A Fully Public-Key Traitor-Tracing Scheme

YUH-DAUH LYUU
Dept. of Computer Science & Information Engineering and Dept. of Finance
National Taiwan University
No 1, Sec 4, Roosevelt Rd, Taipei, Taiwan
lyuu@csie.ntu.edu.tw http://www.csie.ntu.edu.tw/∼lyuu/lyuu.html

MING-LUEN WU
Dept. of Computer Science & Information Engineering
National Taiwan University
No 1, Sec 4, Roosevelt Rd, Taipei, Taiwan
d5526009@csie.ntu.edu.tw

Abstract: - We propose a fully public-key traitor-tracing scheme in which each subscriber can choose his or her own private decryption key without others learning the key. The distributor of the digital content utilizes the public data coming from all subscribers to compute a public encryption key. The paid contents are then transmitted to the subscribers, after being encrypted with the public key. Each subscriber can decrypt the data using his or her own secret key. Even if a coalition of subscribers conspire to create a pirate decoder with a tamper-free decryption key, we have a tracing algorithm to trace them. Our scheme is long-lived, which means that the subscribers' secret keys need not be regenerated after the pirate key is detected or when subscribers join or leave the system. Finally, our scheme guarantees anonymity.

Key-Words: Public-key encryption, Broadcast encryption, Traitor tracing, Semantical security, Long livedness, Anonymity.

1 Introduction

In an open broadcast network, a distributor transmits digital contents to a large number of users in such a way that only subscribers are authorized to extract the contents. Applications include such fee-based services as pay-per-view television and Web financial information channel. Clearly, anyone connected to the open network is able to pick up the data that flow through the broadcast channel, whether authorized or not. A straightforward solution to this problem is for the distributor to separately encrypt the contents with each subscriber's key before broadcasting the ciphertext. Now, only the subscribers have the corresponding private keys to decrypt the ciphertext. This issue of *secure broadcasting* is first addressed in [6]; however, the proposed method carries out n-times encryptions for one copy of data, where n is the number of subscribers. Later, *broadcast encryption* is proposed [2, 9, 12]. A broadcast-encryption scheme prevents nonsubscribers from extracting the contents. By properly generating keys, the distributor can encrypt the contents with the encryption key derived from all subscribers' secret keys, and the subscribers have the decryption key to decrypt the ciphertext. The tradeoff between the bandwidth requirement and the keys' storage space is studied in [3, 13].

A broadcast-encryption scheme remains prone to the *collusion attack*. Some subscribers may collude to create new decryption keys, and the resulting *pirate decoder* allows nonsubscribers to extract the contents. To discourage subscribers to reveal their private keys, *traitor tracing* is initiated in [7] and studied further in [10, 14, 18]. The idea is an algorithm that uses the confiscated pirate decoder to track down at least one colluder without wrongly accusing noncolluders with high probability. Most of these schemes are so-called *black-box traceable*. This means that the pirate decoder can be queried on different inputs as an oracle but cannot be opened to reveal its private key. Most of the traitor-tracing schemes are secret-key systems. Although they can be founded on public keys, complex protocols may result [14]. More recent work in [4] proposes a public-key traitor-tracing scheme; furthermore, as long as the number of colluders is below some threshold, the tracing algorithm catches all and only traitors. The scheme has two disadvantages: It is only partially black-box traceable, and the secret keys of the subscribers are generated by a trusted center (the system is hence not fully public-key). We will present a traitor-tracing scheme with these following strong features: The tracing algorithm is black-box traceable, it tracks down all the colluders regardless of their size, and the subscribers generate their own secret keys (it is thus *fully* public-key).

Key management is a critical issue. Subscribers' keys be affected for at least two reasons. First, a key must be discarded if it is found to be pirated or if its user leaves the system. But then the remaining subscribers' keys may be subject to changes when even one user's key is discarded. Second, when a new subscriber joins the system, the existing subscribers' keys may need to be changed to prevent the new subscriber from decrypting the ciphertext received before the new user joins the system. Both scenarios become problematic if pirating is frequent or if the subscriber base is fluid. In order for the subscribers to easily manage their own keys, the system should minimize the need to regenerate subscribers' secret keys. Such a scheme is said to be *long-lived*. This attribute is studied in [11]. In that proposed scheme, some subscribers may need to be rekeyed when a sufficient number of keys are discarded. In contrast, our scheme does not require the regeneration of the secret keys in the above two scenarios. It is therefore *perfectly* long-lived.

Anonymity is another critical issue for any traitor-tracing schemes because the promise of anonymity usually promotes subscription [12]. We list two cases which may compromise anonymity. When new subscribers join the service, their identities may be revealed because of their interaction with the distributor. Second, the broadcast contents themselves may disclose the subscribers' identities, which makes eavesdropping threaten the privacy of the subscribers. Our scheme solves both problems: Registering with the service is noninteractive, and analyzing the transmissions does not reveal the subscribers' identities. Thus, the subscribers' identities are not known to anyone except the distributor.

We now summarize the features of our traitor-tracing scheme. The traitor-tracing scheme is perfectly long-lived and achieves anonymity. It is a fully public-key system, without relying on a trusted center to generate the keys. The scheme is based on the following ideas. Each subscriber randomly selects a secret key to compute a number which is sent to the distributor. After the distributor receives the numbers from all the subscribers, it combines them to create a single encryption key. Using the ElGamal encryption scheme, digital contents are encrypted with the encryption key. Henceforth, each subscriber uses his or her own secret key to decrypt the ciphertext.

This paper is organized as follows. In Section 2, basic terms are defined. Then in Section 3, useful facts in number theory are presented. Section 4 describes our scheme and discusses its security. Important attributes of our scheme are analyzed in Section 5. Conclusions are given in Section 6.

2 Key Terms

Broadcast encryption consists of three components: key generation, encryption, and decryption. It specifies the way to encrypt and decrypt the digital contents with the generated keys in a broadcast network. The important task facing broadcast encryption is to prevent nonsubscribers from decrypting the encrypted content. Like the point-to-point encryption scheme, a broadcast-encryption scheme can be secret-key or public-key. Traitors are subscribers who allow nonsubscribers to extract the contents. When a broadcast-encryption scheme has the capability to track down traitors, it is said to be traitor-tracing. Traitor tracing involves a *tracing algorithm*, which uses the confiscated pirate decoder to track down traitors. The tracing algorithm is *k-traceable* if at least one of the traitors can be identified given any pirate decoder created by at most k traitors. The tracing algorithm is said to be black-box if the pirate decoder can only be queried as an oracle but not opened to reveal its internal key. The perfect long-livedness attribute means that rekeying the subscribers is unnecessary in the following two situations:

1. An existing key is discarded to prevent its future use.
2. New subscribers join the system without being able to decrypt earlier ciphertext.

The anonymity attribute means that the subscribers' identities are not known to anyone except the distributor. Our proposed traitor-tracing scheme will be perfectly long-lived and anonymous.

A public-key traitor-tracing scheme consists of four components [4]:

1. **Key generation:** Given a security parameter s and a number n, the key-generation algorithm outputs a public encryption key e and n private decryption keys $d_1, d_2, \ldots, d_n$. Any decryption key d_i can be used to decrypt a ciphertext created with the public encryption key e.
2. **Encryption:** Given a public key e and a message x, the encryption algorithm outputs a ciphertext C.
3. **Decryption:** Given a ciphertext C and any decryption key d_i, the decryption algorithm outputs the message x. The algorithm itself is public. Only the decryption keys are secret as in the point-to-point cryptosystem.
4. **Tracing:** Given a pirate decryption box $\mathcal{D}$, the tracing algorithm outputs at least one traitor if at most k of the decryption keys are involved in creating the box. The tracing algorithm may be black-box or otherwise.

A fully public-key traitor-tracing scheme strengthens the key-generation part as follows:

- **Key generation*:** Given a security parameter s, the n private decryption keys $d_1, d_2, \ldots, d_n$ are generated at random. The corresponding public

key e_i for each d_i is computed. The public encryption key e depends on $e_1, e_2, \ldots, e_n$. Any decryption key d_i can be used to decrypt a ciphertext created with the public encryption key e.

The secret keys are now known only to their subscriber owners. This is clearly desirable.

3 Number-theoretic Preliminaries

We next present some number-theoretical results. See [15, 16] for additional information. Let $\phi(n)$ denotes Euler's phi function, which gives the number of positive integers $m \in \{1, 2, \ldots, n-1\}$ such that $\gcd(m, n) = 1$.

Lemma 3.1. *If q and $p = 2q + 1$ are both primes and a is a positive integer with $1 < a < p - 1$, then $-a^2$ is a quadratic nonresidue and a primitive root modulo p.*

Proof. There are $\phi(2q) = q - 1$ primitive roots of p, q quadratic residues of p, and q quadratic nonresidues of p. None of the q quadratic residues of p can be primitive roots. Furthermore, -1, a quadratic nonresidue, cannot be a primitive root either. Thus, the remaining $q-1$ quadratic nonresidues of p must be all the primitive roots. The number $-a^2$ is a quadratic nonresidue of p because

$$\left(\frac{-a^2}{p}\right) = \left(\frac{-1}{p}\right)\left(\frac{a^2}{p}\right) = (-1) \times 1 = -1.$$

Hence $-a^2$ is a primitive root modulo p. □

Fact 3.1 (Chinese remainder theorem). *Let $m_1, m_2, \ldots, m_n$ be positive integers that are relatively prime in pairs. Then, for any given integers $b_1, b_2, \ldots, b_n$, the system of congruences*

$$x \equiv b_i \pmod{m_i}, 1 \le i \le n$$

has a unique solution modulo $M = m_1 m_2 \cdots m_n$. The solution is given by $x = \sum_{i=1}^{n} b_i M_i y_i \bmod M$, where $M_i = M/m_i$ and $M_i y_i \equiv 1 \pmod{m_i}$.

Assume s is the length of every modulus. The computational complexity of applying the Chinese remainder theorem is $O_B(M(sn)\log n) + O_B(nM(s)\log s)$, where $M(x)$ is the time of multiplying two x bit integers, and O_B indicates order of magnitude in bit operations [1, Theorem 8.21].

Fact 3.2. *Simultaneous congruences*

$$x \equiv b_i \pmod{m_i}, 1 \le i \le n$$

have a solution if and only if $\gcd(m_i, m_j)$ divides $b_i - b_j$ for all pairs of integers (i, j) with $1 \le i < j \le n$, in which case the solution is unique modulo $M = \mathrm{lcm}(m_1, m_2, \ldots, m_n)$ and is given by the Chinese remainder theorem with the said M.

Let $\gcd(g, n) = 1$. The least positive integer d such that $g^d \equiv 1 \pmod{n}$ is called the *order* of g modulo n, and is denoted by $\mathrm{ord}_n g$. A *universal exponent* of n is a positive integer u such that $g^u \equiv 1 \pmod{n}$ for all g relatively prime to n. The minimal universal exponent of n is denoted by $\lambda(n)$.

Fact 3.3. *Let M be a positive integer with odd prime factorization $M = p_1 p_2 \cdots p_n$.* (1)

$$\lambda(M) = \mathrm{lcm}(\phi(p_1), \phi(p_2), \ldots, \phi(p_n)).$$

(2) *There exists an integer g such that $\mathrm{ord}_M g = \lambda(M)$, the largest possible order of an integer modulo M.* (3) *Let r_i be a primitive root modulo p_i. The solution of simultaneous congruences $x \equiv r_i \pmod{p_i}$, $i = 1, 2, \ldots, n$, produces such an integer g.*

The above fact implies that if M is a product of large primes $p_i = 2q_i + 1$ where q_i are also primes, then there exists a g whose order contains large prime factors. The reason is that

$$\lambda(M) = \mathrm{lcm}(p_1 - 1, p_2 - 1, \ldots, p_n - 1) = 2q_1 q_2 \cdots q_n. \quad (1)$$

Fact 3.4. *Suppose that g and n are relatively prime with $n > 0$. Then $g^i \equiv g^j \pmod{n}$ if and only if $i \equiv j \pmod{\mathrm{ord}_n g}$.*

4 The Public-key Traitor-Tracing Scheme

Let s be a security parameter and n be the number of subscribers. Choose s-bit primes $p_i = 2q_i + 1$, where q_i are odd primes. For convenience, assume $p_1 < p_2 < \cdots < p_n$. The primes will be such that $q_1^{1/2}$ is large enough. Let $M = \prod_{i=1}^{n} p_i$. Assume the contents (plaintext) are elements of Z_{p_1}. Let g be a *common* primitive root modulo each of the p_i's. By Lemma 3.1 such a g exists because any $-a^2$ is a valid candidate when $1 < a < p_1 - 1$. We now describe the four components of our scheme.

1. **Key generation:** Each subscriber i chooses a private decryption key $d_i \in Z_{p_i}^*$ randomly and sends (β_i, p_i) to the distributor, where $\beta_i = g^{d_i} \bmod p_i$. Next, the distributor computes $\beta = \sum_{i=1}^{n} \beta_i M_i y_i \bmod M$, where $M_i = M/p_i$ and $M_i y_i \equiv 1 \pmod{p_i}$, for $1 \le i \le n$. The triple (g, β, M) is the public encryption key. Note that $\beta \equiv \beta_i \pmod{p_i}$.
2. **Encryption:** Let the plaintext be $x \in Z_{p_1}$. The distributor picks a random element r from $\{0, 1, \ldots, p_1 - 1\}$ and computes

$$z_1 = g^r \bmod M,$$
$$z_2 = x\beta^r \bmod M.$$

The ciphertext is $C = (z_1, z_2)$. Note that the digital content is encrypted only once not n times.

3. **Decryption:** Given a ciphertext $C = (z_1, z_2)$, the decryption algorithm computes the plaintext $x = z_2(z_1^{d_i})^{-1} \bmod p_i$.
4. **Tracing:** The tracing algorithm is described in Section 5.

The encryption and decryption algorithms are similar to those in the ElGamal cryptosystem. The content is encrypted once, and each subscriber can decrypt using his or her own decryption key. There is no need of a key generation center to generate the decryption keys. Instead, subscribers generate their own keys at random. The distributor creates the encryption key without knowing any decryption key. The decryption keys are therefore private to their respective owners. This is identical to the standard point-to-point public-key cryptosystem setup. Our scheme is hence fully public-key.

Consider this straightforward approach to secure broadcasting in a full public-key setting [6]: Before broadcasting, the plaintext is encrypted n times with all subscribers' public keys, and then the Chinese Remainder algorithm is applied. If the straightforward approach uses the ElGamal probabilistic encryption, the number of modular multiplications and squarings in performing encryptions is n-times ours because we encrypt the plaintext only once. The fast modular multiplication algorithm devised in [5] requires $t + 7$ clock pluses, where t is the length of the modulus. Assume our encryption needs a total of ℓ modular multiplications and squarings. For each plaintext, our encryption needs time $\ell(ns + 7)$, but the straightforward approach needs time $\ell n(s + 7)$ plus $O_B(M(ns) \log n) + O_B(nM(s) \log s)$ for performing the Chinese Remainder algorithm. Hence, our scheme is more efficient.

4.1 Security Analysis

We will show that our scheme is plaintext-secure against a passive generic adversary if $q_1^{1/2}$ is large enough. We now explain the terms. A *passive generic adversary* is a generic algorithm and can only eavesdrop the network. A *generic algorithm* does not exploit any special properties of the encodings of group elements except that each group element is encoded as a unique bit string [17]. An encryption system is *plaintext-secure* if the full plaintext about a content cannot be derived from its encryption form.

Let M' be a product of k primes p_i, say,

$$M' = p_{i_1} p_{i_2} \cdots p_{i_k}.$$

Let $\beta' = \beta \bmod M'$. As before, g is a common primitive root modulo p_i, $i = 1, 2, \ldots, n$. The *Diffie-Hellman problem* (DH) in a group with generator g is to compute $g^{r_1 r_2}$ from g^{r_1} and g^{r_2}. Shoup shows that any generic algorithm must perform $\Omega(q_{\max}^{1/2})$ group operations for the problem, where $q_{\max}$ is the largest prime dividing the order of the group [17]. Because g is a common primitive root modulo each of the p_is, the largest possible order of g modulo M' is $\lambda(M') = 2q_{i_1} q_{i_2} \cdots q_{i_k}$ by equation (1). So the subgroup H of $Z_{M'}^*$ generated by g has order $\lambda(M')$. The order of H surely contains a prime factor which is not smaller than q_1. As we choose a large $q_1^{1/2}$, the DH problem in H is intractable for a generic adversary. Based on this intractability, we next prove that our encryption scheme is plaintext-secure against a passive generic adversary. But first we need the following lemma.

Lemma 4.1. *If the ElGamal cryptosystem in $Z_{M'}^*$ can be broken, then the Diffie-Hellman problem in the subgroup H of $Z_{M'}^*$ generated by g can be solved efficiently.*

Proof. Suppose that there is an algorithm $\mathcal{A}$ that breaks the ElGamal cryptosystem in $Z_{M'}^*$. Given g, M', β', z_1 and z_2, algorithm $\mathcal{A}$ computes the plaintext $x = z_2(\beta'^{\log_g z_1})^{-1} \bmod M'$.
Assume that $\beta', \gamma \in H$. When given inputs g, M', β' and γ for the Diffie-Hellman problem, $\mathcal{A}$ can be invoked to solve this DH problem by

$$\begin{aligned} \mathcal{A}(g, M', \beta', \gamma, 1)^{-1} &= ((\beta'^{\log_g \gamma})^{-1})^{-1} \bmod M' \\ &= g^{\log_g \beta' \log_g \gamma} \bmod M'. \end{aligned}$$

□

Theorem 4.1. *Our encryption scheme is plaintext-secure against a passive generic adversary.*

Proof. A passive adversary can only eavesdrop to get $C = (z_1, z_2)$. For such an adversary, to derive plaintext is equivalent to breaking the ElGamal encryption scheme in $Z_{M'}^*$. Nevertheless, by Lemma 4.1, breaking the ElGamal scheme in $Z_{M'}^*$ implies solving the DH problem in the subgroup H of $Z_{M'}^*$ generated by g. For a generic algorithm, solving the DH problem for the subgroup H needs at least $\Omega(q_1^{1/2})$ time. Because $q_1^{1/2}$ is chosen to be large, the theorem is proved. □

4.2 Semantic Security

The security of our scheme is based on the ElGamal encryption scheme in $Z_{M'}^*$. To enhance the security, we show how to choose the generator of subgroups and limit the message so that our scheme is *semantically secure*.

Because g is the common generator of the $Z_{p_i}^*$'s, g^2 has order q_i and generates all the q_i quadratic residues in $Z_{p_i}^*$ for each $i = 1, 2, \ldots, n$. Similarly, g^2 has order $\lambda(M')/2 = q_{i_1} q_{i_2} \cdots q_{i_k}$ and generates all the quadratic residues in $Z_{M'}^*$. So the cyclic subgroup of $Z_{M'}^*$ generated by g^2 has order $q_{i_1} q_{i_2} \cdots q_{i_k}$, each of whose prime factors is at least q_1. The *decision Diffie-Hellman problem* (DDH) in group G generated by h with a large order is to efficiently distinguish the two distributions: (h^{r_1}, h^{r_2}, h^z) where r_1, r_2, z are random and

$(h^{r_1}, h^{r_2}, h^{r_1 r_2})$ where r_1, r_2 are random. Any generic algorithm must perform $\Omega(q_{\min}^{1/2})$ group operations for the DDH problem, where $q_{\min}$ is the smallest prime dividing the order of the group [17]. Modify the parameters in our scheme: Let the common generator be g^2 and assume the plaintext (contents) in Z_{p_1} must be a common quadratic residue of all the p_i's. Because $q_1^{1/2}$ is chosen to be large, the DDH problem for the subgroup generated by g^2 is intractable for a generic adversary. Based on this intractability, our scheme can be shown to be semantically secure by using a similar result in [19] after replacing the modulus p there with our M'.

The parameters were modified so that each plaintext $x \in Z_{p_1}$ is a common quadratic residue of all the p_i's. That is very inconvenient. Here is an alternative. Encrypt x^2 instead of x. After decryption, subscriber i obtains $x' = x^2 \bmod p_i$. As $p_i \equiv 3 \pmod 4$, the solutions to $x^2 \equiv x' \pmod{p_i}$ are $x \equiv \pm x'^{(p_i+1)/4} \pmod{p_i}$. Note that one of the solutions is odd and the other even. If we always pad one bit in the least significant bit of x to make it, say, odd, the plaintext can be uniquely decided. So we have the following theorem.

Theorem 4.2. *Our scheme modified as described above is semantically secure against a passive generic adversity.*

4.3 Forgery of Decryption Keys

Recall that $\beta, M, \beta_i, p_i, g$, and d_i, for $i = 1, 2, \ldots, n$, are all the keys in the system, among which only d_i are not public. The ciphertext (z_1, z_2) is also public. Because our encryption scheme is plaintext-secure, it is impossible to fake subscribers' keys or create new decryption keys from the public information. This implies that combining d_i's is the only way to create decryption keys. Suppose k of the d_i's are used to create a new decryption key d_H, say $d_1, d_2, \ldots, d_k$. Because $d_1, d_2, \ldots, d_k$ are involved in creating d_H, we solve d_H from

$$g^{d_H} \equiv g^{d_i} \pmod{p_i} \tag{2}$$

for $i = 1, \ldots, k$. Let $M'_H = p_1 p_2 \cdots p_k$. The following lemma proves that this d_H works.

Lemma 4.2. *Suppose that $d_1, d_2, \ldots, d_k$ are used to create a new decryption key d_H. Then d_H exists and equals $\sum_{i=1}^{k} d_i M_{H_i} y_i \bmod M_H$ if and only if $\gcd(p_i - 1, p_j - 1)$ divides $d_i - d_j$, where $M_H = \operatorname{lcm}(p_1 - 1, p_2 - 1, \ldots, p_k - 1)$, $M_{H_i} = M_H/(p_i - 1)$, and $M_{H_i} y_i \equiv 1 \pmod{p_i - 1}$ for $i = 1, 2, \ldots, k$.*

Proof. By equation (2), d_H satisfies $g^{d_H} \equiv \beta \pmod{M'_H}$. Hence d_H and M'_H can be used to decrypt the ciphertext. By Fact 3.4, $g^{d_H} \equiv g^{d_i} \pmod{p_i}$ implies $d_H \equiv d_i \pmod{p_i - 1}$. The rest of the lemma follows from Fact 3.2. □

The next theorem is immediate.

Theorem 4.3. *For a passive generic adversary, the only way to create a new decryption key d_H in our scheme is to combine the d_i's in the way mentioned in Lemma 4.2.*

5 Traceability, Long-Livedness, Anonymity

Traceability. The tracing algorithm shall utilize the pirate decoder to apprehend the traitors. First, assume that the pirate decryption key d_H and M'_H are revealed by opening the decoder box to simplify the analysis. The tracing algorithm uses M'_H to detect all traitors as follows: If $M'_H \equiv 0 \pmod{p_i}$, then subscriber i is a traitor; otherwise, he is innocent. Thus all traitors will be captured, and innocent subscribers will not be accused. Because p_i are public, the tracing algorithm does not need the private keys of subscribers to succeed.

Now suppose that the decoder cannot be opened (it is a black box). Then the black-box tracing algorithm performs the following steps for each $i = 1, 2, \ldots, n$ to trace all traitors.

Step 1: Compute $\alpha_i = \beta \bmod M_i$, where $M_i = M/p_i$, for $M = p_1 p_2 \cdots p_n$.

Step 2: Choose a plaintext x. Compute the ciphertext C with the public key (α_i, g, M_i).

Step 3: Feed C to the black-box decoder. If the output is not equal to x, then subscriber i is a traitor.

The tracing algorithm flags the subscribers whose moduli are the prime factors of M'_H. Thus the tracing algorithm does track down all and only traitors. Of course, its performance suffers from not being able to open the box.

Perfectly long-lived keys. When an existing decryption key is discarded or when a new subscriber joins the system, the system shall not require the other subscribers to perform any interaction with the distributor to change their secret keys. The public encryption key will be modified to achieve this goal. Assume that (β, g, M) is the original public encryption key.

Suppose that subscriber i's key is disabled because he is a traitor or he wants to leave the system. Then the new public encryption key is

$$(\hat{\beta}, g, \hat{M}) = (\beta \bmod (M/p_i), g, M/p_i).$$

No rekeying is required for the remaining subscribers. By disabling a traitor's key, the pirate decoder using that key also becomes useless.

Now suppose that a new subscriber $n + 1$ joins the system. Then the new public encryption key is

$$\begin{aligned}&(\hat{\beta}, g, \hat{M})\\ =&(\beta p_{n+1} v + \beta_{n+1} M w \bmod M p_{n+1}, g, M p_{n+1}),\end{aligned}$$

where p_{n+1} is a new prime distinct from $p_1, p_2, \ldots, p_n$, $p_{n+1}v \equiv 1 \pmod{M}$, and $Mw \equiv 1 \pmod{p_{n+1}}$. Existing subscribers do not need to update their secret keys, and the new subscriber cannot decrypt the contents received before he joins. After the public encryption key has been changed, the traitor-tracing scheme still satisfies the same properties. The following theorem demonstrates that our method obtains the correct encryption key.

Theorem 5.1. *Let $p_1, p_2, \ldots, p_{n+1}$ be distinct primes. Suppose that for any integers $\beta_1, \beta_2, \ldots, \beta_n$, the system of congruences*

$$x \equiv \beta_i \pmod{p_i}, \quad i = 1, 2, \ldots, n,$$

has a unique solution β modulo $M = p_1 p_2 \cdots p_n$. Then $\beta' = \beta \bmod M_n$, where $M_n = M/p_n$, is the unique solution modulo M_n to the system of congruences

$$x \equiv \beta_i \pmod{p_i}, \quad i = 1, 2, \ldots, n-1. \tag{3}$$

Furthermore, $\beta'' = \beta p_{n+1} v + \beta_{n+1} M w \bmod M''$ is the unique solution modulo M'' to the system of congruences

$$x \equiv \beta_i \pmod{p_i}, \quad i = 1, 2, \ldots, n+1,$$

where $M'' = M p_{n+1}$, $p_{n+1}v \equiv 1 \pmod{M}$, and $Mw \equiv 1 \pmod{p_{n+1}}$.

Proof. Both β and β' are solutions to congruences (3). As the solutions are congruent modulo M_n, we have $\beta' = \beta \bmod M_n$. That β'' is the desired solution follows from Fact 3.2. □

Anonymity. In our scheme, a subscriber registers by sending his or her own public information to the distributor. This part is noninteractive. Because all the subscribers receive the same ciphertext, the broadcast message need no addressing. Even when one obtains the plaintext and the public key of another subscriber, the subscriber's identity remains hidden. As the encryption scheme is probabilistic, encrypting a plaintext with anyone's public keys and then comparing the resulting ciphertext with any historical ciphertext for clues is wasted efforts. Our scheme is hence anonymous.

6 Conclusions

In order to prevent others from learning the secret keys, we propose a fully public-key traitor-tracing scheme. Perfect long-livedness and anonymity are achieved. Furthermore, it is a simple task to recompute the encryption key if needed. By the choice of parameters, our scheme can be plaintext-secure or semantically secure against a passive generic adversary. The tracing algorithm is n-traceable and captures all and only traitors. This holds even if the pirate decoder is a black box.

References:

[1] A. V. Aho, J. E. Hopcroft and J. D. Ullman. *The Design and Analysis of Computer Algorithm.* Addison-Wesley, 1974.

[2] S. Berkovits. "How to broadcast a secret." In *Proc. Eurocrypt'91*, pp. 535–541.

[3] C. Blundo, L. A. F. Mattos and D. R. Stinson. "Trade-offs between communication and storage in unconditionally secure systems for broadcast encryption and interactive key distribution." In *Proc. Crypto'96*, pp. 387–400.

[4] D. Boneh and M. Franklin. "An efficient public key traitor tracing scheme." In *Proc. Crypto'99*, pp. 338–353.

[5] E. F. Brickell. "A fast modular multiplication algorithm with application to two key cryptography." In *Proc. Crypto'82*, pp. 51–60.

[6] C. H. Chiou and W. T. Chen. "Secure broadcasting using the secure lock." *IEEE Trans. on Software Engineering*, 15, No. 8 (August 1989), 929–934.

[7] B. Chor, A. Fiat and M. Naor. "Tracing traitors." In *Proc. Crypto'94*, pp. 257–270. Final version with B. Pinkas in *IEEE Trans. on Information Theory*, 46, No. 3 (May 2000), 893–910.

[8] T. ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms." *IEEE Trans. on Information Theory*, 31, No. 4 (July 1985), 469–472.

[9] A. Fiat and M. Naor. "Broadcast encryption." In *Proc. Crypto'93*, pp. 480–491.

[10] E. Gafni, J. Staddon and Y. L. Yin. "Efficient methods for integrating traceability and broadcast encryption." In *Proc. Crypto'99*, pp. 372–387.

[11] J. Garay, J. Staddon and A. Wool. "Long-lived broadcast encryption." In *Proc. Crypto 2000*, pp. 333-352.

[12] M. Just, E. Kranakis, D. Krizanc and P. van Oorschot. "On key distribution via true broadcasting." In *Proc. of 2nd ACM Conference on Computer and Communications Security*, 1994, pp. 81–88.

[13] M. Luby and J. Staddon. "Combinatorial bounds for broadcast encryption." In *Proc. Eurocrypt'98*, pp. 512–526.

[14] B. Pfitzmann. "Trials of traced traitors." In *Proc. Information Hiding Workshop*, 1996, pp. 49–64.

[15] K. H. Rosen. *Elementary Number Theory and Its Applications.* Addison-Wesley, 1988.

[16] H. N. Shapiro. *Introduction to the Theory of Numbers.* John Wiley & Sons, 1983.

[17] V. Shoup. "Lower bounds for discrete logarithms and related problems." In *Proc. Eurocrypt'97*, pp. 256–266.

[18] D. R. Stinson and R. Wei. "Key preassigned traceability schemes for broadcast encryption." In *Proc. SAC'98*, pp. 144–156.

[19] Y. Tsiounis and M. Yung. "On the security of ElGamal based encryption." In *Proc. PKC'98*, pp. 117–134.