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 $z_i$ and $z_j$ be two secret keys possessed by two communicating principals $U_i$ and $U_j$, respectively. Let $P$ be a large prime number and let $\alpha$ be a primitive element, mod $P$. Both $P$ and $\alpha$ 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

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

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

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

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

$$K_{ij} = \alpha^{z_i z_j} \pmod{P} = y_i^{z_j} \pmod{P} = y_j^{z_i} \pmod{P}.$$ 

That is, $U_i$ can use his secret key $z_i$ and $U_j$'s public key $y_j$ to recompute $K_{ij}$, and $U_j$ can use his secret key $z_j$ and $U_i$'s 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
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, Koyarna 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 \( \text{GF}(P) \), where \( P \) is a prime number. Based on some properties of cross-product operations on row vectors over \( \text{GF}(P) \), 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 BACKGROUND

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

**Definition 2.1.** Let \( V_i = (v_{i1}, v_{i2}, \ldots, v_{in}) \) be a \( n \)-dimensional vector. We define the vector \( V_i \) over the Galois field \( \text{GF}(P) \) as

\[
V_i \pmod{P} = (v_{i1} \pmod{P}, v_{i2} \pmod{P}, \ldots, v_{in} \pmod{P}).
\]

**Definition 2.2.** The cross-product of \( n - 1 \) as linearly independent \( n \)-dimensional row vectors \( V_1, V_2, \ldots, V_{n-1} \) over \( \text{GF}(P) \) is defined as

\[
V_1 \times V_2 \times \cdots \times V_{n-1} \pmod{P} = \begin{vmatrix}
  v_{12} & v_{13} & \cdots & v_{1n} \\
  v_{22} & v_{23} & \cdots & v_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  v_{n-1,2} & v_{n-1,3} & \cdots & v_{n-1,n}
\end{vmatrix}
\]

where \( |M| \) means the determinant of the matrix \( M \).

From the above definitions, consider \( n = 3 \) and let \( V_1 \) and \( V_2 \) be two linearly independent three-dimensional row vectors. The cross-product operations on row vectors \( V_1 \) and \( V_2 \) have the following properties:
PROPOSITION 2.1. Let \( A \) be an \( m \times 2 \) matrix, \( m \geq 2 \), such that any two row vectors of \( A \) form a full rank square matrix. If

\[
\begin{pmatrix}
    K_1 \\
    K_2 \\
    \vdots \\
    K_m
\end{pmatrix} = A \begin{pmatrix}
    v_1 \\
    v_2
\end{pmatrix} \pmod{P},
\]

then \( K_i \times W = c(v_1 \times v_2) \pmod{P} \), for \( i = 1, 2, \ldots, m \), where \( c \) is a constant and \( W \) is either \( V_1 \) or \( V_2 \).

**Proof.** Let \( V_1 = (v_{11}, v_{12}, v_{13}) \), \( V_2 = (v_{21}, v_{22}, v_{23}) \) and

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

Then we have \( K_i \times W \pmod{P} = c(v_1 \times v_2) \pmod{P} \). Without loss of generality, let \( W = V_1 \). We have

\[
K_i \times W \pmod{P} = (a_{i1} \, v_{11} + a_{i2} \, v_{21}, a_{i1} \, v_{12} + a_{i2} \, v_{22}, a_{i1} \, v_{13} + a_{i2} \, v_{23}) \times (v_{11}, v_{12}, v_{13}) \pmod{P}
\]

\[
= (v_{13} (a_{i1} \, v_{12} + a_{i2} \, v_{22}) - v_{12} (a_{i1} \, v_{13} + a_{i2} \, v_{23}),
    v_{12} (a_{i1} \, v_{11} + a_{i2} \, v_{21}) - v_{11} (a_{i1} \, v_{12} + a_{i2} \, v_{22}) ) \pmod{P}
\]

\[
= (a_{i2} (v_{13} \, v_{22} - v_{12} \, v_{23}), a_{i2} (v_{13} \, v_{21} - v_{11} \, v_{23}), a_{i2} (v_{12} \, v_{21} - v_{11} \, v_{22})) \pmod{P}
\]

\[
= (-a_{i2} (v_{13} \, v_{23} - v_{12} \, v_{23}, v_{11} \, v_{23} - v_{13} \, v_{21}, v_{11} \, v_{22} - v_{12} \, v_{21}) \pmod{P}
\]

\[
= (-a_{i2} (V_1 \times V_2) \pmod{P})
\]

\[
= c(V_1 \times V_2) \pmod{P}, \quad \text{for } c = (-a_{i2}).
\]

PROPOSITION 2.2. Let \( V_1 \times V_2 \pmod{P} = (d_1, d_2, d_3) \) and \( d_1 \neq 0 \). Let \( K_i \times W \pmod{P} = (e_1, e_2, e_3) \) and \( K_j \times W \pmod{P} = (f_1, f_2, f_3) \) for \( i \neq j \), where \( W \) is either \( V_1 \) or \( V_2 \). 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

\[
K_i \times W \pmod{P} = c_i (V_1 \times V_2) \pmod{P}
\]

\[
= c_i (d_1, d_2, d_3) \pmod{P}
\]

\[
= (c_i \, d_1, c_i \, d_2, c_i \, d_3) \pmod{P}
\]

\[
= (e_1, e_2, e_3).
\]

Similarly, we have

\[
K_j \times W \pmod{P} = c_j (V_1 \times V_2) \pmod{P}
\]

\[
= c_j (d_1, d_2, d_3) \pmod{P}
\]

\[
= (c_j \, d_1, c_j \, d_2, c_j \, d_3) \pmod{P}
\]

\[
= (f_1, f_2, f_3).
\]

However, \( c_i \neq 0 \) and \( c_j \neq 0 \), because \( V_1 \) and \( V_2 \) 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

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

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

\[ E = (1, e_2 e_1^{-1}, e_3 e_1^{-1}) \pmod P \]

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

\[ F = (1, f_2 f_1^{-1}, f_3 f_1^{-1}) \pmod P \]

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

\[(d_2 d_1^{-1}, d_3 d_1^{-1}) \equiv (e_2 e_1^{-1}, e_3 e_1^{-1}) \equiv (f_2 f_1^{-1}, f_3 f_1^{-1}) \pmod P.
\]

The following example illustrates Proposition 2.1 and Proposition 2.2.

**Example 2.1.** Let \(P = 31\), \(V_1 = (2, 3, 5)\), \(V_2 = (1, 2, 4)\) and \(A = \begin{pmatrix} 2 & 3 \\ 1 & 2 \\ 1 & 4 \end{pmatrix}\). Then we have

\[
\begin{pmatrix} K_1 \\ K_2 \\ K_3 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 1 & 2 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} 2 & 3 & 5 \\ 1 & 2 & 4 \end{pmatrix} \pmod 31 = \begin{pmatrix} 7 & 12 & 22 \\ 4 & 7 & 13 \\ 6 & 11 & 21 \end{pmatrix}.
\]

Thus, we have

\[
V_1 \times V_2 \pmod {31} = (2, 3, 1), \quad K_1 \times V_1 \pmod {31} = (-6, -9, -3) \pmod {31} = (-3)(2, 3, 1) \pmod {31} = 28(2, 3, 1) \pmod {31} = 28(V_1 \times V_2) \pmod {31} = (25, 22, 28).
\]

The reader may verify that

\[
K_2 \times V_1 \pmod {31} = 29(V_1 \times V_2) \pmod {31}, \quad K_3 \times V_1 \pmod {31} = 27(V_1 \times V_2) \pmod {31},
\]

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 $\alpha$ be a primitive element, mod $P$. Both $P$ and $\alpha$ 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 $z_i$ and a public key $y_i$, where $z_i$ 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 $z_0$ 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 $V_1$, and interpolating polynomials $F_1(X)$, $F_2(X)$ and $F_3(X)$.

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

$$V_1 = (v_{11}, v_{12}, v_{13}) \text{ and } V_2 = (v_{21}, v_{22}, v_{23}),$$

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

Step 2: Compute a row vector $(d_1, d_2, d_3) = V_1 \times V_2 \pmod P$.

Step 3: Set the conference key $CK = (d_2 \alpha_1^{-1}, d_3 \alpha_1^{-1}) \pmod P$, where $\alpha_1^{-1}$ is the inverse of $d_1$ over GF($P$).

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

$$K = A \begin{pmatrix} V_1 \\ V_2 \end{pmatrix} \pmod P,$$

where $K = (k_{11}, k_{12}, k_{13})$.

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 $(ID_i, y_i^{\alpha_0})$,

$$k_{11} \pmod P \text{ and } (ID_j, 0)_s, \text{ for } i = 1, 2, \ldots, m \text{ and } j = m + 1, m + 2, \ldots, n.$$

(5.2): Construct the polynomial $F_2(X)$ over GF($P$) by interpolating on points $(ID_i, y_i^{\alpha_0})$,

$$k_{21} \pmod P \text{ and } (ID_j, 0)_s, \text{ for } i = 1, 2, \ldots, m \text{ and } j = m + 1, m + 2, \ldots, n.$$

(5.3): Construct the polynomial $F_3(X)$ over GF($P$) by interpolating on points $(ID_i, y_i^{\alpha_0})$,

$$k_{31} \pmod P \text{ and } (ID_j, 0)_s, \text{ for } i = 1, 2, \ldots, m \text{ and } j = m + 1, m + 2, \ldots, n.$$

Step 6: Publish $V_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$.

When the public parameters $V_1$, $F_1(X)$, $F_2(X)$, 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 $z_i$ of $U_i$;
2. public key $y_0$ of $U_0$;
3. public parameters $V_1$, $F_1(X)$, $F_2(X)$, and $F_3(X)$.
Output: the conference key $CK$. 
Step 1: Compute
\[ w_{i1} = F_1(ID_i) \pmod{P}, \]
\[ w_{i2} = F_2(ID_i) \pmod{P}, \]
\[ w_{i3} = F_3(ID_i) \pmod{P}. \]

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
\[ k_{i1} = w_{i1} \cdot z_i^{-1} \pmod{P}, \]
\[ k_{i2} = w_{i2} \cdot z_i^{-1} \pmod{P}, \]
\[ k_{i3} = w_{i3} \cdot z_i^{-1} \pmod{P}, \]
where \( z_i^{-1} \) is the inverse of \( z_i \) over \( GF(P) \).

Let \( K_i = (k_{i1}, k_{i2}, k_{i3}) \).

Step 5: Compute \( (e_{i1}, e_{i2}, e_{i3}) = K_i \times V_i \pmod{P} \).

Step 6: Recompute the conference key \( CK \) as
\[ CK = (e_{i2} \cdot e_{i1}^{-1}, e_{i3} \cdot e_{i1}^{-1}) \pmod{P}, \]
where \( e_{i1}^{-1} \) is the inverse of \( e_{i1} \).

In Step 3 of algorithm ORIGINATE, the inverse of \( d_1 \) exists, since \( v_{i2} v_{23} \neq v_{i3} v_{22} \) 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 \( z_i \) and \( U_0 \)'s 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 send 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
\[ (ID_0, x_0, y_0) = (0, 3, 2), \]
\[ (ID_1, x_1, y_1) = (1, 7, 28), \]
\[ (ID_2, x_2, y_2) = (2, 6, 4), \]
\[ (ID_3, x_3, y_3) = (3, 4, 14), \]
\[ (ID_4, x_4, y_4) = (4, 10, 25), \]
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 \( V_1 = (2, 3, 5) \) and \( V_2 = (1, 2, 4) \), and computes
\[ V_1 \times V_2 \pmod{P} = (2, 3, 5) \times (1, 2, 4) \pmod{31} = (2, 3, 1). \]

Then, \( U_0 \) computes the conference key \( CK \) as
\[ CK = (3 \cdot 2^{-1}, 1 \cdot 2^{-1}) \pmod{31} = (17, 16). \]

By performing Step 4 of ORIGINATE, let \( A = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix} \). And the row vectors for the legal intended principals \( U_1 \) and \( U_2 \) are
\[ K_1 = (7, 12, 22) \quad \text{and} \quad K_2 = (4, 7, 13), \]
respectively. After that, three interpolating polynomials can be constructed by applying the secret key of \(U_0\) and all intended principals' identification numbers \(ID_i\)'s and public keys \(y_is\), as

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

In order to let the legal intended principals have the ability to recover \(CK\), \(U_0\) broadcasts \(V_1, F_1(X), F_2(X), \) 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

\[
w_{11} = F_1(ID_1) \pmod{P} = F_1(1) \pmod{31} = 28,
\]
\[
w_{12} = F_2(ID_1) \pmod{P} = F_2(1) \pmod{31} = 17,
\]
\[
w_{13} = F_3(ID_1) \pmod{P} = F_3(1) \pmod{31} = 26.
\]

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

\[
z_1 = y_0^{e_1} \pmod{P} = 2^7 \pmod{31} = 4,
\]

and then retains \(K_i = (k_{i1}, k_{i2}, k_{i3})\) as

\[
k_{i1} = 28 \cdot 4^{-1} \pmod{31} = 7,
\]
\[
k_{i2} = 17 \cdot 4^{-1} \pmod{31} = 12,
\]
\[
k_{i3} = 26 \cdot 4^{-1} \pmod{31} = 22.
\]

Next, \(U_1\) computes

\[
(e_{i1}, e_{i2}, e_{i3}) = K_i \times V_1 \pmod{P} = (7, 12, 22) \times (2, 3, 5) \pmod{31} = (25, 22, 28).
\]

Thus, \(U_1\) recovers \(CK\) as

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

which is identical to the \(CK\) generated by the chairperson \(U_0\).

As to the illegal intended principal, say \(U_3\), he computes

\[
w_{31} = F_1(ID_3) \pmod{P} = F_1(3) \pmod{31} = 0,
\]
\[
w_{32} = F_2(ID_3) \pmod{P} = F_2(3) \pmod{31} = 0,
\]
\[
w_{33} = F_3(ID_3) \pmod{P} = F_3(3) \pmod{31} = 0.
\]

Thus, \(U_3\) cannot recompute the \(CK\) from the public parameters \(V_1, F_1(X), F_2(X), \) 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 \(z_i\) can retain the row vector \(K_i\). And then he can recover \(CK\) by computing \(K_i \times V_1\). However, the difficulty of computing \(z_i\) from \(y_i\) is based on the difficulty of computing a discrete logarithm over GF\(\left(\pmod{P}\right)\) [1]. Suppose the prime number \(P\) is represented as 200 bits, then taking logs mod \(P\) for determining \(z_i\) requires approximately \(10^{30}\) operations. For a sufficiently large value of \(z_i\), say 664 bits, the fast algorithms for computing the discrete logarithm function are intractable [9].
Further, Pohlig 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 \( K_i \)'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 \( U_j \)'s by interpolating on the points \((ID_j, 0)\)'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 \( V_1 \), \( F_1(X) \), \( F_2(X) \), and \( F_3(X) \) are \( 3(n + 1) \log((P + 1)) \) bits, where \( 3n \log(P + 1) \) bits are used for storing the coefficients of the \( F_i(X) \)'s and the \( 3 \log(P + 1) \) bits are used for storing the row vector \( V_i \).

Next, we discuss the computational complexity of our CKDS. Denning [9] presented an efficient algorithm for computing the inverse of a number \( z \) mod \( P \). The average number of divisions performed by his algorithm is approximately \( 0.843 \ln P + 1.47 \). For computing the matrix \( A \) 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