Computers Math. Applic. Vol. 25, No. 4, pp. 39–46, 1993
Printed in Great Britain.

0097-4943/93 $5.00 + 0.00

A CONFERENCE KEY DISTRIBUTION SYSTEM BASED ON CROSS-PRODUCT

TZONG-CHEN WU AND YI-SHIUNG YEH
Institute of Computer Science and Information Engineering,
National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C.

(Received November 1991)

Abstract—Extended from the Diffie-Hellman public key distribution system (PKDS), we propose a conference key distribution system (CKDS) based on the cross-product operations on row vectors over a Galois field GF(P), where P is a prime number. In our CKDS, the chairperson computes a conference key CK and then embeds it to some public interpolating polynomials to let only the legal intended principals recover CK, while the illegal intended principals can not. From the public parameters, an intruder or any intended principal in the network does not know how many and who are the legal intended principals in the conference. Furthermore, since the construction of the CK does not interfere with the secret keys of the intended principals, any intended principals in the network has no useful information for revealing any other principals' secret keys. Besides, our CKDS can be implemented practically.

1. INTRODUCTION

In a computer system, we usually apply encryption techniques to safeguard transmitted information from anyone other than the legal receiver(s), for achieving privacy and secrecy. The so-called key distribution problem concerns how to secretly distribute an encryption/decryption key shared among a sender and the legal receiver(s) in advance. In 1979, Diffie and Hellman [1] first introduced the concept of public key distribution system (PKDS) for achieving such purpose. The Diffie-Hellman PKDS is described in the following.

Let x_i and x_j be two secret keys possessed by two communicating principals U_i and U_j, respectively. Let P be a large prime number and let α be a primitive element, $\bmod P$. Both P and α are known to U_i and U_j. For distributing a common secret key shared between principals U_i and U_j, U_i computes his public key y_i as

$$y_i = \alpha^{x_i} \pmod P,$$

and publishes it to U_j. Similarly, U_j computes his public key y_j as

$$y_j = \alpha^{x_j} \pmod P,$$

and publishes it to U_i. Thereafter, a common secret key K_{ij} shared between U_i and U_j is computed as

$$\begin{aligned} K_{ij} &= \alpha^{x_i x_j} \pmod P \\ &= {y_i}^{x_j} \pmod P \\ &= {y_j}^{x_i} \pmod P. \end{aligned}$$

That is, U_i can use his secret key x_i and U_js public key y_j to recompute K_{ij}, and U_j can use his secret key x_j and U_is public key y_i to reobtain K_{ij}. Once the common secret key K_{ij} has been distributed between U_i and U_j, they can communicate with each other secretly by sending the

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX messages enciphered by an available symmetric cryptosystem, such as DES, with the encryption key K_{ij}.

The Diffie-Hellman PKDS only allows two communicating principals to share a common secret key. With the progress in computer networks, we frequently want to admit any group of communicating principals to share a common conference key so that a secure multi-destination communication or holding a secure electronic conference can be achieved [2,3]. The key distribution system concerned with distributing a secret conference key shared among a group of communicating principals is referred to as the conference key distribution system (CKDS). In 1982 Ingemarson, Tang and Wong [4] generalized the Diffie-Hellman PKDS to a CKDS. Lately, Koyama and Ohta [5], and Okamoto and Tanaka [6] also proposed two identity-based CKDSs. However, these systems involve large computation for generating the conference key. Recently, Laih, Lee and Harn [7] proposed a threshold scheme and its application in designing a CKDS. The Laih-Lee-Harn scheme is based on the property of cross-product operations on row vectors. However, their proposed CKDS exhibits two potential problems:

(1) an illegal intended principal may fortuitously compute the conference key, and
(2) the amount of required storage used for public parameters grows with the square of the number of legal intended principals in the conference.

In this paper, we first extend the definitions of cross-product operations presented in the same paper [7] to be suitable over a Galois field GF(P), where P is a prime number. Based on some properties of cross-product operations on row vectors over GF(P), we shall propose another CKDS that can overcome the disadvantages stated above. In our CKDS, the required storage for public parameters is fixed proportionally to the number of intended principals in the networks. From the public parameters, an intruder or any intended principal does not know how many and who are the legal intended principals participating in the conference. In Section 2, some mathematical backgrounds are introduced. Our CKDS is presented in Section 3. In Section 4, the security analysis and computational complexity of our CKDS are discussed. Finally, conclusions are given in Section 5.

2. MATHEMATICAL BACKGROUNDS

In this section, we introduce some properties of cross-product operations on row vectors over a Galois field GF(P), where P is a prime number.

DEFINITION 2.1. *Let* $\mathbf{V}_i = (v_{i1}, v_{i2}, \dots, v_{in})$ *be a n-dimensional vector. We define the vector* $\mathbf{V}_i$ *over the Galois field GF*(P) *as*

$$\mathbf{V}_i \pmod P = (v_{i1} \bmod P, v_{i2} \bmod P, \dots, v_{in} \bmod P).$$

DEFINITION 2.2. *The cross-product of* $n-1$ *as linearly independent n-dimensional row vectors* $\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_{n-1}$ *over GF*(P) *is defined as*

$$\mathbf{V}_1 \times \mathbf{V}_2 \times \cdots \times \mathbf{V}_{n-1} \pmod P = \left(\begin{vmatrix} v_{12} & v_{13} & \cdots & v_{1n} \\ v_{22} & v_{23} & \cdots & v_{2n} \\ \vdots & \vdots & & \vdots \\ v_{n-1,2} & v_{n-1,3} & \cdots & v_{n-1,n} \end{vmatrix}, \right.$$

$$\left. \begin{vmatrix} v_{11} & v_{13} & \cdots & v_{1n} \\ v_{21} & v_{23} & \cdots & v_{2n} \\ \vdots & \vdots & & \vdots \\ v_{n-1,1} & v_{n-1,3} & \cdots & v_{n-1,n} \end{vmatrix}, \dots, \begin{vmatrix} v_{11} & v_{12} & \cdots & v_{1,n-1} \\ v_{21} & v_{22} & \cdots & v_{2,n-1} \\ \vdots & \vdots & & \vdots \\ v_{n-1,1} & v_{n-1,2} & \cdots & v_{n-1,n-1} \end{vmatrix} \right) \pmod P,$$

where $|\mathbf{M}|$ *means the determinant of the matrix* $\mathbf{M}$.

From the above definitions, consider $n = 3$ and let $\mathbf{V}_1$ and $\mathbf{V}_2$ be two linearly independent three-dimensional row vectors. The cross-product operations on row vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ have the following properties:

Proposition 2.1. *Let $\mathbf{A}$ be an $m \times 2$ matrix, $m \geq 2$, such that any two row vectors of $\mathbf{A}$ form a full rank square matrix. If*

$$\begin{pmatrix} \mathbf{K}_1 \\ \mathbf{K}_2 \\ \vdots \\ \mathbf{K}_m \end{pmatrix} = \mathbf{A} \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix} \pmod{P},$$

then $\mathbf{K}_i \times \mathbf{W} = c(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{P}$, for $i = 1, 2, \ldots, m$, where c is a constant and $\mathbf{W}$ is either $\mathbf{V}_1$ or $\mathbf{V}_2$.

Proof. Let $\mathbf{V}_1 = (v_{11}, v_{12}, v_{13})$, $\mathbf{V}_2 = (v_{21}, v_{22}, v_{23})$ and

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \vdots & \vdots \\ a_{m1} & a_{m2} \end{pmatrix}.$$

Then we have $\mathbf{K}_i = (a_{i1}\, v_{11} + a_{i2}\, v_{21}, a_{i1}\, v_{12} + a_{i2}\, v_{22}, a_{i1}\, v_{13} + a_{i2}\, v_{23}) \pmod{P}$. Without loss of generality, let $\mathbf{W} = \mathbf{V}_1$. We have

$$\begin{aligned}
\mathbf{K}_i \times \mathbf{W} \pmod{P} &= (a_{i1}\, v_{11} + a_{i2}\, v_{21},\ a_{i1}\, v_{12} + a_{i2}\, v_{22},\ a_{i1}\, v_{13} + a_{i2}\, v_{23}) \times (v_{11}, v_{12}, v_{13}) \pmod{P} \\
&= (v_{13}\,(a_{i1}\, v_{12} + a_{i2}\, v_{22}) - v_{12}(a_{i1}\, v_{13} + a_{i2}\, v_{23}), \\
&\quad v_{13}(a_{i1}\, v_{11} + a_{i2}\, v_{21}) - v_{11}(a_{i1}\, v_{13} + a_{i2}\, v_{23}), \\
&\quad v_{12}(a_{i1}\, v_{11} + a_{i2}\, v_{21}) - v_{11}(a_{i1}\, v_{12} + a_{i2}\, v_{22})) \pmod{P} \\
&= (a_{i2}(v_{13}\, v_{22} - v_{12}\, v_{23}),\ a_{i2}(v_{13}\, v_{21} - v_{11}\, v_{23}),\ a_{i2}(v_{12}\, v_{21} - v_{11}\, v_{22})) \pmod{P} \\
&= (-a_{i2})(v_{12}\, v_{23} - v_{13}\, v_{22},\ v_{11}\, v_{23} - v_{13}\, v_{21},\ v_{11}\, v_{22} - v_{12}\, v_{21}) \pmod{P} \\
&= (-a_{i2})(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{P} \\
&= c(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{P}, \qquad \text{for } c = (-a_{i2}).
\end{aligned}$$

∎

Proposition 2.2. *Let $\mathbf{V}_1 \times \mathbf{V}_2 \pmod{P} = (d_1, d_2, d_3)$ and $d_1 \neq 0$. Let $\mathbf{K}_i \times \mathbf{W} \pmod{P} = (e_1, e_2, e_3)$ and $\mathbf{K}_j \times \mathbf{W} \pmod{P} = (f_1, f_2, f_3)$ for $i \neq j$, where $\mathbf{W}$ is either $\mathbf{V}_1$ or $\mathbf{V}_2$. If the inverse of d_1, i.e., d_1^{-1}, over GF(P) exists, then*

(1) the inverse of e_1, i.e., e_1^{-1}, and the inverse of f_1, i.e., f_1^{-1}, over GF(P) exist;
(2) $(d_2\, d_1^{-1}, d_3\, d_1^{-1}) = (e_2\, e_1^{-1}, e_3\, e_1^{-1}) = (f_2\, f_1^{-1}, f_3\, f_1^{-1}) \pmod{P}$.

Proof. (1): From Proposition 2.1, we know

$$\begin{aligned}
\mathbf{K}_i \times \mathbf{W} \pmod{P} &= c_i\,(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{P} \\
&= c_i\,(d_1, d_2, d_3) \pmod{P} \\
&= (c_i\, d_1, c_i\, d_2, c_i\, d_3) \pmod{P} \\
&= (e_1, e_2, e_3).
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
\mathbf{K}_j \times \mathbf{W} \pmod{P} &= c_j\,(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{P} \\
&= c_j\,(d_1, d_2, d_3) \pmod{P} \\
&= (c_j\, d_1, c_j\, d_2, c_j\, d_3) \pmod{P} \\
&= (f_1, f_2, f_3).
\end{aligned}$$

However, $c_i \neq 0$ and $c_j \neq 0$, because $\mathbf{V}_1$ and $\mathbf{V}_2$ are linearly independent. Note that P is a prime number. Again, d_1^{-1} exists over GF(P) since $d_1 \neq 0$. Thus, $e_1^{-1} = (c_i\, d_1)^{-1} \pmod{P}$ and $f_1^{-1} = (c_j\, d_1)^{-1} \pmod{P}$ also exist, because $c_i\, d_1 \neq 0$ and $c_j\, d_1 \neq 0$.

(2): By normalizing the row vector (d_1, d_2, d_3) over GF(P), we have the normalized row vector

$$\mathbf{D} = (1,\ d_2\,d_1^{-1},\ d_3\,d_1^{-1}) \pmod P.$$

Again, by normalizing the row vector (e_1, e_2, e_3) over GF(P), we have

$$\begin{aligned}\mathbf{E} &= (1,\ e_2\,e_1^{-1},\ e_3\,e_1^{-1}) \pmod P\\ &= (1,\ c_i\,d_2\,(c_i\,d_1)^{-1},\ c_i\,d_3\,(c_i\,d_1)^{-1}) \pmod P\\ &= (1,\ d_2\,d_1^{-1},\ d_3\,d_1^{-1}) \pmod P,\end{aligned}$$

since $c_i\,c_i^{-1} = 1 \pmod P$. Similarly, we have

$$\begin{aligned}\mathbf{F} &= (1,\ f_2\,f_1^{-1},\ f_3\,f_1^{-1}) \pmod P\\ &= (1,\ c_j\,d_2\,(c_j\,d_1)^{-1},\ c_j\,d_3\,(c_j\,d_1)^{-1}) \pmod P\\ &= (1,\ d_2\,d_1^{-1},\ d_3\,d_1^{-1}) \pmod P,\end{aligned}$$

since $c_j\,c_j^{-1} = 1 \pmod P$. Therefore, $\mathbf{D} = \mathbf{E} = \mathbf{F}$. That is,

$$(d_2\,d_1^{-1},\ d_3\,d_1^{-1}) = (e_2\,e_1^{-1},\ e_3\,e_1^{-1}) = (f_2\,f_1^{-1},\ f_3\,f_1^{-1}) \pmod P. \qquad \blacksquare$$

The following example illustrates Proposition 2.1 and Proposition 2.2.

EXAMPLE 2.1. Let $P = 31$, $\mathbf{V}_1 = (2,3,5)$, $\mathbf{V}_2 = (1,2,4)$ and $\mathbf{A} = \begin{pmatrix} 2 & 3\\ 1 & 2\\ 1 & 4\end{pmatrix}$. Then we have

$$\begin{pmatrix}\mathbf{K}_1\\ \mathbf{K}_2\\ \mathbf{K}_3\end{pmatrix} = \begin{pmatrix} 2 & 3\\ 1 & 2\\ 1 & 4\end{pmatrix}\begin{pmatrix} 2 & 3 & 5\\ 1 & 2 & 4\end{pmatrix} \pmod{31} = \begin{pmatrix} 7 & 12 & 22\\ 4 & 7 & 13\\ 6 & 11 & 21\end{pmatrix}.$$

Thus, we have

$$\begin{aligned}\mathbf{V}_1 \times \mathbf{V}_2 \pmod{31} &= (2,3,1), \text{ and}\\ \mathbf{K}_1 \times \mathbf{V}_1 \pmod{31} &= (-6,-9,-3) \pmod{31}\\ &= (-3)\,(2,3,1) \pmod{31}\\ &= 28\,(2,3,1) \pmod{31}\\ &= 28\,(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{31}\\ &= (25,22,28).\end{aligned}$$

The reader may verify that

$$\begin{aligned}\mathbf{K}_2 \times \mathbf{V}_1 \pmod{31} &= 29\,(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{31}, \text{ and}\\ \mathbf{K}_3 \times \mathbf{V}_1 \pmod{31} &= 27\,(\mathbf{V}_1 \times \mathbf{V}_2) \pmod{31},\end{aligned}$$

from which Proposition 2.1 follows. Since P is prime, the inverse of 25 over GF(31) exists, and $25^{-1} = 5 \pmod{31}$. Thus, $(22 \cdot 25^{-1},\ 28 \cdot 25^{-1}) \pmod{31} = (17,16)$. Again, $2^{-1} = 16 \pmod{31}$. We have $(3 \cdot 2^{-1},\ 1 \cdot 2^{-1}) \pmod{31} = (17,16) = (22 \cdot 25^{-1},\ 28 \cdot 25^{-1}) \pmod{31}$, from which Proposition 2.2 follows. Based upon the above properties, we shall propose a CKDS in the next section.

3. OUR CKDS

Let there be $n+1$ intended principals $U_0, U_1, \ldots, U_n$ in a network system. Let P be a large prime number and α be a primitive element, mod P. Both P and α are known to all intended principals. When the network is set up, each principal U_i is initially assigned an identification number ID_i, a distinct secret key x_i and a public key y_i, where x_i and y_i are derived from the Diffie-Hellman public key system. Without loss of generality, let U_0 be the chairperson who wants to originate a secure conference. Let $U_1, U_2, \ldots, U_m$ be legal intended principals, and let $U_{m+1}, U_{m+2}, \ldots, U_n$ be illegal intended principals. For holding a secure conference, U_0 computes a common conference key CK to let only the legal intended principals recover it; while the illegal intended principals cannot. Once the conference key CK is retained by all participating members of the conference, they can broadcast the conference messages enciphered by CK. Thereafter, a secure conference is achieved. The algorithm for originating a secure conference by the chairperson U_0 is stated as follows.

Algorithm ORIGINATE

Input: 1. the secret key x_0 of U_0;
2. all public keys y_is of U_i, for $i = 1, 2, \ldots, n$.

Output: 1. a conference key CK;
2. a three-dimensional row vector $\mathbf{V}_1$, and interpolating polynomials $F_1(X)$, $F_2(X)$ and $F_3(X)$.

Step 1: Randomly choose two linear independent three-dimensional row vectors

$$\mathbf{V}_1 = (v_{11}, v_{12}, v_{13}) \text{ and } \mathbf{V}_2 = (v_{21}, v_{22}, v_{23}),$$

such that $v_{12}\, v_{23} \neq v_{13}\, v_{22}$.

Step 2: Compute a row vector $(d_1, d_2, d_3) = \mathbf{V}_1 \times \mathbf{V}_2 \pmod P$.

Step 3: Set the conference key $\mathrm{CK} = (d_2\, d_1^{-1},\, d_3\, d_1^{-1}) \pmod P$, where d_1^{-1} is the inverse of d_1 over GF(P).

Step 4: Randomly choose an $m \times 2$ matrix $\mathbf{A}$, such that any two row vectors in $\mathbf{A}$ form a full rank square matrix and then compute

$$\begin{pmatrix} \mathbf{K}_1 \\ \mathbf{K}_2 \\ \vdots \\ \mathbf{K}_m \end{pmatrix} = \mathbf{A} \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix} \pmod P,$$

where $\mathbf{K}_i = (k_{i1}, k_{i2}, k_{i3})$.

Step 5: Using an interpolation method [8], do the following:

(5.1): Construct the polynomial $F_1(X)$ over GF(P) by interpolating on points $(\mathrm{ID}_i, y_i^{x_0} \cdot k_{i1} \pmod P)$s and $(\mathrm{ID}_j, 0)$s, for $i = 1, 2, \ldots, m$ and $j = m+1, m+2, \ldots, n$.

(5.2): Construct the polynomial $F_2(X)$ over GF(P) by interpolating on points $(\mathrm{ID}_i, y_i^{x_0} \cdot k_{i2} \pmod P)$s and $(\mathrm{ID}_j, 0)$s, for $i = 1, 2, \ldots, m$ and $j = m+1, m+2, \ldots, n$.

(5.3): Construct the polynomial $F_3(X)$ over GF(P) by interpolating on points $(\mathrm{ID}_i, y_i^{x_0} \cdot k_{i3} \pmod P)$s and $(\mathrm{ID}_j, 0)$s, for $i = 1, 2, \ldots, m$ and $j = m+1, m+2, \ldots, n$.

Step 6: Publish $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$.

When the public parameters $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$ are retained, each intended principal U_i performs the following algorithm to recover the conference key CK.

Algorithm RECOVER-CK

Input: 1. secret key x_i of U_i;
2. public key y_0 of U_0;
3. public parameters $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$.

Output: the conference key CK.

Step 1: Compute

$$w_{i1} = F_1(\mathrm{ID}_i) \pmod P,$$
$$w_{i2} = F_2(\mathrm{ID}_i) \pmod P, \quad \text{and}$$
$$w_{i3} = F_3(\mathrm{ID}_i) \pmod P.$$

Step 2: If $w_{i1} = w_{i2} = w_{i3} = 0$, then stop, because U_i is an illegal intended principal to the conference.
Step 3: Compute $z_i = y_0^{x_i} \pmod P$.
Step 4: Compute

$$k_{i1} = w_{i1} \cdot z_i^{-1} \pmod P,$$
$$k_{i2} = w_{i2} \cdot z_i^{-1} \pmod P, \quad \text{and}$$
$$k_{i3} = w_{i3} \cdot z_i^{-1} \pmod P,$$

where z_i^{-1} is the inverse of z_i over GF(P).
Let $\mathbf{K}_i = (k_{i1}, k_{i2}, k_{i3})$.
Step 5: Compute $(e_{i1}, e_{i2}, e_{i3}) = \mathbf{K}_i \times \mathbf{V}_1 \pmod P$.
Step 6: Recompute the conference key CK as

$$\mathrm{CK} = (e_{i2} \cdot e_{i1}^{-1},\ e_{i3} \cdot e_{i1}^{-1}) \pmod P,$$

where e_{i1}^{-1} is the inverse of e_{i1}.

In Step 3 of algorithm ORIGINATE, the inverse of d_1 exists, since $v_{12}\,v_{23} \neq v_{13}\,v_{22}$ and P is prime. Similarly, the inverse of e_1 exists in Step 6 of algorithm RECOVER-CK. It is to see that if anyone is able to determine the vector (k_{i1}, k_{i2}, k_{i3}), then he can recover the conference key CK computed by the chairperson U_0. Further, by the Diffie-Hellman PKDS, we have

$$y_0^{x_i} \equiv y_i^{x_0} \pmod P.$$

Consequently, each legal intended principal U_i can use his secret key x_i and U_0s public key y_0, associated with the public parameters $F_1(X)$, $F_2(X)$, and $F_3(X)$, to recover CK. From Proposition 2.1 and Proposition 2.2, we see that the conference key CK chosen by the chairperson and the CK recovered by the legal intended principals are the same. When all the participating members of the conference have recovered the conference key CK, they can transmit conference messages enciphered by CK along with the broadcast links of the network. We will give examples to show how the algorithms ORIGINATE and RECOVER-CK work.

EXAMPLE 3.1 [ORIGINATE]. Let there be five principals U_0, U_1, U_2, U_3, and U_4 in the network. Let $P = 31$ and $\alpha = 7$. Initially, the identification numbers, secret keys and public keys are as $(\mathrm{ID}_0, x_0, y_0) = (0,3,2)$, $(\mathrm{ID}_1, x_1, y_1) = (1,7,28)$, $(\mathrm{ID}_2, x_2, y_2) = (2,6,4)$, $(\mathrm{ID}_3, x_3, y_3) = (3,4,14)$, and $(\mathrm{ID}_4, x_4, y_4) = (4,10,25)$, respectively. Suppose that U_0 wants to originate a secure conference, and U_1 and U_2 are legal intended principals, while U_3, U_4 are illegal intended principals. First, U_0 randomly chooses two row vectors, say $\mathbf{V}_1 = (2,3,5)$ and $\mathbf{V}_2 = (1,2,4)$, and computes

$$\mathbf{V}_1 \times \mathbf{V}_2 \pmod P = (2,3,5) \times (1,2,4) \pmod{31} = (2,3,1).$$

Then, U_0 computes the conference key CK as

$$\mathrm{CK} = (3 \cdot 2^{-1},\ 1 \cdot 2^{-1}) \pmod{31} = (17,16).$$

By performing Step 4 of ORIGINATE, let $\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix}$. And the row vectors for the legal intended principals U_1 and U_2 are

$$\mathbf{K}_1 = (7,12,22) \quad \text{and} \quad \mathbf{K}_2 = (4,7,13),$$

respectively. After that, three interpolating polynomials can be constructed by applying the secret key of U_0 and all intended principals' identification numbers ID_is and public keys y_is, as

$$F_1(X) = 20\,X^3 + 10\,X^2 + 27\,X + 2 \pmod{31},$$
$$F_2(X) = 30\,X^3 + 16\,X^2 + 18\,X + 15 \pmod{31}, \qquad \text{and}$$
$$F_3(X) = 19\,X^3 + 28\,X^2 + 10 \pmod{31}.$$

In order to let the legal intended principals have the ability to recover CK, U_0 broadcasts $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$ in the network.

EXAMPLE 3.2 [RECOVER-KEY]. Reconsider Example 3.1. We will show how the legal intended principal, say U_1, recovers the conference key CK. By Step 1 of algorithm RECOVER-CK, we have

$$w_{11} = F_1(\mathrm{ID}_1) \pmod P = F_1(1) \pmod{31} = 28,$$
$$w_{12} = F_2(\mathrm{ID}_1) \pmod P = F_2(1) \pmod{31} = 17, \qquad \text{and}$$
$$w_{13} = F_3(\mathrm{ID}_1) \pmod P = F_3(1) \pmod{31} = 26.$$

Since $w_{11} \neq 0$, $w_{12} \neq 0$, and $w_{13} \neq 0$, U_1 can confirm that he is a legal intended principal. From Step 3 of algorithm RECOVER-CK, U_1 computes

$$z_1 = y_0^{x_1} \pmod P = 2^7 \pmod{31} = 4,$$

and then retains $\mathbf{K}_i = (k_{11}, k_{12}, k_{13})$ as

$$k_{11} = 28 \cdot 4^{-1} \pmod{31} = 7,$$
$$k_{12} = 17 \cdot 4^{-1} \pmod{31} = 12, \qquad \text{and}$$
$$k_{13} = 26 \cdot 4^{-1} \pmod{31} = 22.$$

Next, U_1 computes

$$(e_{11}, e_{12}, e_{13}) = \mathbf{K}_1 \times \mathbf{V}_1 \pmod P = (7, 12, 22) \times (2, 3, 5) \pmod{31} = (25, 22, 28).$$

Thus, U_1 recovers CK as

$$\mathrm{CK} = (22 \cdot 25^{-1},\, 28 \cdot 25^{-1}) \pmod{31} = (17, 16),$$

which is identical to the CK generated by the chairperson U_0.

As to the illegal intended principal, say U_3, he computes

$$w_{31} = F_1(\mathrm{ID}_3) \pmod P = F_1(3) \pmod{31} = 0,$$
$$w_{32} = F_2(\mathrm{ID}_3) \pmod P = F_2(3) \pmod{31} = 0, \qquad \text{and}$$
$$w_{33} = F_3(\mathrm{ID}_3) \pmod P = F_3(3) \pmod{31} = 0.$$

Thus, U_3 cannot recompute the CK from the public parameters $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$. The reader may verify the performing of algorithm RECOVER-CK for U_2 and U_4.

4. SECURITY ANALYSIS AND DISCUSSIONS

From algorithm RECOVER-CK, it is easy to see that anyone who knows the secret key x_i can retain the row vector $\mathbf{K}_i$. And then he can recover CK by computing $\mathbf{K}_i \times \mathbf{V}_1$. However, the difficulty of computing x_i from y_i is based on the difficulty of computing a discrete logarithm over GF(P) [1]. Suppose the prime number P is represented as 200 bits, then taking logs mod P for determining x_i requires approximately 10^{30} operations. For a sufficiently large value of x_i, say 664 bits, the fast algorithms for computing the discrete logarithm function are intractable [9].

Further, Pholig and Hellman [10] pointed out that if $P-1$ has at least one large prime factor, then it is very difficult to compute discrete logarithms on mod P.

In our scheme, the secret keys of participating members of a secure conference do not interfere with the construction of $\mathbf{K}_i$s, which can be used to recover the conference key CK. Thus, any intended principals in the network have no useful information for revealing any other principals' secret keys. Again, for the construction of the interpolating polynomials $F_1(X)$, $F_2(X)$, and $F_3(X)$, we exclude the illegal intended principals U_js by interpolating on the points $(\mathrm{ID}_j, 0)$s. Therefore, our CKDS can prevent any illegal intended principals from fortuitously recomputing the conference key CK. The amount of storage for the public parameters $\mathbf{V}_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$ are $3(n+1)\log\lceil(P+1)\rceil$ bits, where $3n\lceil\log(P+1)\rceil$ bits are used for storing the coefficients of the $F_i(X)$s and the $3\lceil\log(P+1)\rceil$ bits are used for storing the row vector $\mathbf{V}_i$.

Next, we discuss the computational complexity of our CKDS. Denning [9] presented an efficient algorithm for computing the inverse of a number $x \bmod P$. The average number of divisions performed by his algorithm is approximately $(0.843 \ln P + 1.47)$. For computing the matrix $\mathbf{A}$ used in Step 4, a straightforward algorithm can be performed by $O(m)$ multiplications, where m is the number of legal intended principals. Thus, the complexity of our CKDS heavily depends on the construction of interpolating polynomials in Step 5 of algorithm ORIGINATE. By using the Lagrange formula, it requires n additions, $2n^2+2$ subtractions, $2n^2+n-1$ multiplications, and $n+1$ divisions, plus one modular operation to compute an interpolating polynomial $F(X)$ with degree of n [8]. As to evaluate the interpolating polynomial $F(X)$, we only require n multiplications, $2n$ additions, plus one modular operation by applying Horner's rule [8]. Thus, our CKDS is practical to implement.

5. CONCLUSIONS

We have extended the Diffie-Hellman PKDS to a CKDS. Our proposed CKDS is based on the properties of cross-product operations on row vectors. We have also shown that our CKDS is crypto-secure. The characteristics of our CKDS are:

(1) From the public parameters, an intruder or any intended principal in the network cannot know how many and who are the legal intended principals in the conference.

(2) The construction of the conference key does not interfere with the secrets of the intended principals. Thus, any intended principal in the network has no useful information for revealing any other principals' secret keys.

(3) Due to the computational complexity discussed in the previous section, our CKDS can be implemented practically.

REFERENCES

1. W. Diffie and M.E. Hellman, New directions in cryptography, *IEEE Trans. on Information Theory* **IT-22** (6), 644–654 (1976).
2. G.H. Chiou and W.T. Chen, Secure broadcasting using secure lock, *IEEE Trans. on Software Engineering* **SE-15** (8), 929–934 (1989).
3. C. Mitchell, Multi-destination secure electronic mail, *The Computer Journal* **32** (1), 13–15 (1989).
4. I. Ingemarson, D.T. Tang and C.K. Wong, A conference key distribution system, *IEEE Trans. on Information Theory* **IT-28** (5), 714–720 (1982).
5. K. Koyama and K. Ohta, Identity-based conference key distribution system in broadcasting networks, *Electronic Letters* **23** (10), 647–649 (1987).
6. E. Okamoto and K. Tanaka, Key distribution system based on identification information, *IEEE Journal of Selected Areas in Communications* **7** (4), 481–485 (1989).
7. C.S. Laih, J.Y. Lee and L. Harn, A new threshold scheme and its application in designing the conference key distribution cryptosystem, *Information Processing Letters* **32** (3), 95–99 (1989).
8. D.E. Knuth, *The Art of Computer Programming: Volume 2, Seminumerical Algorithms*, 2nd edition, Addison-Wesley, MA, (1981).
9. D.E. Denning, *Cryptography and Data Security*, Addison-Wesley, MA, (1982).
10. S. Pohlig and M.E. Hellman, An improved algorithm for computing logarithms over GF(P) and its cryptographic significance, *IEEE Trans. on Information Theory* **IT-24** (1), 106–110 (1978).