

國立交通大學

資訊科學與工程研究所

碩士論文

對數儲存量及常數計算量的
公開金鑰廣播加密系統



Efficient Public Key Broadcast Encryption with
Logarithmic Key Size and Constant Decryption Time

研究生：沈宣佐

指導教授：曾文貴 教授

中華民國九十七年六月

對數儲存量及常數計算量的公開金鑰廣播加密系統
Efficient Public Key Broadcast Encryption with Logarithmic Key Size
and Constant Decryption Time

研究生：沈宣佐

Student：Shiuan-Tzuo Shen

指導教授：曾文貴

Advisor：Wen-Guey Tzeng

國立交通大學
資訊科學與工程研究所
碩士論文



Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年六月

國立交通大學

博碩士論文全文電子檔著作權授權書

(提供授權人裝訂於紙本論文書名頁之次頁用)

本授權書所授權之學位論文，為本人於國立交通大學資訊科學與工程研究所 資訊系統 組，96 學年度第 2 學期取得碩士學位之論文。

論文題目：對數儲存量及常數計算量的公開金鑰廣播加密系統
指導教授：曾文貴

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

論文全文上載網路公開之範圍及時間：

本校及台灣聯合大學系統區域網路	<input checked="" type="checkbox"/> 立即公開
校外網際網路	<input checked="" type="checkbox"/> 立即公開

■ 全文電子檔送交國家圖書館

授權人：沈宣佐

親筆簽名：沈宣佐

中華民國 九十七 年 七 月 二十二 日

國立交通大學

博碩士紙本論文著作權授權書

(提供授權人裝訂於全文電子檔授權書之次頁用)

本授權書所授權之學位論文，為本人於國立交通大學資訊科學與工程研究所 資訊系統 組，96 學年度第 2 學期取得碩士學位之論文。

論文題目：對數儲存量及常數計算量的公開金鑰廣播加密系統
指導教授：曾文貴

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學，基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學圖書館得以紙本收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行閱覽或列印。

本論文為本人向經濟部智慧局申請專利(未申請者本條款請不予理會)的附件之一，申請文號為：_____，請將論文延至____年____月____日再公開。

授權人：沈宣佐

親筆簽名：沈宣佐

中華民國 九十七 年 七 月 二十二 日

國家圖書館

博碩士論文電子檔案上網授權書

(提供授權人裝訂於紙本論文本校授權書之後)

ID:GT009555560

本授權書所授權之論文為授權人在國立交通大學資訊科學與工程研究所 96 學年度第 2 學期取得碩士學位之論文。

論文題目：對數儲存量及常數計算量的公開金鑰廣播加密系統
指導教授：曾文貴

茲同意將授權人擁有著作權之上列論文全文(含摘要)，非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授權人：沈宣佐

親筆簽名：沈宣佐

民國 九十七 年 七 月 二十二 日

國立交通大學

研究所碩士班

論文口試委員會審定書

本校 資訊科學與工程 研究所 沈宣佐

君

所提論文：

對數儲存量及常數計算量的公開金鑰廣播加密系統

合於碩士資格水準、業經本委員會評審認可。

口試委員：

蔡錫爵 常之貴

呂及人 孫宏民

指導教授：

常之貴

所

長：

常之貴

中華民國九十七年六月二十七日

Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

As members of the Final Examination Committee, we certify that
we have read the thesis prepared by Shiuan-Tzuo Shen
entitled Efficient Public Key Broadcast Encryption with
Logarithmic Key Size and Constant Decryption Time

and recommend that it be accepted as fulfilling the thesis
requirement for the Degree of Master of Science.

Committee Members:

Shu-chu Tzeng Wen-Guey Tseng

Chin-Jon Lin Hung-Min Su

Thesis Advisor: Wen-Guey Tseng

Director: Wen-Guey Tseng

Date: 2008/06/27

摘要

我們提出一個完全抵禦共謀的公開金鑰廣播加密系統，達到 $O(1)$ 公開金鑰量、 $O(\log n)$ 私密金鑰量、 $O(n)$ 密文大小、 $O(1)$ 解密時間，其中 n 為使用者的數量， r 為非合法接收者的數量。據我們所知，我們的系統是目前最有效率的公開金鑰廣播加密系統。在 random oracle 的模型下，本系統同樣達到了 IND-CCA2 的安全性。我們的系統是建立在 [LT08] 的架構之上，並應用了 [Boy07] 所提出的方法。我們的主要貢獻在於提出了一套金鑰衍生的方法，使得使用者所需要儲存的私密金鑰量降為 $O(\log n)$ ，改進了 [LT08] 使用者需要儲存 $O(\log^2 n)$ 的私密金鑰。我們應用 [Boy07] 的方法使得本系統的安全性達到 IND-CCA2，並且不須額外的成本負擔。



關鍵字：廣播加密、Lagrange 內插法、金鑰衍生

Abstract

We propose a fully collusion resistant public key broadcast encryption scheme that achieves $O(1)$ public key size, $O(\log n)$ private key size, $O(r)$ ciphertext size, and $O(1)$ decryption time where n is the number of users in the system and r is the number of the revoked users. To the best of our knowledge, our scheme is the most efficient scheme in the existing broadcast encryption schemes. Our scheme also achieves the IND-CCA2 security in the random oracle model. It is based on the idea of [LT08] and the result of [Boy07]. We provide a key derivation method that reduces the private key size to $O(\log n)$ while [LT08] is $O(\log^2 n)$. We apply the method of [Boy07] to enhance the security to IND-CCA2 without redundancy.

Keyword: Broadcast Encryption, Lagrange Interpolation, Key Derivation



Contents

1	Introduction	1
1.1	Related Work	2
1.2	Our Contribution	5
2	Preliminary	6
2.1	Bilinear Map	6
2.2	Gap-BDH Assumption	6
2.3	Broadcast Encryption	7
3	Idea of SD	9
4	Idea of PK-SD	14
5	Idea of PK-SD-PI	16
6	Our Scheme	20
6.1	Construction	25
7	Security	30
8	Remark	37
9	Conclusion	38



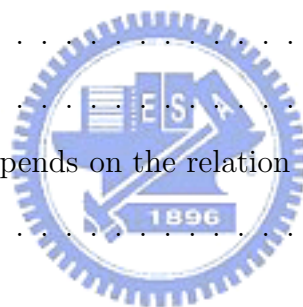
List of Tables

1	Comparison of some broadcast encryption schemes	5
---	---	---



List of Figures

1	Tree-based structure	9
2	Subset difference	10
3	Partition of receivers	10
4	Key storage	12
5	Hash chain method	13
6	Naming system	15
7	Polynomials of subtrees	17
8	PK-SD-PI	19
9	Key derivation	21
10	Polynomial derivation	21
11	Assignment of h and h'	23
12	Polynomial derivation depends on the relation	24
13	The notation $I_{i,j}^v$	26



1 Introduction

Broadcast encryption is an encryption scheme useful in the multi-receiver scenario. There are many applications of broadcast encryption, such as pay-TV programs, private communication between a group of users, etc. Assume that there are totally n users in the system and we want to send a message to a group of m users in the system. If we use general public key encryption systems (not broadcast encryption), we need to encrypt the message m times (encrypted with each of the receiver's public key) and send them to the receivers. It needs much network bandwidth. There is another way to send a message to the m receivers. For each subset of the n users, we assign a public key to the subset respectively. Each user holds the private keys for the public keys of the subsets where he is in. Then we can use the public key of the set of the m receivers to encrypt the message and send the encrypted message to the receivers. But for each user, there are 2^{n-1} subsets where he is in. Thus each user needs to store $O(2^n)$ private keys. It is not efficient enough for practical usage, because network bandwidth is expensive and storing an exponential number of secret keys is quite inconvenient.

Broadcast encryption is a solution of striking a balance between bandwidth and storage. In a public key broadcast encryption scheme, a trusted third party sets up the whole system. It generates the system public key and the secret keys for the users. To send a message to some users, we encrypt the message with the system public key and generate the necessary information for the receivers to derive the secret key for decryption. Then we just broadcast the ciphertext and the necessary information to all users. Only the valid receivers can derive the secret key for decryption from the necessary information and their own secret keys, then they can decrypt the message correctly with the derived secret key. But the other users can not since that the information is not sufficient enough for them to derive the secret key and decrypt the message.

Broadcast encryption schemes can be symmetric or asymmetric. It depends on the type of encryption function used to encrypt the message. For a symmetric broadcast encryption scheme, only the users in the system can broadcast messages since the senders must know the secret keys. While for an asymmetric broadcast encryption scheme, everyone (even not in the system) can broadcast messages to the users in the system with the system public key. Broadcast encryption also can be stateful or stateless. For a stateful broadcast encryption scheme, the secret key of each user will be updated periodically. Thus the structure of users can be dynamic, users can join to or quit from the system if needed. While for a stateless broadcast encryption scheme, the secret key of each user won't be updated. Thus the structure of users must be fix after the setup of the whole system. Here we deal with the stateless public key broadcast encryption scheme in this paper.

1.1 Related Work

In 1993, Fiat and Naor [FN93] firstly gave a formal definition of broadcast encryption. They defined the ciphertext in the form of $\langle Hdr(S, K), E_K(M) \rangle$, where $Hdr(S, K)$ is the header about the set S of receivers, the session key K is used for encryption, and $E_K(M)$ is the symmetric encryption of the message M with K . Only the users in S can gain the session key K from $Hdr(S, K)$ and then decrypt the message M from $E_K(M)$, while other users can not. We say that a broadcast encryption scheme is fully collusion resistant if the users not in S can not gain the session key K from $Hdr(S, K)$ even they all collude together. The header size is usually dominated by the number of partitions of the receivers, and the efficiency of a broadcast encryption is usually measured by the size of secret key, the size of header, and the time of decryption.

In 2001, D. Naor, M. Naor, and Lotspiech [NNL01] proposed a *Subset-Cover* framework and two broadcast encryption schemes under the framework. Let n be the number of total users and r be the number of the revoked users. The first scheme is called *Complete Subtree*

(CS), and it achieves $O(\log n)$ secret key size, $O(r \log \frac{n}{r})$ header size, and $O(\log \log n)$ decryption time. The second scheme is called *Subset Difference* (SD), it achieves $O(\log^2 n)$ secret key size, $O(r)$ header size, and $O(\log n)$ decryption time. CS and SD are very important in the history of broadcast encryption, on which many broadcast encryption schemes are based.

In 2003, Dodis and Fazio [DF03] used Identity Based Encryption (IBE) and Hierarchical Identity Based Encryption (HIBE) as building blocks to construct two public key broadcast encryption schemes based on the idea of CS and SD respectively. The two schemes are called PK-CS and PK-SD respectively. PK-CS achieves $O(1)$ public key size, $O(\log n)$ secret key size, $O(r \log \frac{n}{r})$ header size, and $O(1)$ decryption time. PK-SD achieves $O(1)$ public key size, $O(\log^2 n)$ secret key size, $O(r)$ header size, and $O(\log n)$ decryption time.

In 2004, Yoo, Jho, Cheon, and Kim [YJCK04] proposed a symmetric broadcast encryption scheme based on the idea of Naor and Pinkas [NP01]. They used the idea of user set partition and multiple polynomials interpolation to construct a broadcast encryption scheme that is more efficient than SD. The scheme achieves $O(\log \frac{n}{r})$ secret key size and $O(\alpha r + m)$ header size, where $1 < \alpha < 2$ is a predetermined system parameter.

In 2005, Boneh, Gentry, and Waters [BGW05] proposed a fully collusion resistant public key broadcast encryption scheme with short ciphertext and short private key. The scheme achieves $O(1)$ secret key size and $O(1)$ header size. But the public key size is $O(n)$ and the decryption time is $O(n - r)$. Boneh, Gentry, and Waters also proposed a trade-off between the public key size and the header size. For example, a fully collusion resistant public key broadcast encryption scheme with $O(\sqrt{n})$ public key size, $O(1)$ secret key size, $O(\sqrt{n})$ header size, and $O(\sqrt{n})$ decryption time.

In the same year, Hwang and Lee [HL06] used one-way hash functions as the building block to construct a symmetric broadcast encryption with logarithm secret key size. The scheme achieves $O(\log n)$ secret key size and $O(r)$ header size. But their decryption time is $O(n^\beta)$, where $0 < \beta \leq 1$ is a system parameter.

In 2007, Liu and Tzeng [LT08] proposed three fully collusion resistant broadcast encryption schemes based on the idea of polynomial interpolation. The first scheme is called BE-PI. Each user holds a share of the polynomial. When broadcasting the message, the shares of revoked users are broadcasted. Thus the valid receivers can obtain enough shares to recover the necessary information for decrypting the message, while the revoked users can not have enough shares from the broadcasted messages. BE-PI achieves $O(1)$ public key size, $O(\log n)$ secret key size, $O(r)$ header size, and $O(r)$ decryption time. The second scheme is called PK-SD-PI, and it is based on the scheme PK-SD. PK-SD-PI achieves $O(1)$ public key size, $O(\log^2 n)$ secret key size, $O(r)$ header size, and $O(1)$ decryption time. The third scheme is called PK-LSD-PI, and it is based on the scheme PK-LSD. PK-LSD-PI achieves $O(1)$ public key size, $O(\log^{1+\epsilon} n)$ secret key size, $O(r/\epsilon)$ header size, and $O(1)$ decryption time, where $0 < \epsilon < 1$ is a system parameter.


In 2008, Selvi, Vivek, Gopalakrishnan, Karuturi, and Rangan [SVG+08] found a vulnerability of the ID-based broadcast signcryption scheme that was proposed by Mu, Susilo, Lin, and Ruan [MSLR04] in 2004. They proposed a new scheme that is called IBBSC is provably secure and achieves the IND-CCA2 and EUF-CMA2 security. IBBSC achieves $O(n)$ public key size, $O(1)$ secret key size, $O(n - r)$ header size, and $O(n - r)$ decryption time.

Table 1: Comparison of some broadcast encryption schemes

Scheme	Public Key Size	Secret Key Size	Header Size	Decryption Time
CS	N/A	$O(\log n)$	$O(r \log \frac{n}{r})$	$O(\log \log n)$
SD	N/A	$O(\log^2 n)$	$O(r)$	$O(\log n)$
YJCK04	N/A	$O(\log \frac{n}{r})$	$O(\alpha r + n)^1$	$O(d_w)^2$
HL05	N/A	$O(\log n)$	$O(r)$	$O(n^\beta)^3$
PK-CS	$O(1)$	$O(\log n)$	$O(r \log \frac{n}{r})$	$O(1)$
PK-SD	$O(1)$	$O(\log^2 n)$	$O(r)$	$O(\log n)$
BGW05 (i)	$O(n)$	$O(1)$	$O(1)$	$O(n - r)$
BGW05 (ii)	$O(\sqrt{n})$	$O(1)$	$O(\sqrt{n})$	$O(\sqrt{n})$
BE-PI	$O(1)$	$O(\log n)$	$O(r)$	$O(r)$
PK-SD-PI	$O(1)$	$O(\log^2 n)$	$O(r)$	$O(1)$
PK-LSD-PI	$O(1)$	$O(\log^{1+\epsilon} n)^4$	$O(r/\epsilon)^4$	$O(1)$
Our Scheme	$O(1)$	$O(\log n)$	$O(r)$	$O(1)$

¹ $1 < \alpha < 2$ is a system parameter.² $d_1 = 1$ and $d_i = \lfloor \alpha(d_{i-1} + 1) \rfloor$. Let w be the minimal integer such that $n \leq (\alpha + 1)(d_w + 1) + 1$.³ $0 < \beta \leq 1$ is a system parameter.⁴ $0 < \epsilon < 1$ is a system parameter.

1.2 Our Contribution



We proposed a fully collusion resistant public key broadcast encryption scheme based on the idea of key derivation, the scheme PK-SD-PI, and the result of Boyen [Boy07]. Our scheme is an improvement on PK-SD-PI. We used the idea of key derivation to reduce the number of secret keys to $O(\log n)$. In our scheme, the polynomials are not all independent. The share of a polynomial can be derived from the share of another polynomial. Thus users don't need to store as many as $O(\log^2 n)$ shares, and the size of secret key is reduced. We also used the result of [Boy07] to achieve the IND-CCA2 security without the extra redundancy in the random oracle model. Thus the header size is still the same as the original scheme PK-SD-PI, and the communication cost doesn't rise up. Our scheme achieves $O(1)$ public key size, $O(\log n)$ secret key size, $O(r)$ header size, and $O(1)$ decryption time. To the best of our knowledge, our scheme is the most efficient scheme in the existing broadcast encryption schemes.

2 Preliminary

We use the *bilinear map* as a fundamental tool to build our public key *broadcast encryption* scheme. We prove that our scheme is IND-CCA2 secure by reducing from the *Gap-BDH problem*. Thus our scheme is secure if *Gap-BDH assumption* is held. We are going to introduce the background knowledge used in this paper.

2.1 Bilinear Map

It is also known as pairing on elliptic curve. Let p be a large prime, $G = \langle g \rangle$ and $G_t = \langle g_t \rangle$ be two multiplicative groups such that $|G| = |G_t| = p$, and $\hat{e} : G \times G \rightarrow G_t$ be a computable pairing. Then \hat{e} should satisfy the following properties:

- Bilinear. $\forall x, y \in \mathbb{Z}_p$, we have $\hat{e}(g^x, g^y) = \hat{e}(g, g)^{xy}$.
- Non-Degenerate. We have $\hat{e}(g, g) = g_t \neq 1$.
- Computable. It can be computed in polynomial time with respect to $|p|$.

2.2 Gap-BDH Assumption

Let k be the security parameter, p be a large prime with $|p| = k$, $G = \langle g \rangle$ and $G_t = \langle g_t \rangle$ be two multiplicative groups such that $|G| = |G_t| = p$, and $\hat{e} : G \times G \rightarrow G_t$ be a computable pairing. For an instance (g, g^a, g^b, g^c) of Bilinear Diffie-Hellman (BDH) problem where $a, b, c \in_R \mathbb{Z}_p$, the decisional oracle O about the instance is:

$$O(g, g^a, g^b, g^c, W) = \begin{cases} 1, & \text{if } W = \hat{e}(g, g)^{abc} \\ 0, & \text{otherwise} \end{cases}$$

The Gap-BDH problem is described as follows:

- Given (g, g^a, g^b, g^c) as input and the access right of O , compute $\hat{e}(g, g)^{abc}$.

Then the Gap-BDH assumption is defined as follows:

- It is hard to solve the BDH problem for any polynomial-time adversary A , even with the help of the decisional oracle O . It means that the Gap-BDH problem is hard for any polynomial-time adversary. Therefore, we have

$$\Pr[A^O(g, g^a, g^b, g^c) = \hat{e}(g, g)^{abc}] \leq \text{neg}(k)$$

where $\text{neg}(k)$ is some negligible function respect to k .

Gap-BDH assumption is a variant of BDH assumption, and the former is stronger than the latter. Gap-BDH assumption is not as widely used as BDH assumption. However, it is getting popular. For example, see [Boy07], [AFG+06], [BSNS05], [TNJ+05], [HSaFZ06], and [HSaFZ08].

2.3 Broadcast Encryption

Usually, there is a trusted third party, also known as the private key generator (PKG), to setup a public key broadcast encryption system. A scheme consists of the three algorithms:

- $\text{Setup}(1^k, ID, U)$. k is the security parameter, ID is the identity of system, and $U = \{u_1, u_2, \dots, u_n\}$ is the set of users in this system where n is the total number of users. Setup takes these parameters as input and outputs the public key PK , the master secret key MSK for PKG, and the set $SK = \{SK_1, SK_2, \dots, SK_n\}$ of secret keys where SK_i is the secret key of user $u_i \in U$.
- $\text{Enc}(PK, T, Msg)$. PK is the public key, $T \subseteq U$ is the set of valid receivers for this encryption, and Msg is the message to be encrypted. Enc takes these parameters as input and outputs the ciphertext Ctx of Msg . Only the user u_i in T can decrypt Ctx with his secret key SK_i , while other users can not.
- $\text{Dec}(PK, SK_i, Ctx)$. PK is the public key, SK_i is the secret key of user u_i in T , and Ctx is the ciphertext to T . Dec takes these parameters as input and outputs the

message Msg of Ctx if SK_i is valid.

We say that a public key broadcast encryption scheme is IND-CCA2 secure if there is no polynomial-time adversary A that can win the following game with non-negligible advantage:

- Initial. A chooses a system identity ID and a set $T \subseteq U$ of users to attack.
- Setup. Challenger C runs algorithm $\text{Setup}(1^k, ID, U)$, and obtains the public key PK and secret keys $SK = \{SK_1, SK_2, \dots, SK_n\}$ of all users in U . Then C sends the public key PK and the secret keys SK_u to A for each user $u \in U \setminus T$.
- Query Phase 1. A issues decryption queries $Q = \{Q_1, Q_2, \dots, Q_q\}$ to C adaptively where q is polynomial to k . Then C responds to A 's queries with the answers $R = \{R_1, R_2, \dots, R_q\}$ such that $Q_i = \text{Enc}(PK, T, R_i)$.
- Challenge. A chooses two messages M_0 and M_1 to C . Then C chooses a random bit b , and runs the algorithm $\text{Enc}(PK, T, M_b)$. C returns the ciphertext Ctx to A .
- Query Phase 2. It is almost the same as the Query Phase 1, except that A can not issue the decryption query about Ctx .
- Guess. A makes a guess b' . He wins the game if $b' = b$.

3 Idea of SD

D. Naor, M. Naor, and Lotspiech [NNL01] firstly proposed the tree-based broadcast encryption system. They saw the users as the leaves of a complete binary tree, and the internal nodes of the binary tree present the subsets of users that are descendants of them. We use T_i to denote the set of users that are leaves of the subtree which is rooted in node i . In figure 1, we assume that there are totally 8 users here. These users are presented by the leaf nodes u_1, u_2, \dots, u_8 respectively. The users $u_1, u_2, u_3,$ and u_4 is covered by the node that is labeled 2, and we use T_2 to present this set.

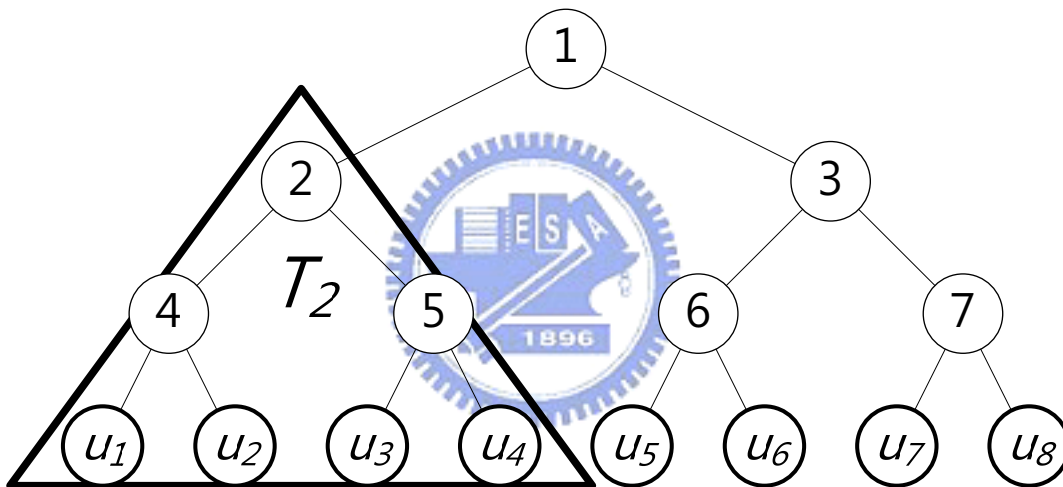


Figure 1: Tree-based structure

D. Naor, M. Naor, and Lotspiech viewed designing a broadcast encryption scheme as solving the subset-cover problem in the tree-based system. They proposed a *Subset-Cover* framework and two broadcast encryption schemes based on the framework. Let n be the number of total users and r be the number of the revoked users. The first scheme is called CS. In CS, each user only needs to store $O(\log n)$ secret keys. But it partitions the receivers into $O(r \log \frac{n}{r})$ disjoint subsets, then the header size is too large. The second scheme is called SD, it reduces the header size from $O(r \log \frac{n}{r})$ to $O(r)$. But the number of secret keys a user holds is increasing to $O(\log^2 n)$. We let the subset difference T_x^i denotes the set of users that

are in the subtree T_i but not in the subtree T_x , see figure 2.

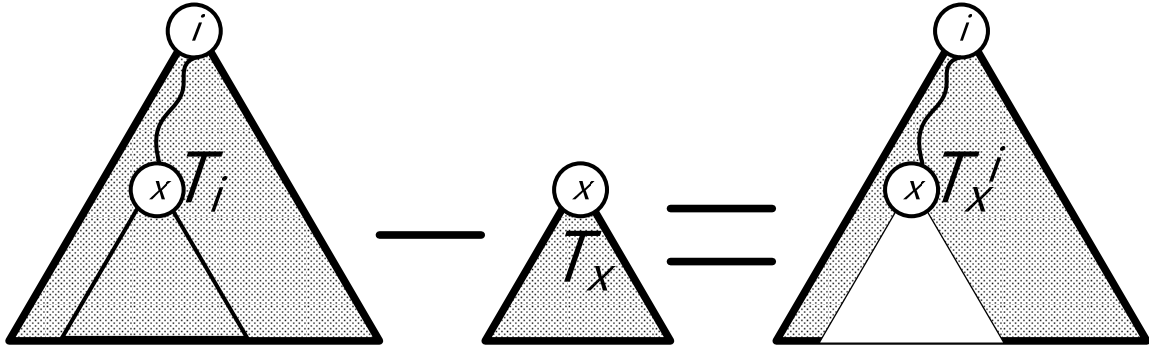


Figure 2: Subset difference

In SD, we partition the receivers into disjoint subset differences. Use figure 3 as an example, we want to send a message to the users u_1, u_2, u_6, u_7 , and u_8 . Then we can partition these receivers into the subset differences T_5^2 and $T_{u_5}^3$ respectively, users $\{u_1, u_2\} \in T_5^2$ and users $\{u_6, u_7, u_8\} \in T_{u_5}^3$.

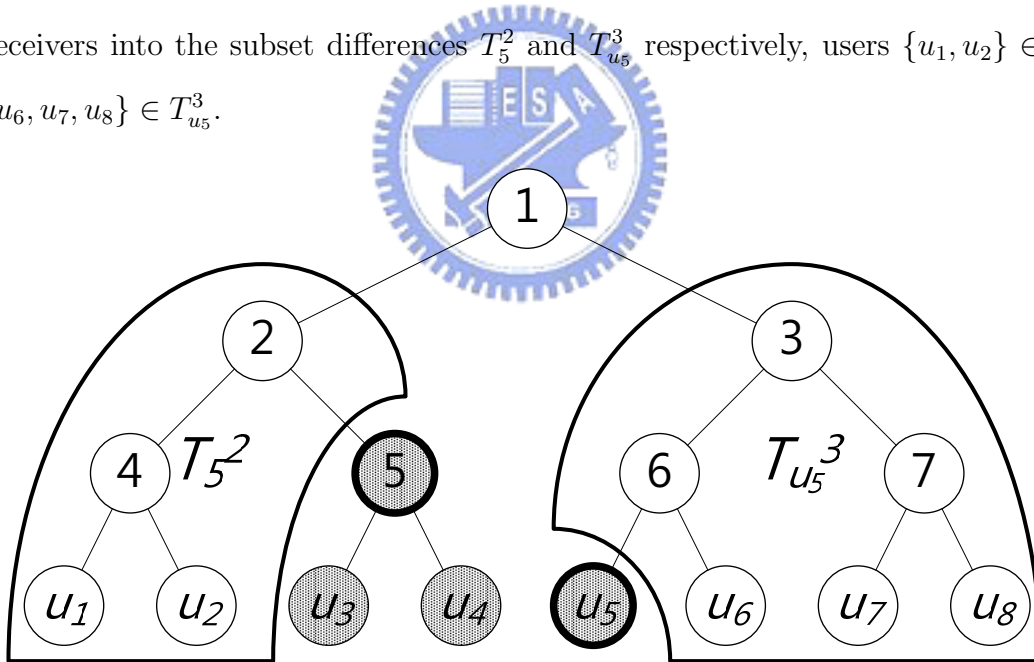


Figure 3: Partition of receivers

Let U be the set of users in the system and R be the set of revoked users for this time of message sending. We are going to partition the receivers $U \setminus R$ into the disjoint subset differences $T_{x_1}^{i_1}, T_{x_2}^{i_2}, \dots, T_{x_r}^{i_r}$ where r is the number of partitions. Let $ST(R)$ be the directed

Steiner Tree induced by R and the root. We generate the collection of the disjoint subsets iteratively. We maintain a subtree T' of the tree $ST(R)$ with the property that any leaf node $u \in U \setminus R$ of T' has been covered. In the beginning, we let T' be equal to $ST(R)$. Then we remove the nodes from T' and add the subset difference to the collection iteratively, until T' consists of only one single node:

1. Find two leaves v_i and v_j in the tree T' such that their least common ancestor v does not contain any other leaf of T' in the subtree T_v . Let v_k and v_l be the two children of v such that v_k is an ascendant of v_i and v_l is an ascendant of v_j . If there is only one leaf in T' , let $v_i = v_j$ be the leaf and $v = v_k = v_l$ be the root of T' .
2. If $v_k \neq v_i$, add the subset difference T_i^k to the collection. Likewise, if $v_l \neq v_j$, add the subset difference T_j^l to the collection.
3. Remove the subtree T_v from T' , and make the node v a leaf in T' .

Let $T = T_{x_1}^{i_1} \cup T_{x_2}^{i_2} \cup \dots \cup T_{x_r}^{i_r}$ be the union of the disjoint subset differences of valid receivers and r be the number of partition of T in SD. While broadcasting a message M , we encrypt M with the session key K and encrypt K for r times with the secret keys SK_1, SK_2, \dots, SK_r respectively where SK_k , $1 \leq k \leq r$, is the secret key of the subset difference $T_{x_k}^{i_k}$ of receivers. Thus the ciphertext is $\langle E_K(M), E'_{SK_1}(K), E'_{SK_1}(K), \dots, E'_{SK_r}(K) \rangle$ where E and E' are some symmetric encryptions, e.g. AES.

In SD, each user needs to store all the secret keys of the subset differences that he belongs to. Use figure 4 as an example, user u_3 needs to store the secret keys of the subset differences T_{11}^5 , $\{T_4^2, T_8^2, T_9^2, T_{11}^2\}$, and $\{T_3^1, T_4^1, T_6^1, T_7^1, T_8^1, T_9^1, T_{11}^1, T_{12}^1, T_{13}^1, T_{14}^1, T_{15}^1\}$. The storage amount of each user is too high, it is $O(n)$.

To reduce the storage amount, D. Naor, M. Naor, and Lotspiech proposed the hash chain solution. For each subtree T_i , the node x of T_i is assigned an unique label L_x^i . Then they let

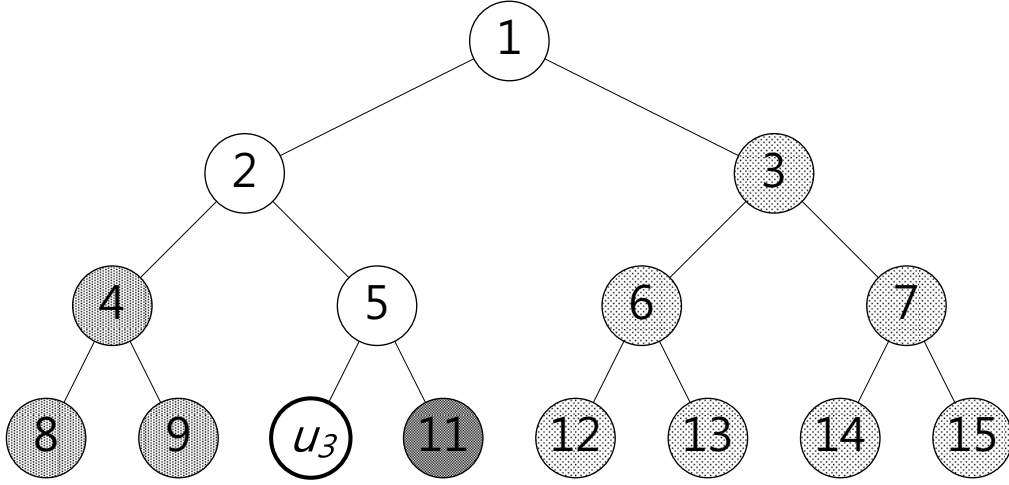


Figure 4: Key storage

the labels of the children nodes can be generated from the label of the parent node by using the two one-way hash functions H_L and H_R . Moreover, the hash function H_L is for the left child node and the hash function H_R is for the right child node, thus we have $L_{2x}^i = H_L(L_x^i)$ and $L_{2x+1}^i = H_R(L_x^i)$. The secret key SK_x^i of the subset difference T_x^i is generated by the one-way hash function H_K such that $SK_x^i = H_K(L_x^i)$. Thus each user doesn't need to store all the secret keys anymore, he just stores the labels of the sibling nodes along the path from himself to the root node i for each subtree T_i . The storage amount for each user is reduced from $O(n)$ to $O(\log^2 n)$ by the hash chain solution.

In figure 5, we use the subtree T_1 as an example and let L_x denotes the label L_x^1 for simplicity. The user u_3 needs to store the labels L_{11} , L_4 , and L_3 in the subtree T_1 . Then he can use the hash functions H_L and H_R to generate the other labels that he should own, like $\{L_8, L_9\}$ from L_4 , $\{L_6, L_7\}$ from L_3 , $\{L_{12}, L_{13}\}$ from L_6 , and $\{L_{14}, L_{15}\}$ from L_7 . User u_3 can use the hash function H_K to generate the secret keys of subtree T_1 he needs.

An user needs to store $O(\log n)$ secret keys for a subtree, and there are totally $O(\log n)$ subtrees for an user. Thus the number of secret keys a user needs to store is bounded by $O(\log^2 n)$.

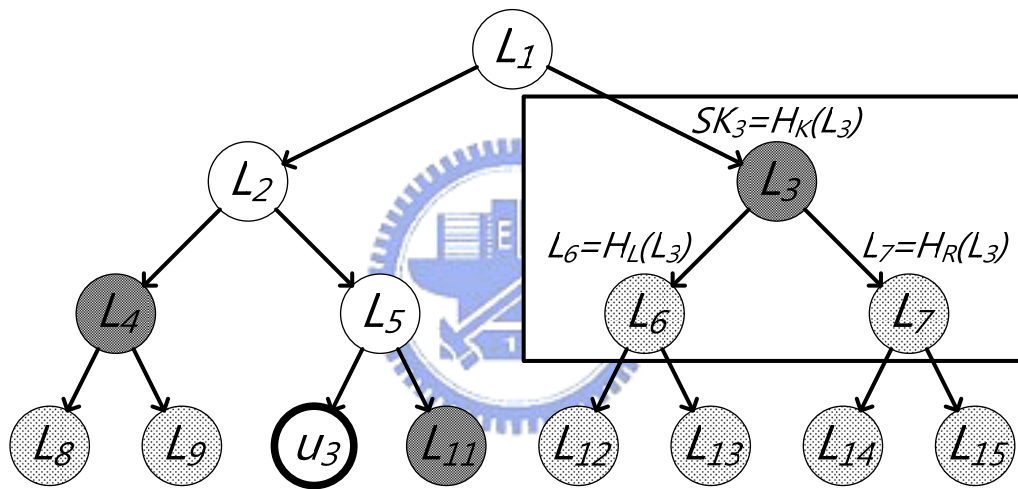


Figure 5: Hash chain method

4 Idea of PK-SD

Dodis and Fazio [DF03] proposed two public key broadcast encryption schemes that are called PK-CS and PK-SD by applying Identity Based Encryption (IBE) and Hierarchical Identity Based Encryption (HIBE) to the schemes CS and SD respectively.

The distinctive difference between the general public key encryption and IBE system is that the public key of user can be any arbitrary or meaningful string in IBE while it usually is a random or meaningless string in the general public key encryption system. In IBE system, we usually use the receiver's identity, e-mail address, or telephone number as receiver's public key to encrypt the message. There is a trusted third party that is called Private Key Generator (PKG). PKG holds the master secret key, and he can generate any private keys of any identity if he wants to. Each user needs to register himself to the PKG, thus he can obtain his own private key.

HIBE system enjoys more features than IBE system. The users in HIBE are in the tree-based structure, and we can see them as the nodes of a binary tree. The maximum level of the binary tree is a system parameter that is decided before the implementation of the system. The important feature of HIBE is that the ascendant user can derivate the private keys of his descendant users from his private key. Thus the message for the descendant user can also be decrypted by his ascendant users, while the message for the ascendant user can not be decrypted by his descendant users. It means that the users in the higher level have more authority than the users in the lower level.

Dodis and Fazio proposed a naming system to generate the public key of each subset difference. For each subtree T_i , they give the root node i a unique identity independently. Then the identity of the left child is generated by concatenating 0 to his parent's identity, while the identity of the right child is generated by concatenating 1. In figure 6, we use the subtree T_1 as an example. We give the root node 1 of the subtree T_1 a unique identity R .

Then the identities of his children are R_0 and R_1 respectively.

We can use these identities as public keys to encrypt the messages. The users in PK-CS or PK-SD store their private keys as if they are in CS or SD respectively. But in PK-SD, the user needs to be able to derivate the other private keys that he should hold from his private keys. So we have to use the HIBE system in PK-SD, while we just use IBE system in PK-CS.

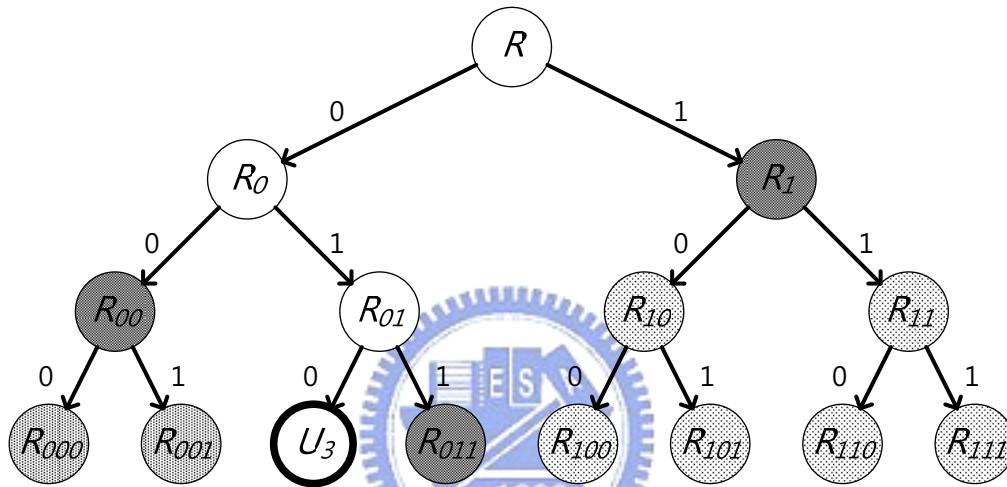


Figure 6: Naming system

5 Idea of PK-SD-PI

Liu and Tzeng [LT08] proposed three fully collusion resistant broadcast encryption schemes based on the idea of polynomial interpolation. The schemes are called BE-PI, PK-SD-PI, and PK-LSD-PI respectively. In PK-SD-PI, the users are the leaves of a complete binary tree, and the internal nodes of the binary tree present the sets of users that are covered. We use T_i to denote the set of users that are leaves of the subtree which is rooted in node i , and we let the subset difference T_x^i denotes the set of users that are in the subtree T_i but not in the subtree T_x . These settings are the same as SD, and the distinctive difference between PK-SD-PI and SD is about the secret keys. In PK-SD-PI, an user stores the secret keys that are related to his ascendant nodes. While in SD, an user stores the secret keys that are related to his sibling nodes.

For each subtree T_i , we denote the maximum number of its levels as l_i . Then there are l_i independent polynomials belonging to T_i , one polynomial to one level of the subtree. These polynomials are named as $f_1^i, f_2^i, \dots, f_{l_i}^i$ respectively, and the polynomial f_j^i , $1 \leq j \leq l_i$, is used for the nodes that are in the j -th level of T_i . Use figure 7 as an example, we assume that there are totally 8 users here. The polynomial f_1^1 is used for node 2 and node 3 in the subtree T_1 , and the polynomial f_2^2 is used for the leaf nodes u_1, u_2, u_3 , and u_4 in the subtree T_2 . These polynomials are all degree of one, and we let $f_j^i(X) = a_{j,1}^i X + a_{j,0}^i$ for $1 \leq j \leq l_i$. The secret of the polynomial f_j^i is $f_j^i(0)$, the coefficient $a_{j,0}^i$ of the constant term. Thus the coefficients of these polynomials are made private, but their masked values, hidden in the exponent, are public information.

For each subtree T_i , the covered user needs to store the shares of the polynomials f_j^i that are at the path from himself to the root node i . Use figure 7 as an example, the user u_3 needs to store the shares $\{f_1^1(2), f_2^1(5), f_3^1(u_3)\}$ for subtree T_1 , $\{f_1^2(5), f_2^2(u_3)\}$ for subtree T_2 , and $f_1^5(u_3)$ for subtree T_5 .

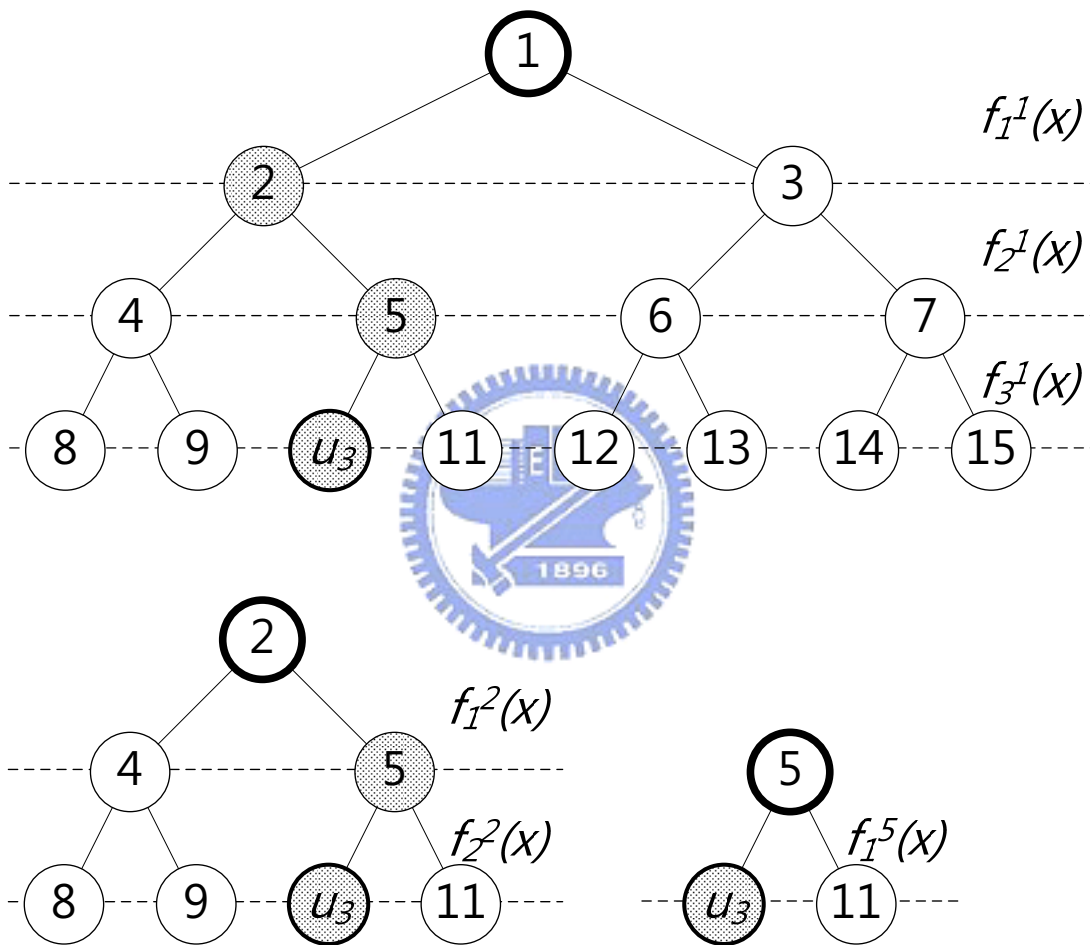


Figure 7: Polynomials of subtrees

Let $T = T_{x_1}^{i_1} \cup T_{x_2}^{i_2} \cup \dots \cup T_{x_r}^{i_r}$ be the union of the disjoint subset differences of valid receivers and r be the number of subset differences of receivers. To broadcast a message M , we encrypt M with the system public key, and the master secret key is masked by the secrets of the polynomials that are related to the subset differences respectively. The idea is secret sharing. The valid receivers can obtain enough shares to recover the secret of the polynomial for decrypting the message, while the revoked users can not have enough shares from the broadcasted messages. Thus we broadcast the ciphertext and the shares held by the revoked users. The ciphertext is $\langle Enc(M), P_1, P_2, \dots, P_r \rangle$ where $Enc(M)$ is the ciphertext of M and P_k is the shares.

Use figure 8 as an example, we want to send a message to the users u_1, u_2, u_6, u_7 , and u_8 . These users are partitioned into the subset differences T_5^2 and $T_{u_5}^3$, and the related polynomials are f_1^2 and f_2^3 respectively. We use the system public key to encrypt the message, and the master secret key is masked by $f_1^2(0)$ and $F_2^3(u_5)$ respectively. Then we broadcast the share $f_1^2(5)$ held by the revoked users $\{u_3, u_4\}$ and the share $f_2^3(u_5)$ held by the revoked user u_5 . The users u_1 and u_2 can have enough shares $\{f_1^2(4), f_1^2(5)\}$ to recover the secret for decryption. But the revoked users u_3 and u_4 only have their own share $f_1^2(5)$, and it is not enough for decryption.

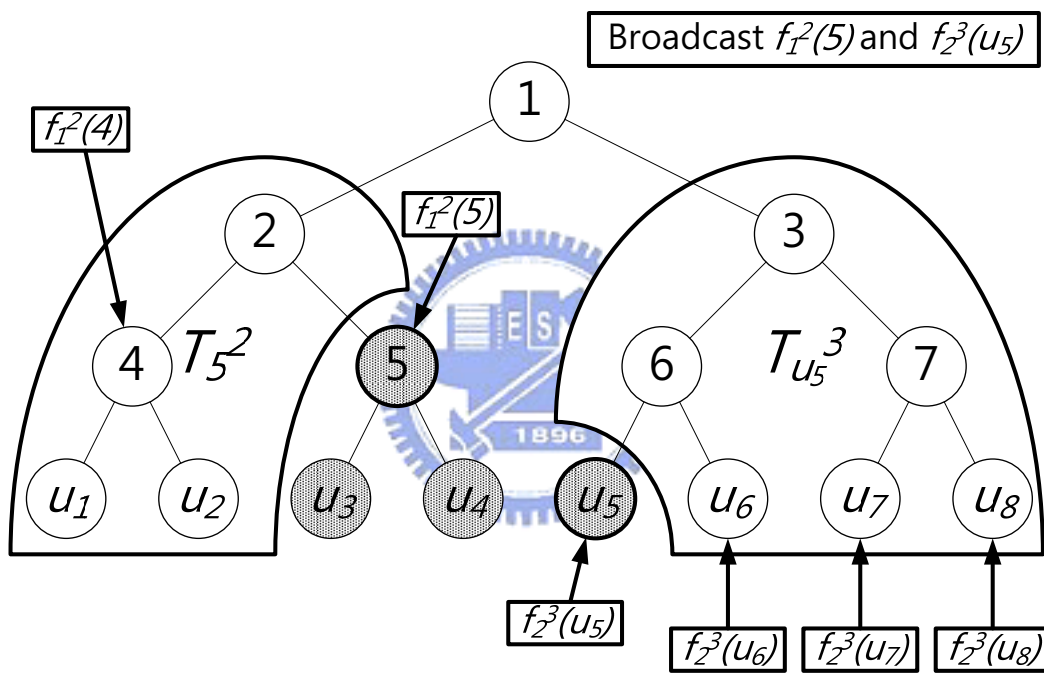


Figure 8: PK-SD-PI

6 Our Scheme

In PK-SD-PI, an user needs to store $O(\log^2 n)$ secret keys. Use figure 7 as an example, the user u_3 needs to store the shares $\{f_1^1(2), f_2^1(5), f_3^1(u_3)\}$, $\{f_1^2(5), f_2^2(u_3)\}$, and $f_1^5(u_3)$. We observe that these shares usually use the same input values but apply to different polynomials. For example, $\{f_2^1(5), f_1^2(5)\}$ share the same input 5 and $\{f_3^1(u_3), f_2^2(u_3), f_1^5(u_3)\}$ share the same input u_3 . It means that we can have an easy method of key derivation, because we only need to focus on the translation of coefficients between the polynomials. That's why we can use $O(\log n)$ independent polynomials and $O(\log n)$ independent information to substitute the $O(\log^2 n)$ independent polynomials in PK-SD-PI.

Our key derivation method is in the horizontal form, not in the intuitional vertical form. We let the polynomials of subtree T_i can be derivated from the polynomials of subtree T_1 respectively. Using the information kd_i , the polynomial f_j^i can be derivated from the polynomial $f_{j+\lfloor \log i \rfloor}^1$ directly. In figure 9, $\{f_1^2(5), f_2^2(u_3)\}$ can be derivated from $\{f_2^1(5), f_3^1(u_3)\}$ respectively with the same information kd_2 , and $f_1^5(u_3)$ can be derivated from $f_3^1(u_3)$ with the information kd_5 .

The polynomials belong to subtree T_1 are all independent, and the information kd_i , $2 \leq i \leq n - 1$, used for key derivation are all independent, too. Figure 10 is a sketch of the polynomial derivation. The polynomial $f_j^i(x)$ can be derivated from the polynomial $f_{j'}^1(x) = A_{j'}x^2 + B_{j'}x + C_{j'}$ with $kd_i = (D_i, E_i)$ by adding D_i to the coefficient of x -term and E_i to the coefficient of constant term, that is $f_j^i(x) = f_{j'}^1(x) + D_ix + E_i$.

Let k be the security parameter, and p be a large prime such that $|p| = k$. We have a multiplicative group $G = \langle g \rangle$ such that $|G| = p$. Let $H_1 : \{0, 1\}^* \rightarrow \{0, 1\}^k$ and $H_2 : \{0, 1\}^* \rightarrow G$ be collision resistant hash functions. Furthermore, let ID be the identity name of the scheme, h and $h' \in G$ be the assigned values, and n be the total number of users in the scheme. Without loss of generality, we assume that n is a power of 2, and let \log be the

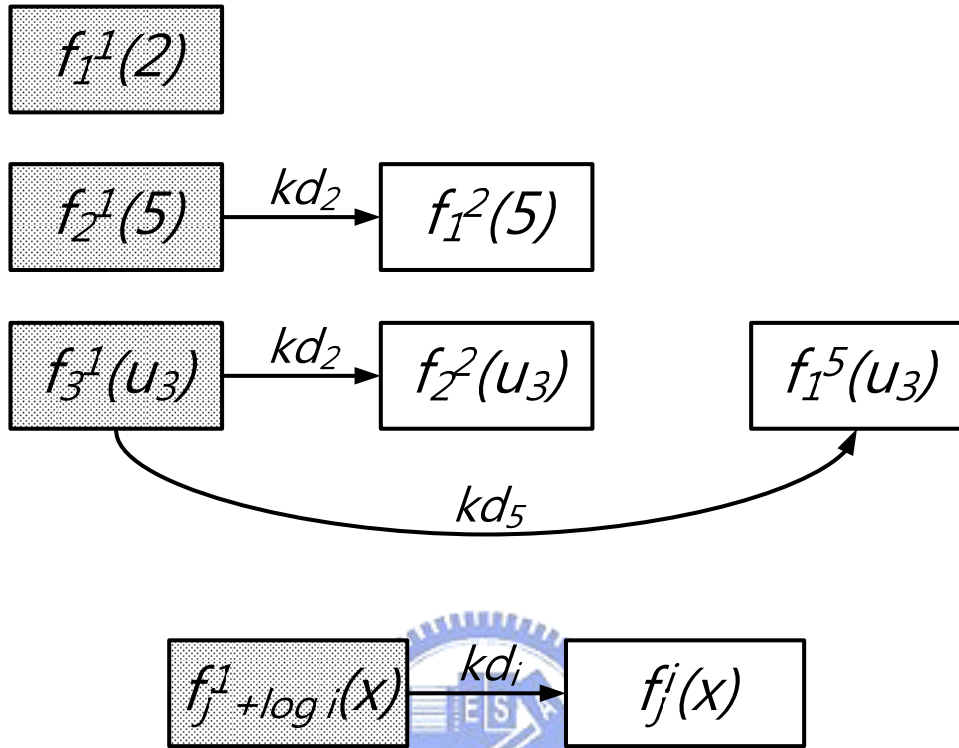


Figure 9: Key derivation

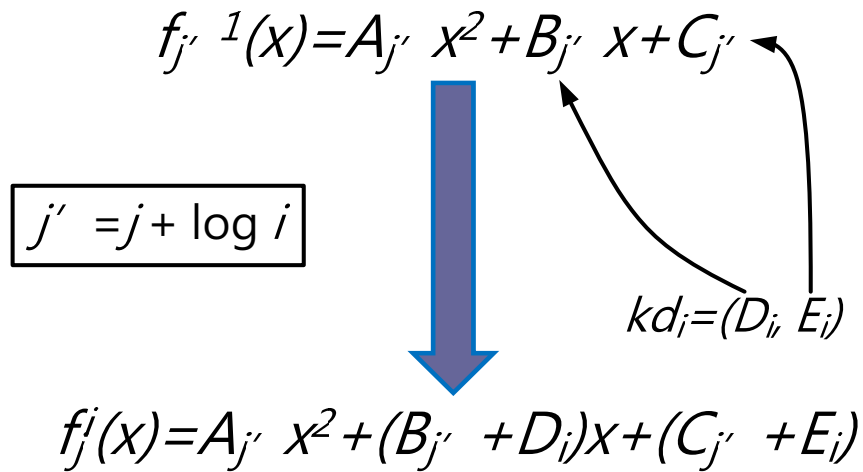


Figure 10: Polynomial derivation

logarithm with the base 2.

In our scheme, the users are the leaves of a complete binary tree, and the internal nodes cover the users in its descendant leaves. The polynomial f_j^i is used for the nodes that are descendants of node i and are in the j -th level from node i . These settings are almost the same as PK-SD-PI, except that we use the polynomials of degree of 2 while PK-SD-PI uses the polynomials of degree of 1. We denote $f_j^i(X) = a_{j,2}^i X^2 + a_{j,1}^i X + a_{j,0}^i \pmod{p}$ be a 2-degree polynomial function.

Each polynomial f_j^i is assigned a specific value either h or h' . Most of them are assigned h , and the remains are assigned h' . We claim that for each subtree T_i , there are at most one polynomial $f_{j'}^i$ that is assigned h' . We use the hash function H_1 to pick up which polynomials are assigned h' :

For each subtree T_i , we let

$$j'_i \equiv H_1(ID||i) \pmod{\log n - \lfloor \log i \rfloor + 1}$$

where $\log n - \lfloor \log i \rfloor + 1$ equals to the number of levels of T_i plus one.

- If $j'_i = 0$, the polynomials belong to subtree T_i are all assigned h .
- If $j'_i \neq 0$, the polynomial $f_{j'_i}^i$ is assigned h' . The other polynomials belong to subtree T_i are assigned h .

Figure 11 is an example. Subtree T_u has 4 levels, and we have $H_1(ID||u) \equiv 2 \pmod{5}$. Thus the polynomial f_2^u is assigned h' , and the remaining polynomials of T_u are assigned h . Likewise, Subtree T_v has 3 levels, and we have $H_1(ID||v) \equiv 0 \pmod{4}$. Thus there is no polynomial of T_v assigned h' , and the polynomials belong to T_v are all assigned h .

Then there are three possible relation between the polynomial $f_{j+\lfloor \log i \rfloor}^1$ and f_j^i :

1. The polynomial $f_{j+\lfloor \log i \rfloor}^1$ is assigned h (h'), and the polynomial f_j^i is also assigned the same value h (h'). We denote it as $h \rightarrow h$ ($h' \rightarrow h'$).

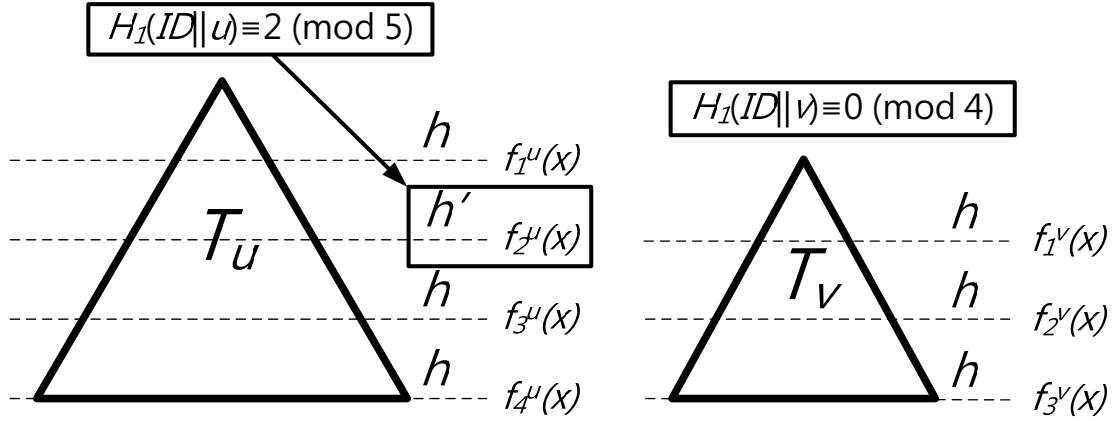


Figure 11: Assignment of h and h'

2. The polynomial $f_{j+\lceil \log i \rceil}^1$ is assigned h , while the polynomial f_j^i is assigned h' . We denote it as $h \rightarrow h'$.

3. The polynomial $f_{j+\lceil \log i \rceil}^1$ is assigned h' , while the polynomial f_j^i is assigned h . We denote it as $h' \rightarrow h$.

For different relation between $f_{j+\lceil \log i \rceil}^1$ and f_j^i , we use different information to derivate the polynomial f_j^i . Thus kd_i is composed of three parts $(D_{i,1}, E_{i,1})$, $(D_{i,2}, E_{i,2})$, and $(D_{i,3}, E_{i,3})$.

Figure 12 is a sketch for the polynomial derivation:

- The first part $(D_{i,1}, E_{i,1})$ is used for the case 1: $h \rightarrow h$ or $h' \rightarrow h'$.
- The second part $(D_{i,2}, E_{i,2})$ is used for the case 2: $h \rightarrow h'$.
- The third part $(D_{i,3}, E_{i,3})$ is used for the case 3: $h' \rightarrow h$.

The coefficients of the polynomial $f_j^i(X) = a_{j,2}^i X^2 + a_{j,1}^i X + a_{j,0}^i$ are secrets, but their masked values are public information. The masked values are the coefficients put in the exponents of g , that is $g^{a_{j,k}^i}$ for $0 \leq k \leq 2$. The information kd_i for key derivation also needs to be masked, that is $kd_i = \{(g^{b_{1,0}^i}, g^{b_{1,1}^i}), (g^{b_{2,0}^i}, g^{b_{2,1}^i}), (g^{b_{3,0}^i}, g^{b_{3,1}^i})\}$.

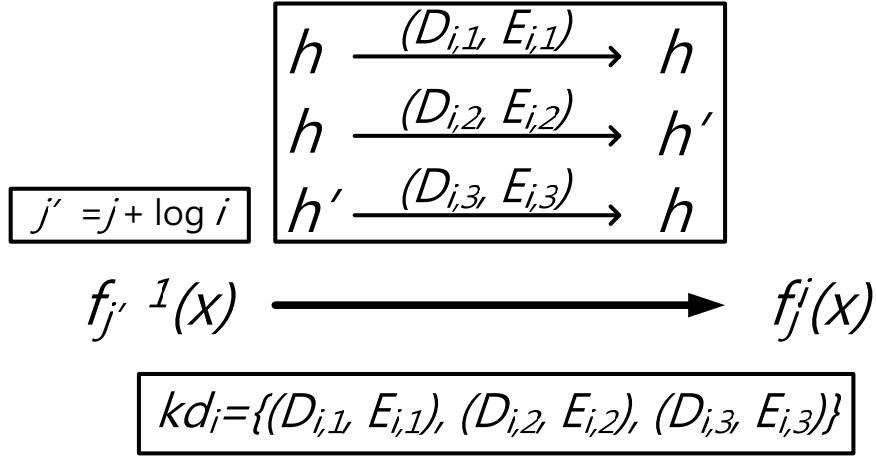


Figure 12: Polynomial derivation depends on the relation

The masked values of the polynomials belong to subtree T_1 and the information kd_i where $2 \leq i \leq n - 1$ are computed as follows:

- $g^{a_{j,s}^1} = H_2(ID||1||j||s)$, for $1 \leq j \leq \log n$ and $0 \leq s \leq 2$
- $g^{b_{l,s}^i} = H_2(ID||i||l||s)$, for $1 \leq l \leq 3$ and $0 \leq s \leq 1$

where $a_{j,s}^1$ is the coefficient of the X^s -term of the polynomial f_j^1 , and $b_{l,s}^i$ is the key derivation information for the X^s -term of the polynomial belong to subtree T_i respect to the case l .

Thus the masked values of the polynomials belong to subtree T_i where $2 \leq i \leq n - 1$ are computed as follows:

- $g^{a_{j,2}^i} = g^{a_{j+\lfloor \log i \rfloor, 2}^1}$, for $1 \leq j \leq \log n - \lfloor \log i \rfloor$
- $g^{a_{j,s}^i} = g^{a_{j+\lfloor \log i \rfloor, s}^1} \times g^{b_{l,s}^i}$ if it is case l , for $1 \leq j \leq \log n - \lfloor \log i \rfloor$ and $0 \leq s \leq 1$

The masked coefficients of the polynomials can be computed easily. But these polynomials can not be accessed directly since their function values are the secrets to guarantee the security. The users can access these polynomials only in the masked way (in the exponent of g).

6.1 Construction

Let k be the security parameter, and p be a large prime such that $|p| = k$. We have two multiplicative groups $G = \langle g \rangle$ and $G_t = \langle g_t \rangle$ such that $|G| = |G_t| = p$, and a bilinear map $\hat{e} : G \times G \rightarrow G_t$ such that $\hat{e}(g, g) = g_t$. Let $H_1 : \{0, 1\}^* \rightarrow \{0, 1\}^k$, $H_2 : \{0, 1\}^* \rightarrow G$, $\pi : G_t \rightarrow Z_p$, $\Psi : G_t \rightarrow G_t$, and $\Phi : G \times G_t \rightarrow G_t$ be collision resistant hash functions. Furthermore, let ID be the identity name of the scheme, h and $h' \in G$ be the assigned values, and n be the total number of users in the scheme. Without loss of generality, we assume that n is a power of 2, and let \log be the logarithm with the base 2. We denote $\Delta h = h'/h$.

The scheme is composed of the following algorithms:

- Setup($1^k, ID, U$):

k is the security parameter, ID is the identity of this system, and U is the set of users in the system. The private key generator (PKG) chooses the master secret key $\rho \in Z_p$ randomly, thus $MSK = \rho$. PKG then publishes the public key PK and generates the secret key SK_v for each user u_v as follows:

$$\begin{aligned} - PK &= \{ID, G, G_t, \hat{e}, g, g^\rho, h, h', H_1, H_2, \pi, \Psi, \Phi\} \\ - SK_v &= \left\{ \begin{array}{l} sk_j^1, \text{ for } 1 \leq j \leq \log n \\ kd_i, \text{ for } i \in \text{Ancestor}(u_v) \end{array} \right\} \end{aligned}$$

where

$$sk_j^1 = \{g^{r_v}, g^{r_v f_j^1(I_{1,j}^v)}, g^{r_v f_j^1(0)} h^\rho\} \text{ or } sk_j^1 = \{g^{r_v}, g^{r_v f_j^1(I_{1,j}^v)}, g^{r_v f_j^1(0)} h'^\rho\}$$

it depends on the assigned value of polynomial f_j^1 , and

$$kd_i = \{(g^{r_v b_{1,1}^i}, g^{r_v b_{1,0}^i}), (g^{r_v (b_{2,1}^i I_{i,j'}^v + b_{2,0}^i)}, g^{r_v b_{2,0}^i} \Delta h^\rho), (g^{r_v (b_{3,1}^i I_{i,j''}^v + b_{3,0}^i)}, g^{r_v b_{3,0}^i} \frac{1}{\Delta h^\rho})\}$$

Here sk_j^1 is the share of polynomial f_j^1 held by the user u_v , and kd_i is the information used to derivate the polynomial f_j^i from the polynomial $f_{j+\lceil \log i \rceil}^1$. Moreover, $r_v \in Z_p$ is

chosen randomly, $g^{b_{i,s}^i} = H_2(ID||i||l||s)$ is the key derivation information, and $I_{i,j}^v$ is an identity of the node that is an ascendant of u_v and in the j -th level of the subtree T_i , see figure 13. We denote $I_{i,j'}^v$ as the identity of the node whose polynomials $f_{j'+\lfloor \log i \rfloor}^1$ and $f_{j'}^i$ are in the case 2: $h \rightarrow h'$, and $I_{i,j''}^v$ as the identity of the node whose polynomials $f_{j''+\lfloor \log i \rfloor}^1$ and $f_{j''}^i$ are in the case 3: $h' \rightarrow h$.

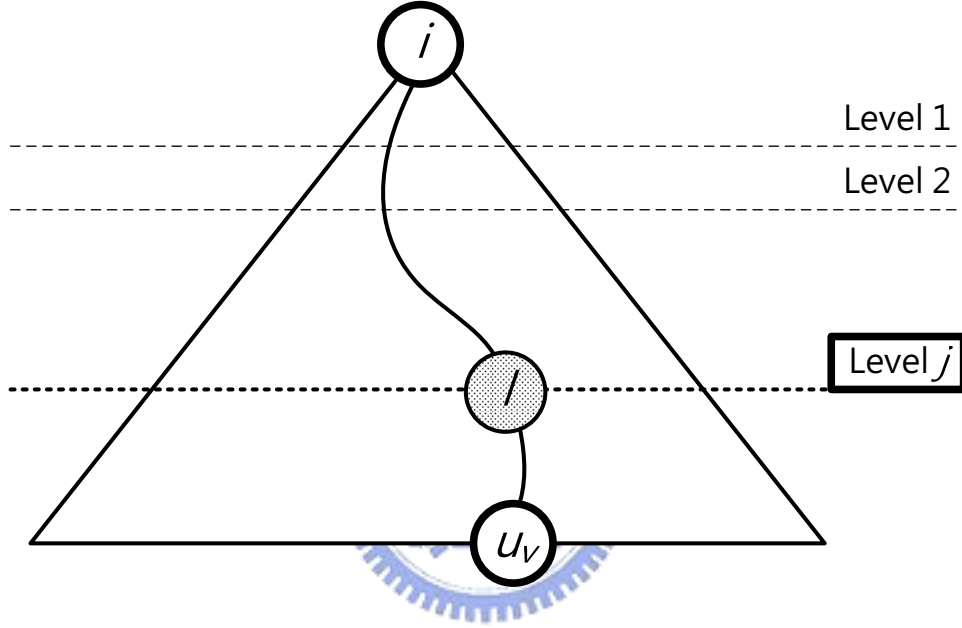


Figure 13: The notation $I_{i,j}^v$

When $f_{j+\lfloor \log i \rfloor}^1$ and f_j^i are assigned the same value h (h'), we use the first element $(g^{r_v b_{i,1}^i}, g^{r_v b_{i,0}^i})$ of kd_i to generate sk_j^i from $sk_{j+\lfloor \log i \rfloor}^1$ as follows:

$$sk_j^i = sk_{j+\lfloor \log i \rfloor}^1 \times \{1, (g^{r_v b_{i,1}^i})^{I_{i,j}^v} g^{r_v b_{i,0}^i}, g^{r_v b_{i,0}^i}\}$$

If $f_{j'+\lfloor \log i \rfloor}^1$ is assigned h and $f_{j'}^i$ is assigned h' , we use the second element $(g^{r_v (b_{2,1}^i I_{i,j'}^v + b_{2,0}^i)}, g^{r_v b_{2,0}^i} \Delta h^\rho)$ of kd_i to generate $sk_{j'}^i$ from $sk_{j'+\lfloor \log i \rfloor}^1$ as follows:

$$sk_{j'}^i = sk_{j'+\lfloor \log i \rfloor}^1 \times \{1, g^{r_v (b_{2,1}^i I_{i,j'}^v + b_{2,0}^i)}, g^{r_v b_{2,0}^i} \Delta h^\rho\}$$

If $f_{j''+\lfloor \log i \rfloor}^1$ is assigned h' and $f_{j''}^i$ is assigned h , we use the third element $(g^{r_v (b_{3,1}^i I_{i,j''}^v + b_{3,0}^i)}, g^{r_v b_{3,0}^i} \frac{1}{\Delta h^\rho})$

of kd_i to generate $sk_{j''}^i$ from $sk_{j''+\lceil \log i \rceil}^i$ as follows:

$$sk_{j''}^i = sk_{j''+\lceil \log i \rceil}^1 \times \left\{ 1, g^{r_v(b_{3,1}^i I_{i,j''}^v + b_{3,0}^i)}, g^{r_v b_{3,0}^i} \frac{1}{\Delta h^\rho} \right\}$$

- Enc(PK, T, Msg):

PK is the public key, $T \subseteq U$ is the set of valid receivers for this encryption, and Msg is the message to be encrypted. Let $T = \bigcup_{t=1}^r T_{x_t}^{i_t}$ be the union of disjoint subset differences of users, and r is the number of partitions for T .

The procedure of encryption is defined as follows:

1. Choose $z \in Z_p$ randomly and compute:

$$\left\{ \begin{array}{l} A = \hat{e}(g^\rho, h)^z \\ B = \Psi(A) \oplus Msg \\ C = \hat{e}(g^\rho, h)^{z/\pi(B)} \\ D = g^{z/\pi(B)} \\ E = \Phi(D, C) \oplus B \end{array} \right. \text{ and } \left\{ \begin{array}{l} A' = \hat{e}(g^\rho, h')^z \\ B' = \Psi(A') \oplus Msg \\ C' = \hat{e}(g^\rho, h')^{z/\pi(B')} \\ D' = g^{z/\pi(B')} \\ E' = \Phi(D', C') \oplus B' \end{array} \right.$$

2. Generate the broadcasted shares depending on $T = T_{x_1}^{i_1} \cup T_{x_2}^{i_2} \cup \dots \cup T_{x_r}^{i_r}$. For a subset difference $T_{x_t}^{i_t}$, assume that the node x_t is in the j_t -th level from the node i_t and $y_t \neq 0$ is not in the system (y_t does not present any nodes). For each subset difference $T_{x_t}^{i_t}$ where $1 \leq t \leq r$, compute the shares:

$$\left\{ \begin{array}{l} P_{t,1} = (i_t, x_t, g^{z/\pi(B)} f_{j_t}^{i_t}(x_t)) \\ P_{t,2} = (i_t, y_t, g^{z/\pi(B)} f_{j_t}^{i_t}(y_t)) \end{array} \right. \text{ and } \left\{ \begin{array}{l} P'_{t,1} = (i_t, x_t, g^{z/\pi(B')} f_{j_t}^{i_t}(x_t)) \\ P'_{t,2} = (i_t, y_t, g^{z/\pi(B')} f_{j_t}^{i_t}(y_t)) \end{array} \right.$$

Thus the ciphertext is like this:

$$Ctx = \left\{ \underbrace{(E, D, P_{1,1}, P_{1,2}, \dots, P_{r,1}, P_{r,2})}_{\text{for } h}, \underbrace{(E', D', P'_{1,1}, P'_{1,2}, \dots, P'_{r,1}, P'_{r,2})}_{\text{for } h'} \right\}$$

If the polynomials related to T are all assigned h , the part for h' of Ctx can be omitted. Likewise, If the polynomials related to T are all assigned h' , the part for h of Ctx can be omitted.

- Dec(PK, SK_v, Ctx):

PK is the public key, SK_v is the secret key of the user $u_v \in T$, and Ctx is the ciphertext to T . Without loss of generality, we assume that the user $u_v \in T_x^i$ where the node x is in the j -th level from the node i , and the polynomial f_j^i is assigned h . Thus we use the part for h of Ctx to decrypt. We denote P_1 and P_2 be the published shares that are related to the subset difference T_x^i . In practically, which pair $(P_{t,1}, P_{t,1})$ should be used for decryption is according to the first two elements (i_t, x_t) of the share $P_{t,1}$.

The procedure of decryption is defined as follows:

1. If $i \neq 1$, generate the secret key sk_j^i from $sk_{j+\lceil \log i \rceil}^1$ and kd_i respect to the real case. Thus we have:

$$sk_j^i = \{g^{r_v}, g^{r_v f_j^i(I_{i,j}^v)}, g^{r_v f_j^i(0)} h^\rho\}$$

2. Compute the pairing with $D = g^{z/\pi(B)}$ and $g^{r_v f_j^i(I_{i,j}^v)}$ of sk_j^i :

$$\hat{e}(g^{z/\pi(B)}, g^{r_v f_j^i(I_{i,j}^v)}) = g_t^{z/\pi(B) r_v f_j^i(I_{i,j}^v)}$$

3. Compute the pairings with $g^{z/\pi(B) f_j^i(x)}$ of P_1 , $g^{z/\pi(B) f_j^i(y)}$ of P_2 , and g^{r_v} of sk_j^i :

$$\hat{e}(g^{r_v}, g^{z/\pi(B) f_j^i(x)}) = g_t^{z/\pi(B) r_v f_j^i(x)}$$

$$\hat{e}(g^{r_v}, g^{z/\pi(B) f_j^i(y)}) = g_t^{z/\pi(B) r_v f_j^i(y)}$$

4. Compute $g_t^{z/\pi(B) r_v f_j^i(0)}$ from $\begin{cases} (I_{i,j}^v, g_t^{z/\pi(B) r_v f_j^i(I_{i,j}^v)}) \\ (x, g_t^{z/\pi(B) r_v f_j^i(x)}) \\ (y, g_t^{z/\pi(B) r_v f_j^i(y)}) \end{cases}$ by Lagrange Interpolation:

$$\begin{aligned} g_t^{z/\pi(B) r_v f_j^i(0)} &= (g_t^{z/\pi(B) r_v f_j^i(I_{i,j}^v)})^{\frac{(0-x)(0-y)}{(I_{i,j}^v-x)(I_{i,j}^v-y)}} \\ &\times (g_t^{z/\pi(B) r_v f_j^i(x)})^{\frac{(0-y)(0-I_{i,j}^v)}{(x-y)(x-I_{i,j}^v)}} \\ &\times (g_t^{z/\pi(B) r_v f_j^i(y)})^{\frac{(0-I_{i,j}^v)(0-x)}{(y-I_{i,j}^v)(y-x)}} \end{aligned}$$

5. Compute $C = \hat{e}(g^\rho, h)^{z/\pi(B)}$ from D , $g^{r_v f_j^i(0)} h^\rho$ of sk_j^i , and $g_t^{z/\pi(B) r_v f_j^i(0)}$:

$$C = \frac{\hat{e}(D, g^{r_v f_j^i(0)} h^\rho)}{g_t^{z/\pi(B) r_v f_j^i(0)}} = \hat{e}(D, h^\rho) \frac{\hat{e}(D, g^{r_v f_j^i(0)})}{g_t^{z/\pi(B) r_v f_j^i(0)}} = \hat{e}(D, h^\rho) = \hat{e}(g^{z/\pi(B)}, h^\rho)$$


6. Compute B from $E = \Phi(D, C) \oplus B$ and D, C :

$$B = E \oplus \Phi(D, C)$$

7. Compute $A = \hat{e}(g^\rho, h)^z$ from $C = \hat{e}(g^\rho, h)^{z/\pi(B)}$ and B :

$$A = C^{\pi(B)}$$

8. Compute Msg from $B = \Psi(A) \oplus Msg$ and A :

$$Msg = B \oplus \Psi(A)$$


7 Security

The scheme is IND-CCA2 secure with random oracle under the Gap-BDH assumption.

Theorem 1. *If Gap-BDH problem is (t, ϵ) -hard, the scheme is $(t-t', \epsilon+\epsilon')$ -IND-CCA2 secure in the random oracle model where t' is polynomial k -bounded and $\epsilon' \leq \frac{q_d(q_d+q_\Psi)}{p} = \frac{O(k^2)}{2^k}$ is negligible to k (q_d is the number of decryption query and q_Ψ is the number of hash- Ψ query).*

Proof. We reduce Gap-BDH problem to the scheme. Let H_1 , H_2 , Φ , and Ψ be random oracles, while π just be a collision resistant hash function. We denote C as the challenger of Gap-BDH problem, A as the adversary of the scheme, and S as the simulator between C and A . Then S is described as follows:

- Initial. S receives the instance (g, g^a, g^b, g^c) of BDH problem as input and the access right of decisional oracle O about BDH problem from C . Then A chooses an system identity ID and a set $T \subseteq U$ of users to attack.
- Setup. Let $T = \bigcup_{t=1}^r T_{x_t}^{i_t}$ be the union of disjoint subset differences, and S sets up the random oracles H_1 and H_2 depending on T . For each $T_{x_t}^{i_t}$ where $1 \leq t \leq r$, we assume that node x_t is in the j_t -th level from node i_t . Then H_1 and H_2 are defined as follows:
 - H_1 . For each subset difference $T_{x_t}^{i_t}$ where $1 \leq t \leq r$, we have the pair (i_t, j_t) and set

$$H_1(ID||i_t) = j_t + m_{i_t}(\log n - \lfloor \log i_t \rfloor + 1)$$

where m_{i_t} is a random number. Then for the remaining subtree T_i in the system, set

$$H_1(ID||i) = m_i(\log n - \lfloor \log i \rfloor + 1)$$

where m_i is a random number.

The setting assigns h' to the polynomial $f_{j_t}^{i_t}$ where $1 \leq t \leq r$ and assigns h to the other polynomials. If the input is not meaningful, H_1 just returns a random number under the consistence. Note that there doesn't exist polynomials f_j^i and $f_{j+\lfloor \log i \rfloor}^1$ that are both assign h' .

– H_2 . The setting of H_2 is composed of the two steps:

1. Set up the polynomials f_j^1 that are belong to the subtree T_1 .

If subset difference $T_{x'}^1 \subseteq T$ (assume that the node x' is in the j' -th level from node 1), set

$$f_{j'}^1(0) = m' - b \quad (1)$$

$$f_{j'}^1(x'') = x'' \quad (2)$$

$$f_{j'}^1(y'') = y'' \quad (3)$$

where $y' \neq 0$ is not in the system, and $(m', x'', y'') \in_R Z_p^3$ is chosen by S (S knows m' , x'' , and y'' , but he does not know b). Then compute $g^{f_{j'}^1(x'')} = g^{a_{j',2}^1 X^2 + a_{j',1}^1 X + (m'-b)}$ by Lagrange Interpolation over the exponent of g , and have

$$H_2(ID||1||j'|||s) = \begin{cases} g^{m'-b}, & \text{if } s = 0 \\ g^{a_{j',s}^1}, & \text{for } 1 \leq s \leq 2 \end{cases}$$

For the remaining polynomials $f_j^1(X) = a_{j,2}^1 X^2 + a_{j,1}^1 X + a_{j,0}^1$ of subtree T_1 , choose them randomly and have

$$H_2(ID||1||j|||s) = g^{a_{j,s}^1}, \text{ for } 0 \leq s \leq 2$$

2. Set up the information that is used for key derivation.

If subset difference $T_{x'}^1 \subseteq T$ (assume that the node x' is in the j' -th level from the node 1), then for $2 \leq i \leq 2^{j'} - 1$, set

$$H_2(ID||i||3|||s) = \begin{cases} g^{b+m_i''}, & \text{if } s = 0 \\ g^{-b/x'}, & \text{if } s = 1 \end{cases} \quad (4)$$

where $m_i'' \in_R Z_p$ is chosen by S . Then for each of the remaining subset difference $T_{x_t}^{i_t} \subseteq T$, set

$$H_2(ID||i_t||2||s) = \begin{cases} g^{m_i''-b}, & \text{if } s = 0 \\ g^{b/x_t}, & \text{if } s = 1 \end{cases} \quad (5)$$

where $m_i' \in_R Z_p$ is chosen by S . Moreover, for $2 \leq i \leq n-1$, set

$$H_2(ID||i||1||s) = g^{m_{i,s}}, \text{ for } 0 \leq s \leq 1 \quad (6)$$

where $m_{i,s} \in_R Z_p$ is chosen by S .

If the input is not meaningful, H_2 just returns a random number under the consistency.

Then S sets the public key

$$PK = \{ID, G, G_t, \hat{e}, g, g^p = g^a, h = g^s, h' = g^b, H_1, H_2, \pi, \Psi, \phi\}$$

where $s \in_R Z_p$ is chosen by S , and the master secret key $MSK = a$ is unknown. The secret key SK_v of the revoked user $u_v \in U \setminus T$ is computed as follows:

1. Set $g^{r_v} = g^{a+n_v}$ where $n_v \in_R Z_p$ is chosen by S .
2. Compute the secret key sk_j^1 where $1 \leq j \leq \log n$.

If subset difference $T_{x'}^1 \subseteq T$ (assume that the node x' is in the j' -th level from node 1), compute

$$\begin{aligned} sk_{j'}^1 &= \{g^{r_v}, g^{r_v f_{j'}^1(I_{1,j'}^v)}, g^{r_v f_{j'}^1(0)} h'^p\} \\ &= \{g^{a+n_v}, g^{(a+n_v)x''}, g^{(a+n_v)(m'-b)} g^{ba}\} \text{ by (1) and (2)} \\ &= \{g^{a+n_v}, g^{(a+n_v)x''}, g^{am'+n_v m'-bn_v}\} \end{aligned}$$

where $I_{1,j'}^v = x'$, otherwise the user u_v is not revoked. Then for the remaining

secret key sk_j^1 , compute

$$\begin{aligned} sk_j^1 &= \{g^{r_v}, g^{r_v f_j^1(I_{1,j}^v)}, g^{r_v f_j^1(0)} h^\rho\} \\ &= \{g^{a+n_v}, g^{(a+n_v)X_j}, g^{(a+n_v)Y_j} g^{as}\} \end{aligned}$$

where $X_j = f_j^1(I_{i,j}^u)$ and $Y_j = f_j^1(0)$ can be easily computed by S , since the polynomial f_j^1 is decided by S .

3. Compute the information kd_i where $i \in \text{Ancestor}(u_v)$.

If subset difference $T_{x'}^1 \subseteq T$ (assume that the node x' is in the j' -th level from the node 1), compute

$$\begin{aligned} (g^{r_v(b_{3,1}^i I_{i,j'}^v + b_{3,0}^i)}, g^{r_v b_{3,0}^i} \frac{1}{\Delta h^\rho}) &= (g^{(a+n_v)(\frac{-b}{x'} x' + b + m_i'')}, g^{(a+n_v)(b + m_i'')} g^{(s-b)a}) \text{ by (4)} \\ &= (g^{(a+n_v)m_i''}, g^{bn_v + am_i'' + n_v m_i'' + as}) \end{aligned}$$

where $I_{i,j'}^v = x'$, otherwise the user u_v is not revoked. Then for each of the remaining subset difference $T_{x_t}^{i_t} \subseteq T$ (assume that the node x_t is in the j_t -th level from the node i_t), compute

$$\begin{aligned} (g^{r_v(b_{2,1}^i I_{i_t,j_t}^v + b_{2,0}^i)}, g^{r_v b_{2,0}^i} \Delta h^\rho) &= (g^{(a+n_v)(\frac{b}{x_t} x_t + m_{i_t}' - b)}, g^{(a+n_v)(m_{i_t}' - b)} g^{(b-s)a}) \text{ by (5)} \\ &= (g^{(a+n_v)m_{i_t}'}, g^{am_{i_t}' + n_v m_{i_t}' - bn_v - as}) \end{aligned}$$

where $I_{i_t,j_t}^v = x_t$, otherwise the user u_v is not revoked. Moreover, compute

$$(g^{r_v b_{1,1}^i}, g^{r_v b_{1,0}^i}) = (g^{(a+n_v)m_{i,1}}, g^{(a+n_v)m_{i,0}}) \text{ by (6)}$$

Then S sends the public key PK and the secret key SK_v of the revoked user $u_v \in U \setminus T$ to A .

- Query Phase 1. A issues decryption query, hash- Φ query, and hash- Ψ query adaptively.

Then S responds to these queries as follows:

– Decryption query. According to the above setting, the ciphertext to T only consists of the part for h' . Otherwise, S can use the part for h of ciphertext to decrypt the message trivially, since $h = g^s$ is chosen by S and he knows the exponent s . S receives the query $Q_d = \{E', D', P'_{1,1}, P'_{1,2}, \dots, P'_{r,1}, P'_{r,2}\}$ and checks the query list of hash- Φ .

* If record $\Phi(D', C') = \phi'$ exists and $O(g, g^{\rho}, h', D', C') = 1$, computes

$$B' = E' \oplus \phi'$$

$$A' = C'^{\pi(B')}$$

$$Msg = B' \oplus \Psi(A')$$

and returns Msg .

* Otherwise, returns $Msg' \in_R G_t$ and sets $(E', D') \rightarrow Msg'$ into the watch list of hash- Φ .

– Hash- Φ query. S receives the query $Q_{\Phi} = (D_{\Phi}, C_{\Phi})$ and checks the query list of hash- Φ .

* If record $\Phi(D_{\Phi}, C_{\Phi}) = \phi$ exists, returns ϕ .

* Otherwise, returns $\Phi(D_{\Phi}, C_{\Phi}) = \phi' \in_R G_t$. Moreover, if $O(g, g^{\rho}, h', D_{\Phi}, C_{\Phi}) = 1$, for each record $(E', D') \rightarrow Msg'$ in the watch list such that $D' = D_{\Phi}$, computes

$$B' = E' \oplus \phi'$$

$$A' = C_{\Phi}^{\pi(B')}$$

$$\Psi(A') = B' \oplus Msg'$$

and sets $\Psi(A') = B' \oplus Msg'$ into the query list of hash- Ψ .

– Hash- Ψ query. S receives the query $Q_{\Psi} = A_{\Psi}$ and checks the query list of hash- Ψ .

- * If record $\Psi(A_\Psi) = \psi$ exists, returns ψ .
- * Otherwise, returns $\Psi(A_\Psi) = \psi' \in_R G_t$.

- Challenge. A chooses two messages M_0 and M_1 . Then C returns the ciphertext

$$Ctx = \{E', D' = g^c, P'_{1,1}, P'_{1,2}, \dots, P'_{r,1}, P'_{r,2}\}$$

where E' is a random element in G_t , and the shares $(P'_{1,1}, P'_{1,2}, \dots, P'_{r,1}, P'_{r,2})$ can be computed by S under the above setting (Like the computation for the second element $g^{r_v f_j^i(I_{i,j}^v)}$ of secret key sk_j^i in the simulation).

- Query Phase 2. It is almost same as query phase 1, except that A can not issue the decryption query about Ctx .
- Guess. S checks the query list of hash- Φ and sees whether there is a record $\Phi(D', C') = \phi'$ such that $O(g, g^p = g^a, h' = g^b, D' = g^c, C') = 1$. If there is, S returns $C' = \hat{e}(g, g)^{abc}$ as the answer of Gap-BDH problem. Otherwise, we claim that A can not have non-negligible advantage to crack our system, since that the simulation is almost perfect and the challenge ciphertext Ctx is independent to M_0 and M_1 .

Finally, we are going to analyze the efficacy of the simulation. It is almost perfect, except that hash- Ψ may not be consistent. Inconsistency is caused by the artificial settings to hash- Ψ in the manipulation of the watch list of Hash- Φ . The artificial setting to hash- Ψ may contradict another artificial setting or the query list of hash- Ψ . For example, we may face the situation that we have set $\Psi(A) = W$ before, but now we have to set $\Psi(A) = W'$ to complete the manipulation of watch list. The number of the artificial settings is bounded by the number of decryption query, that is q_d . Thus the probability of this situation occurring is at most $C_2^{q_d} \frac{1}{p}$. Moreover, the adversary can issue at most q_Ψ hash- Ψ queries. After issuing a hash- Ψ query, the adversary will leave a query record in the query list. Our settings may

contradict the existing records in the query list. The probability of this situation occurring is at most $C_1^{q_d} \frac{q_\Psi}{p}$.

The probability of inconsistency is negligible respect to k :

$$Pr[\text{Inconsistent}] \leq C_2^{q_d} \frac{1}{p} + C_1^{q_d} \frac{q_\Psi}{p} \leq \frac{q_d(q_d + q_\Psi)}{p} = \frac{O(k^2)}{2^k}$$

□



8 Remark

In our construction, the assignment of the specific value h and h' is not intuitive. Someone may ask that why do not use one specific value h only, or just like PK-SD-PI that assigns each polynomial f_j^i an unique value $h_{i,j}$ respectively. The reasons are as follows:

- Like PK-SD-PI. If we assign each polynomial f_j^i an unique value $h_{i,j}$ that is generated by the hash function, then the assigning values are all independent to each other. Thus for each subtree T_i , we need $O(\log n)$ information for changing the assigning value $h_{i,j}$ in the element $g^{r_v f_j^i(0)} h_{i,j}^\rho$ of the secret key during the key derivation from the secret keys belong to the subtree T_1 to the secret keys belong to the subtree T_i . Then we have to put $O(\log^2 n)$ information in the secret keys of each user. In order to reduce secret key size to $O(\log n)$, the information needed for changing the assigning value during the key derivation for each subtree T_i must be bounded by $O(1)$.
- Use one value only. If we only use one specific value h , then we must set $h = g^b$ in the simulation to feed the challenge input g^b to the adversary. Thus we have to set all the polynomials f_j^i with $f_j^i(0) = b + m_{i,j}$ where $m_{i,j} \in_R Z_p$ is chosen by us, because we can only compute the element $g^{r_v f_j^i(0)} h^\rho$ of the secret key in this way that $g^{r_v f_j^i(0)} h^\rho = (g^{-a})^{(b+m_{i,j})} (g^b)^a = (g^a)^{-m_{i,j}}$. But this setting makes all polynomials be secret to us, and we only know two function values of each polynomial. If the number of revoked users is greater than 2, we can not compute all the elements $g^{r_v f_{\log n}^1(x)}$ of the secret keys of the revoked users. We can only compute two secret keys for revoked users, but the number of revoked users is more than two.

9 Conclusion

We have proposed a fully collusion resistant public key broadcast encryption scheme that achieves $O(1)$ public key size, $O(\log n)$ secret key size, $O(r)$ ciphertext size, and $O(1)$ decryption time. Our scheme is the most efficient scheme in the existing broadcast encryption schemes.

We propose some open questions about broadcast encryption:

1. Is it possible to construct a more efficient scheme?
2. Is it possible to construct a scheme with the same efficiency but in normal model?



References

- [AFG⁺06] Nuttapon Attrapadung, Jun Furukawa, Takeshi Gomi, Goichiro Hanaoka, Hideki Imai, and Rui Zhang 0002. Efficient identity-based encryption with tight security reduction. In *CANS*, pages 19–36, 2006.
- [BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO*, pages 258–275, 2005.
- [Boy07] Xavier Boyen. Miniature cca2 pk encryption: Tight security without redundancy. In *ASIACRYPT*, pages 485–501, 2007.
- [BSNS05] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In *Public Key Cryptography*, pages 380–397, 2005.
- [DF03] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In *Public Key Cryptography*, pages 100–115, 2003.
- [FN93] Amos Fiat and Moni Naor. Broadcast encryption. In *CRYPTO*, pages 480–491, 1993.
- [HL06] Yong Ho Hwang and Pil Joong Lee. Efficient broadcast encryption scheme with log-key storage. In *Financial Cryptography*, pages 281–295, 2006.
- [HSaFZ06] Xinyi Huang, Willy Susilo, and Yi Mu amd Futai Zhang. Short (identity-based) strong designated verifier signature schemes. ISPEC, 2006. LNCS 3903.

- [HSaFZ08] Xinyi Huang, Willy Susilo, and Yi Mu and Futai Zhang. Short designated verifier signature scheme and its identity-based variant. *International Journal of Network Security*, 2008. Vol.6, No.1.
- [LT08] Yi-Ru Liu and Wen-Guey Tzeng. Public key broadcast encryption with low number of keys and constant decryption time. In *Public Key Cryptography*, pages 380–396, 2008.
- [MSLR04] Yi Mu, Willy Susilo, Yan-Xia Lin, and Chun Ruan. Identity-based authenticated broadcast encryption and distributed authenticated encryption. In *ASIAN*, pages 169–181, 2004.
- [NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *CRYPTO*, pages 41–62, 2001.
- [NP01] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In *FC '00: Proceedings of the 4th International Conference on Financial Cryptography*, pages 1–20, London, UK, 2001. Springer-Verlag.
- [SVG⁺08] S. Sharmila Deva Selvi, S. Sree Vivek, Ragavendran Gopalakrishnan, Naga Naresh Karuturi, and C. Pandu Rangan. Provably secure id-based broadcast signcryption (ibbsc) scheme. *Cryptology ePrint Archive*, Report 2008/225, 2008. <http://eprint.iacr.org/>.
- [TNJ⁺05] GOMI TAKESHI, ATTRAPADUNG NUTTAPONG, FURUKAWA JUN, ZHANG RUI, HANAOKA GOICHIRO, and IMAI HIDEKI. Cca-secure ibe scheme with tight security reduction based on the gap bdh assumption. *Proceedings of the Symposium on Information Theory and Its Applications*, 2005. VOL.28th;NO.Vol.1.

[YJCK04] Eun Sun Yoo, Nam-Su Jho, Jung Hee Cheon, and Myung-Hwan Kim. Efficient broadcast encryption using multiple interpolation methods. In *ICISC*, pages 87–103, 2004.

