2-q(i-1))=1-1/(q-i+1)$. Therefore, the decryption oracle rejects all invalid ciphertexts except with a negligible probability. □

Lemma 4. Proof. We should prove that if the decryption oracle rejects all invalid ciphertexts, the distribution of the hidden bit b is independent of the adversary's view. Furthermore, we still need to prove that the decryption oracle will reject all invalid ciphertexts, except with a negligible probability.

Let $t_1 = \log_g u_1$ and $t_2 = log_{g_2} u_2$. Assume that $t_1 \neq t_2$ since this holds except with a negligible probability $1/q$. The public key $l = g^{a_0} g^{aw}$ determines the equation:

$$\log_g l = a_0 + aw. \tag{5}$$

Since the decryption oracle only decrypts valid ciphertexts, the adversary obtains only linearly dependent equation $r' \log_g l = r'a_0 + r'wa$. Thus, no further information about (a_0, w) is leaked.

Consider that the output of the encryption oracle, we have $S = s_d g_1^{t_1 a_0} g_2^{t_2 w}$. Let $\beta = g_1^{t_1 a_0} g_2^{t_2 w}$. We get the equation:

$$\log_g \beta = t_1 a_0 + t_2 wa. \tag{6}$$

Clearly, Equations 5 and 6 are linearly independent. We can view β as a perfect one-time pad. As a result that d is independent of the adversary's view.

Next, we argue that the decryption oracle will reject all invalid ciphertexts except with a negligible probability. Let us examine the distribution of $P = (x_1, x_2, y_1, y_2) \in Z_q^4$, based on the adversary's view. ¿From the output v of the encryption oracle, we get the equation:

$$\log_g v = at_1 x_1 + bt_2 x_2 + \alpha a t_1 y_1 + \alpha b t_2 y_2. \tag{7}$$

¿From the adversary's view, P is a random point on the line **L** formed by intersecting the hyperplanes of Equations 1, 2 and 7. Assume that the adversary submits an invalid ciphertext $\langle S', (j_1, F'_{j_1}), \ldots, (j_z, F'_{j_z}), F'_a, F'_b, v' \rangle$. Let $\log_g u'_1 = t'_1$ and $\log_{g_2} u'_2 = t'_2$. There are three cases to consider:

Case I. $\langle S', (j_1, F'_{j_1}), \ldots, (j_z, F'_{j_z}), F'_a, F'_b \rangle = \langle S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b \rangle$. Although the hash values are the same, the decryption oracle still reject because of $v' \neq v$.

Case II. $\langle S', (j_1, F'_{j_1}), \ldots, (j_z, F'_{j_z}), F'_a, F'_b \rangle \neq \langle S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b \rangle$ and the hash values are not the same. Unless the point **P** satisfies the hyperplane $\log_g v'$, the decryption oracle will reject. Moreover, Equations 1, 2, 4 and 7 are linearly independent by observing that

$$\det \begin{pmatrix} a & b & 0 & 0 \\ 0 & 0 & a & b \\ t_1 a & t_2 b & \alpha t_1 a & \alpha t_2 b \\ t'_1 a & t'_2 b & \alpha' t'_1 a & \alpha' t'_2 b \end{pmatrix} = a^2 b^2 (\alpha - \alpha')(t'_2 - t'_1)(t_2 - t_1) \neq 0.$$

Thus, the hyperplane and the line **L** intersect at a point. Therefore, the decryption oracle rejects the query except with a negligible probability.

Case III. $\langle S', (j_1, F'_{j_1}), \ldots, (j_z, F'_{j_z}), F'_a, F'_b \rangle \neq \langle S, (j_1, F_{j_1}), \ldots, (j_z, F_{j_z}), F_a, F_b \rangle$, but the hash values are the same, which contradicts with the assumption that H is collision-resistant.

This completes the proof. □