N-Party Encrypted Diffie-Hellman Key Exchange Using Different Passwords*

Jin Wook Byun and Dong Hoon Lee

Center for Information Security Technologies (CIST),
Korea University, Anam Dong, Sungbuk Gu, Seoul, Korea
{byunstar,donghlee}@korea.ac.kr

Abstract. We consider the problem of password-authenticated group *Diffie-Hellman* key exchange among N parties, N−1 clients and a single-server, using *different* passwords. Most password-authenticated key exchange schemes in the literature have focused on an authenticated key exchange using a *shared* password between a client and a server. With a rapid change in modern communication environment such as ad-hoc networks and ubiquitous computing, it is necessary to construct a secure end-to-end channel between clients, which is a quite different paradigm from the existing ones. To achieve this end-to-end security, only a few schemes of three-party setting have been presented where two clients exchange a key using their own passwords with the help of a server. However, up until now, no formally treated and round efficient protocols which enable group members to generate a common session key with clients' distinct passwords have been suggested.
In this paper we securely and efficiently extend three-party case to N-party case with a formal proof of security. Two provably secure N-party EKE protocols are suggested; N-party EKE-U in the unicast network and N-party EKE-M in the multicast network. The proposed N-party EKE-M is provable secure and provides forward secrecy. Especially, the scheme is of constant-round, hence scalable and practical.

Keywords: Password, Encrypted key exchange, N-party authentication, different password authentication, authenticated key exchange, dictionary attacks.

1 Introduction

To communicate securely over an insecure public network it is essential that secret keys are exchanged securely. An authenticated key exchange protocol allows two or more parties to agree on a common secret key over an insecure public network in a secure and authenticated manner. That is, no adversary can impersonate any participant during the protocol or learn any information about the value of the agreed secret. An authenticated key exchange protocol is essential for building secure communications between parties, and commonly used

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment).

in cryptographic protocols such as IPsec, SSL, et al. In a distributed system, a password-authenticated key exchange (PAKE) scheme is practical, where key exchange is done using only a human-memorable password. Actually, the setting such that users are only capable of storing human-memorable passwords is arisen more often in practice because of its mobility and efficiency. However a password has a low-entropy because it is drawn from a relatively small dictionary. This makes PAKE schemes susceptible to a dictionary attack. Even tiny amounts of redundancy in the flows of the protocol could be used by an adversary to mount a dictionary attack.

1.1 Related Works

Over the years, there have been much research on password-authenticated key exchange protocols. Most password-authenticated key exchange schemes in the literature have focused on the *shared password-authentication* (SPWA, for short) model which provides password-authenticated key exchange using a shared password between a client and a server [4, 7, 11, 17, 27, 28, 34, 41]. In the SPWA model two parties, client and server, use a shared password to generate a common session key and perform key confirmation. Bellovin and Merrit first proposed Encrypted Key Exchange (EKE) scheme secure against dictionary attacks [7]. EKE scheme has been basis for many of the subsequent works in the SPWA model. Recently Bresson et al. proposed a password-authenticated group *Diffie-Hellman* key exchange protocol which allows group members to generate a session key with a shared password [11].

Few schemes have been presented to provide password-authenticated key exchange between two clients with their different passwords [2, 19, 32, 33, 35]. In this *different password-authentication* (DPWA, for short) model two clients generate a common session key with their distinct passwords by the help of a server. This DPWA model is particularly well-suited for applications that require secure end-to-end communication between light-weight mobile clients. Steiner et al. proposed 3-party EKE which provides a password-authenticated key exchange between two clients using a single-server [35]. However, Ding and Horster showed that 3-party EKE protocol had a weakness under an undetectable on-line guessing attack [23]. In [32] Lin et al., pointed out that 3-party EKE was susceptible to an off-line password guessing attack, and proposed LSH-3PEKE protocol in which the server holds publicly known keys to prevent both attacks above. However, LSH-3PEKE protocol requires a high burden on the clients such that clients have to obtain and verify the public key of the server. Lin et al. presented LSSH-3PEKE protocol which is resistant to both off-line and undetectable on-line password guessing attacks but does not require server public keys [33]. Byun et al. [19] proposed two secure C2C-PAKE schemes; one for a cross-realm setting where two clients are in two different realms and hence there exist two servers involved, the other for a single-server setting where two clients are in the same realm. They have proved that the schemes are secure against all attacks considered. Unfortunately, the scheme was found to be flawed. Chen firstly pointed out that in the scheme with a cross-realm setting one malicious server can mount a dictionary attack to obtain the password of client who belongs to the other realm [20]. This attack was recently mentioned in [30, 40] too. Very recently, Abdalla et al. [2] give formal treatments and provable security for this three-party setting. They also present a generic construction of a three-party protocol based on any two-party authenticated key exchange protocol.

1.2 Our Contribution

In this paper we extend three-party case in the DPWA to *N-party* case which allows group members holding different passwords to agree on a group session key with the help of a single-server. In the SPWA, as mentioned above, Bresson et al. combined password-based authentication with group *Diffie-Hellman* key exchange protocols and presented a password-authenticated group *Diffie-Hellman* key exchange protocols when group members share a same password [11]. However the setting such that all group members have a same password is not practical since a password is not a common secret but a secret depending on an individual. Generally, in a mobile computing environment and distributed environment such as an ad-hoc network, the setting in which group mobile users have different passwords is more suitable. Furthermore, the scheme in [11] requires $O(n)$ rounds and $O(n)$ modular exponentiation per party to establish a group key. So the scheme is inefficient when the number of group members is large.

In our paper we consider two network environments, unicast network and multicast network. In the unicast network, we assume that one client can send messages one by one to the next client in one direction. To establish a common key among the clients in the unicast network, the keying messages should be conveyed to the all clients, and hence it is inevitable that the round complexity required is linear in the number of group members. In the unicast network we propose an N-party EKE-U scheme which requires $O(n)$ round complexity to establish a group key. Many applications in the mobile ad-hoc network (MANET) are based upon unicast communication [9, 21, 31]. For example, mission-critical matters such as emergency rescue and military operations may occur in the setting which is absent from fixed infrastructure and advanced multicast routing network. So N-party EKE-U scheme may be well suited for making a secure session in the unicast routing MANET [3, 29].

In the multicast network, any client can send messages to multiple recipients only in one round−one round includes all the messages that can be sent in parallel during the protocol. Therefore more round-efficient group key exchange protocols can be designed under multicast network than under unicast network. We design a constant round N-party EKE-M scheme in the multicast network. N-party EKE-M protocol can be used to assure multicast message confidentiality or multicast data integrity in the various multicast scenarios. For example, the scheme can be used under ad-hoc network environment such as BSS (Basic Service Set), which is a component of the IEEE 802.11 architecture [26]. IEEE 802.11 supports multicast and broadcast messages. In a BSS infrastructure network (that is, network using an access point), multicasts are only sent from an access point to mobile devices, while mobile devices are not allowed to send broadcast messages directly. The access point first makes a group master key (GMK), then derives a group transient key (GTK) from the GMK. After each pairwise secure connection between access point and mobile devices is established, the access point sends the GTK to mobile devices by the secure pairwise connection. Finally the mobile users generate a group key by using the GTK.

The process of group key generation in N-party EKE-M protocol is very similar to the process of the above one. Clients and a server in our scheme generate intermediate keys, then they generate a common group key by using the intermediate keys. Hence, our protocol is well-suitable for the protection of multicast and broadcast messages in the IEEE 802.11.

Other examples are collaborative works, personal area networking (PAN), video conference and multiplayer game. For these scenario, the proposed scheme may be used for attractive security method that establishes a session key to protect a session.

The proposed N-party EKE-U protocol allows clients to generate a common session key using their own different passwords in the unicast networks. We prove the proposed scheme is secure under the *Diffie-Hellman*-like assumptions such as group computational *Diffie-Hellman* [11, 36], computational *Diffie-Hellman*, and multi-decisional *Diffie-Hellman* [12]. Above all, to construct a secure N-Party EKE-U protocol in the DPWA model, we must consider insider adversaries who may perform dictionary attacks on one specific password using all other $n-1$ passwords. Preventing insider attacks is not an easy work. To prevent this serious attack in our model, keying materials generated by each client are blinded by the server using a *Transformation Protocol* (**TF**, for short) between the client and the server. In **TF**, the server first decrypts keying materials encrypted by the client's password, blinds and encrypts the materials with the other client's password. Actually all clients with distinct passwords can generate a common session key by executing **TF**. We construct a secure N-Party EKE-U protocol containing **TF** in the DPWA model by modifying a password-authenticated group *Diffie-Hellman* key exchange protocol in [11] which provides a group *Diffie-Hellman* key exchange in the SPWA model.

The proposed N-party EKE-M protocol is also strong against insider dictionary attacks. We prove that the protocol is secure under the assumption of computational *Diffie-Hellman.* As mentioned above, N-party EKE-M protcol requires only a constant number of rounds to establish a session key. Accurately, one round is demanded by clients, and two rounds are demanded by a server. Furthermore only 2 modular exponentiations are required by each client. The proposed N-party EKE-M is the first constant round and provable secure scheme with forward secrecy in the group DPWA model.

1.3 Organization

The remainder of this paper is organized as follows. In Section 2 we newly define our model and security for our proofs. In Section 3 we present an N-party EKE-U protocol in the unicast network and prove its security formally in the random oracle and ideal cipher models. In Section 4, we present an N-party EKE-M protocol in the multicast network and prove its security.

2 Model and Definition

In this section we formalize the adversary capabilities and the security definitions in N-party EKE protocols. We modify the adversary model and the definition of security defined in [4, 11, 12], based on priori work of [6, 8]. The model of [11] are designed to enable n clients to generate a session key with a priori shared password. In the model of [12], each client possesses a distinct strong secret key, not a password, to generate a session key. For the DPWA model of N-party case, we need a security model to allow n clients to possess different passwords. We construct the model by combining two models in [11, 12] to be suitable for the DPWA model of N-party case. That is, by giving different passwords to N parties and modifying adversary abilities of the model in [12], we construct DPWA security model for N-party case. As compared with the previous results a significant change in our model is that clients have different weak secrets. Notation of participants, session ID (SID) and partner ID (PID) are slightly changed. We also give a security definition of a DPWA protocol for N-party case according to the changed setting. Other security notions and adversary abilities are similar to those in the previous models [4, 6, 11].

2.1 Communication Model

Participants. We have two types of protocol participants, clients and a server. Let $ID = Clients \cup Server$ be a non-empty set of protocol participants, and the set ID remains stable. We assume that $Server$ consists of a single-server S, and $Clients=\{C_1, ..., C_{n-1}\}$ consists of identities of $n-1$ clients. Each client $C_i \in Clients$ has a secret password pw_i, and server S keeps password verifiers in its database. A client $C_i \in Clients$ may execute a key exchange protocol multiple times with different partners, and we denote the t-th instance of the protocol executed by entity C_i (S) as oracle C_i^t (S^t, respectively).

Algorithm. An N-party EKE protocol requires the following two probabilistic polynomial time algorithms.

– ***Password Generation Algorithm*** $\mathcal{G}_{pw}$ is given an input of 1^k, where k is a security parameter, and then provides each client $C_i \in Clients$ with password pw_i.
– ***Registration Algorithm*** $\mathcal{R}$ is given an input of a fixed client $C_i \in Clients$, and then registers each password pw_i of C_i at S

To define the notion of security, we define capabilities of an adversary. We allow the adversary to potentially control all communication in the network via access to a set of oracles as defined below. We consider an *experiment* in which the adversary asks queries to oracles, and the oracles answer back to the adversary. Oracle queries model attacks which the adversary may use in the real system. We consider the following types of queries in this paper.

- A query $\mathsf{Send}(C_i^t, M)$ is used to send a message M to instance C_i^t. When C_i^t receives M, it responds according to the key-exchange protocol. The adversary may use this query to perform *active* attacks by modifying and inserting the messages of the key-exchange protocol. Hence, impersonation attacks and man-in-the-middle attacks are possible using this query.
- A query $\mathsf{Execute}(Clients)$ represents passive eavesdropping of the adversary on an execution of the protocol between honest clients in $Clients$. Namely, all clients in $Clients$ execute the protocol without any interference from the adversary, and the adversary is given the resulting transcript of the execution. (Although the output of an $\mathsf{Execute}$ query can be simulated by repeated Send oracle queries, this particular query is needed to define a forward secrecy[1].)
- A query $\mathsf{Reveal}(C_i^t)$ models the *known key* attacks (or Denning-Sacco Attacks [22]) in the real system. The adversary is given the session key of the specified instance C_i^t.
- A query $\mathsf{Corrupt}(C_i)$ models exposure of the long-term password of C_i. The adversary is assumed to be able to obtain long-term passwords of clients, but cannot control the behavior of these clients directly (of course, once the adversary has asked a query $\mathsf{Corrupt}(C_i)$, the adversary may impersonate C_i in subsequent Send queries).
- A query $\mathsf{Test}(C_i^t)$ is used to define the advantage of an adversary. If C_i^t is a fresh oracle (defined in Section 2.3), then the oracle C_i^t flips a coin b. If b is 1, then a session key is returned. Otherwise, a string randomly drawn from a session key distribution is returned. The adversary is allowed to make a single Test query, at any time during the experiment.

2.2 Security Definition and Assumption

Session IDS, Partnering, Freshness. For unicast communication, there are well established definitions with respect to session IDS and partnering [11–13]. In the unicast network (for instance, in N-party EKE-U protocol), we directly use these notions defined in [12] without modification. For N-party EKE-M protocol, we define newly session IDS and partnering in the multicast communication as follows.

Definition 2.1 [Session IDS (SIDS)]. Suppose that S and $n-1$ clients, C_1,..,C_{n-1}, participate in an N-party EKE-M protocol. First, SIDS for any C_i is defined as SIDS(C_i) $= \{SID_{1,..,n-1} : C_1,..,C_{n-1} \in Clients\}$, where $SID_{1,..,n-1}$

[1] The definition of forward secrecy is introduced in the full version of paper, informally. Here we do not give a formal treatment on the forward secrecy for simplicity of security proofs. For a formal definition, refer to [4].

is the concatenation of all flows between oracles $C_1^{s_1}$,..,$C_{n-1}^{s_{n-1}}$ and S^u. The values of s_i and u are instances of the protocol executed by C_i and S for $1 \leq i \leq n-1$. Note that the multicast setting allows all the messages to be sent to all participating clients in parallel during the protocol, hence all participating clients can make SIDS. By using SIDS, we formally define **partnering** as follows.

Definition 2.2 [Partnering, PIDS]. The notion of **partnering** captures that the participating oracles, $C_1^{s_1}$,..,$C_{n-1}^{s_{n-1}}$ and S^u have jointly run a protocol in the multicast network. After running the protocol, the oracles $C_1^{s_1}$,..,$C_{n-1}^{s_{n-1}}$, S^u where $\{C_1, .., C_{n-1}\} \in$ *Clients*, *are partnering if* the all oracles accept with same session key and same SIDS. The partner IDS, PIDS($C_i^{s_i}$), is a set of clients' IDs which are partnered with oracles $C_i^{s_i}$ for $1 \leq i \leq n-1$.

Definition 2.3 [Freshness]. An oracle C_i^t *is fresh if* neither C_i^t nor one of its partners have been asked for a Reveal query after oracle C_i^t and its partners have computed a session key sk.

Security Definition. Now we formally define an advantage of an adversary against N-party EKE (NEKE) protocols. Our model is designed for different password-authenticated key exchange between clients using a single-server. The goal of our protocols is for clients to share a common session key sk which is known to nobody but participants under passive or active adversaries. The output session key must be indistinguishable from a random key by the adversary. The property that a session key is indistinguishable is a well-known security condition in key exchange protocols. However we note that if an adversary initiates m ($\leq |\mathcal{D}|$, which is a size of dictionary) instances of a password-authenticated key exchange protocol, and guesses an appropriate password in each initiation, then it will succeed in guessing the password with probability $m/|\mathcal{D}|$. So the given password-authenticated protocol is considered secure if the active adversary can not do significantly better than this trial bound. We define this formally as follows.

Definition 2.4 [NEKE Security] Consider the following experiment. Firstly, a password is assigned to each client by running the password generation algorithm $\mathcal{G}_{pw}$, and the password is registered in the server S by running the registration algorithm $\mathcal{R}$. Then an adversary $\mathcal{A}$ is run. It will interact with finite oracles by asking queries defined above during the experiment. The adversary $\mathcal{A}$ can ask a Test query to a fresh oracle only once at any time. When $\mathcal{A}$ asks a Test query, then experiment flips a coin for bit b. If it lands $b = 1$, a real session key is returned to the adversary. Otherwise, a random key is returned to the adversary. After $\mathcal{A}$ is given a random or a real key, $\mathcal{A}$ may ask other queries continuously and perform an on-line password guessing attacks adaptively. Eventually $\mathcal{A}$ guesses b, outputs the guessed bit b' and terminates the experiment. Let $\mathbf{Succ}_{\mathcal{A}}^{neke}$ be the event that $b' = b$. The session key advantage of $\mathcal{A}$ in attacking protocol P is defined as

$$\mathbf{Adv}_P^{neke}(\mathcal{A}) = 2Pr[\mathbf{Succ}_{\mathcal{A}}^{neke}] - 1.$$

A given N-party EKE protocol P *is secure if* the following condition is satisfied:

- **Indistinguishability:** For all probabilistic polynomial time adversary $\mathcal{A}$, every finite dictionary $\mathcal{D}$, and for all $m \leq |\mathcal{D}|$,

$$\mathbf{Adv}_P^{neke}(\mathcal{A}) \leq \frac{m}{|\mathcal{D}|} + \varepsilon(k).$$

where $\varepsilon(k)$ is a negligible function and k is a security parameter.

Computational Assumptions. For our formal proofs, we require well-known intractable assumptions such as the computational *Diffie-Hellman* (CDH), group computational *Diffie-Hellman* (GCDH), and multi-decisional *Diffie-Hellman* (MDDH) assumptions. The MDDH assumptions were shown to be reasonable by relating it to the DDH assumption in [11]. Formal descriptions and notions of these assumptions are presented in the full version of the paper.

3 N-Party EKE-U Protocol

In this section we describe an N-party EKE-U protocol which enables $n-1$ clients to generate a common session key, sk, by the help of a single-server S in the unicast network. Let G=$\langle g \rangle$ be cyclic group of prime order q. A common session key between clients is $sk = \mathcal{H}(Clients||K)$ where $K = (g^{x_1 \cdots x_{n-1}})^{v_1 \cdots v_n}$ for random values $x_1, .., x_{n-1}, v_1, .., v_n$. $\mathcal{H}$ is an ideal hash function from $\{0,1\}^*$ to $\{0,1\}^l$. We assume that $\mathcal{E}$ is an ideal cipher which is a random one-to-one function such that $\mathcal{E}_K : M \rightarrow C$, where $|M| = |C|$. Several methods to instantiate an ideal cipher for practical use are described in [16].

3.1 Description of N-Party EKE-U

In N-party EKE-U protocol, as illustrated in Fig. 1, participants (clients $C_1, ..,$ C_{n-1} and sever S) are arranged on a line, and there are two stages: up-flow and down-flow. In the up-flow, each client raises the received intermediate values to the power of its own secret and forwards the resulting values to the next client (or sever S in the last step) on the line. Note that this process makes N-party EKE-U protocol contributory. In the down-flow, server S computes a keying material and distributes it to each client, encrypted with the receiver's password. In the up-flow, each client receives the values encrypted with the password of the client who sends them. To decrypt the values, each client needs to execute a **TF** protocol with server S. As illustrated in Fig. 2, S in **TF** protocol plays a role of interpreter by transforming the message encrypted with other's password into the one encrypted with the requesting client's password. Since a client does not know other clients' passwords, the intervention of server S is inevitable in the DPWA model.

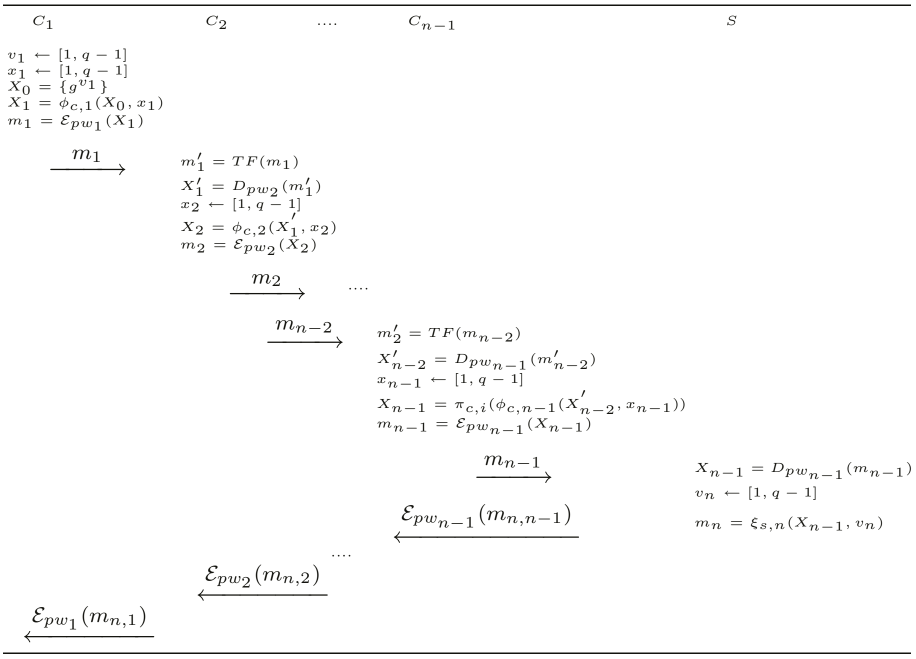
C_1
C_2
....
C_{n-1}
S
$v_1 \leftarrow [1, q-1]$
$x_1 \leftarrow [1, q-1]$
$X_0 = \{g^{v_1}\}$
$X_1 = \phi_{c,1}(X_0, x_1)$
$m_1 = \mathcal{E}_{pw_1}(X_1)$
$\xrightarrow{m_1}$
$m_1' = TF(m_1)$
$X_1' = D_{pw_2}(m_1')$
$x_2 \leftarrow [1, q-1]$
$X_2 = \phi_{c,2}(X_1', x_2)$
$m_2 = \mathcal{E}_{pw_2}(X_2)$
$\xrightarrow{m_2}$
....
$\xrightarrow{m_{n-2}}$
$m_2' = TF(m_{n-2})$
$X_{n-2}' = D_{pw_{n-1}}(m_{n-2}')$
$x_{n-1} \leftarrow [1, q-1]$
$X_{n-1} = \pi_{c,i}(\phi_{c,n-1}(X_{n-2}', x_{n-1}))$
$m_{n-1} = \mathcal{E}_{pw_{n-1}}(X_{n-1})$
$\xrightarrow{m_{n-1}}$
$X_{n-1} = D_{pw_{n-1}}(m_{n-1})$
$v_n \leftarrow [1, q-1]$
$m_n = \xi_{s,n}(X_{n-1}, v_n)$
$\xleftarrow{\mathcal{E}_{pw_{n-1}}(m_{n,n-1})}$
....
$\xleftarrow{\mathcal{E}_{pw_2}(m_{n,2})}$
$\xleftarrow{\mathcal{E}_{pw_1}(m_{n,1})}$

Fig. 1. N-party EKE-U.

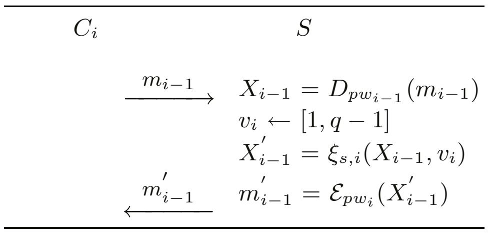
C_i
S
$\xrightarrow{m_{i-1}}$
$X_{i-1} = D_{pw_{i-1}}(m_{i-1})$
$v_i \leftarrow [1, q-1]$
$X_{i-1}' = \xi_{s,i}(X_{i-1}, v_i)$
$\xleftarrow{m_{i-1}'}$
$m_{i-1}' = \mathcal{E}_{pw_i}(X_{i-1}')$

Fig. 2. TF protocol.

In our protocol three types of functions are used. All clients or server contribute to generation of a common session key by using function $\phi_{c,i}$, $\pi_{c,i}$, and $\xi_{s,i}$ for positive integer i. The description of functions are as follows:

$$\phi_{c,i}(\{\alpha_1, .., \alpha_{i-1}, \alpha_i\}, x) = \{\alpha_1^x, .., \alpha_{i-1}^x, \alpha_i, \alpha_i^x\},$$
$$\pi_{c,i}(\{\alpha_1, .., \alpha_i\}) = \{\alpha_1, .., \alpha_{i-1}\},$$
$$\xi_{s,i}(\{\alpha_1, \alpha_2, .., \alpha_i\}, x) = \{\alpha_1^x, \alpha_2^x, .., \alpha_i^x\}.$$

We now describe the protocol in detail. In the up-flow, C_1 first chooses two numbers in $[1,\ q-1]$ randomly, calculates $X_1 = \phi_{c,1}(X_0, x_1) = \{g^{v_1},\ g^{v_1 x_1}\}$, and sends m_1 to C_2, which is an encryption of X_1 with the password pw_1. Upon receiving m_1, C_2 executes a **TF** protocol with server S. In the **TF** protocol,

C_2 sends m_1 to S. Then S selects a random number v_2 and calculates $X_1' = \xi_{s,2}(X_1, v_2)$. The purpose of using v_2 is to prevent an insider dictionary attack. Since S knows all clients' passwords, it can construct $m_1' = \mathcal{E}_{pw_2}(X_1')$ and sends it back to C_2. This is the end of **TF** protocol. On receiving $m_1' = \mathcal{E}_{pw_2}(X_1')$, C_2 decrypts it to get X_1'. Next C_2 chooses its own random number x_2 and computes $X_2 = \phi_{c,2}(X_1, x_2)$. Finally C_2 sends a ciphertext $m_2 = \mathcal{E}_{pw_2}(X_2)$ to the next client C_3. The above process is repeated up to C_{n-2}. The last client C_{n-1} chooses a random number x_{n-1}, and calculates $X_{n-1} = \pi_{c,n-1}(\phi_{c,n-1}(X_{n-2}^{'}, x_{n-1}))$. The function $\pi_{c,n-1}$ only eliminates the last element of $\phi_{c,n-1}(X_{n-2}^{'}, x_{n-1})$. Finally the client C_{n-1} encrypts X_{n-1} with pw_{n-1}, and sends the ciphertext, m_{n-1} to the server S. By using the function $\pi_{c,n-1}$, the protocol does not allow the server to get the last element of $\phi_{c,n-1}(X_{n-2}^{'}, x_{n-1})$, hence the server is not able to compute a session key.

In the down-flow, S first decrypts m_{n-1} to get X_{n-1}, chooses a random number v_n, and computes $m_n = \xi_{s,n}(X_{n-1}, v_n)$. For $1 \le i \le n-1$, let $m_{n,i} = (g^{x_1 \cdots x_{i-1} x_{i+1} \cdots x_{n-1}})^{v_1 \cdots v_n}$ which is the i-th component of m_n. S encrypts each $m_{n,i}$ with password pw_i and sends the resulting ciphertexts to the clients. Each client C_i decrypts $\mathcal{E}_{pw_i}(m_{n,i})$ to obtain $m_{n,i}$. Next, C_i computes session key $sk = \mathcal{H}(Clients||K)$ where $K = (m_{n,i})^{x_i} = (g^{x_1 \cdots x_{n-1}})^{v_1 \cdots v_n}$ and $Clients = \{C_1, ..., C_{n-1}\}$. In Fig. 3, we present an example of execution of the protocol with three clients and a sever S where $K = (g^{x_1 x_2 x_3})^{v_1 v_2 v_3 v_4}$ and $sk = \mathcal{H}(Clients||K)$.

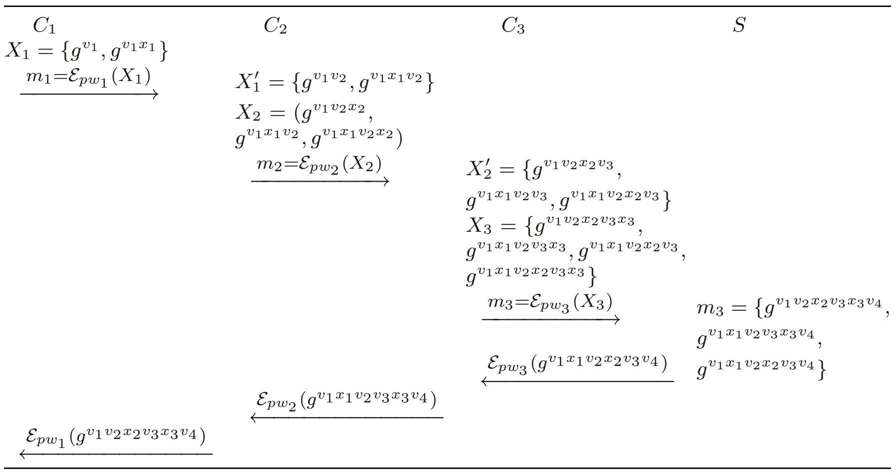
C_1
C_2
C_3
S
$X_1 = \{g^{v_1}, g^{v_1 x_1}\}$
$\xrightarrow{m_1=\mathcal{E}_{pw_1}(X_1)}$
$X_1' = \{g^{v_1 v_2}, g^{v_1 x_1 v_2}\}$
$X_2 = (g^{v_1 v_2 x_2}, g^{v_1 x_1 v_2}, g^{v_1 x_1 v_2 x_2})$
$\xrightarrow{m_2=\mathcal{E}_{pw_2}(X_2)}$
$X_2' = \{g^{v_1 v_2 x_2 v_3}, g^{v_1 x_1 v_2 v_3}, g^{v_1 x_1 v_2 x_2 v_3}\}$
$X_3 = \{g^{v_1 v_2 x_2 v_3 x_3}, g^{v_1 x_1 v_2 v_3 x_3}, g^{v_1 x_1 v_2 x_2 v_3}, g^{v_1 x_1 v_2 x_2 v_3 x_3}\}$
$\xrightarrow{m_3=\mathcal{E}_{pw_3}(X_3)}$
$m_3 = \{g^{v_1 v_2 x_2 v_3 x_3 v_4}, g^{v_1 x_1 v_2 v_3 x_3 v_4}, g^{v_1 x_1 v_2 x_2 v_3 v_4}\}$
$\xleftarrow{\mathcal{E}_{pw_3}(g^{v_1 x_1 v_2 x_2 v_3 v_4})}$
$\xleftarrow{\mathcal{E}_{pw_2}(g^{v_1 x_1 v_2 v_3 x_3 v_4})}$
$\xleftarrow{\mathcal{E}_{pw_1}(g^{v_1 v_2 x_2 v_3 x_3 v_4})}$

Fig. 3. An example of N-party EKE-U (N=4).

3.2 Mutual Authentication

If clients want to make sure that other clients really have computed the agreed key, then key confirmation for all clients can be incorporated into the key exchange protocol, so-called mutual authentication. That is, mutual authentication provides an assurance that all clients participated in the protocol have actually computed the agreed key. To add mutual authentication to our scheme, we use an additional *authenticator* structure described in [12]. The authenticator is computed as the hash of the session key *sk* and some other information such as client index. The client C_i makes an authenticator $Auth_i = \mathcal{H}(i||sk)$, where $sk = \mathcal{H}(Clients||K)$, and then sends $Auth_i$ to all clients. The client C_i verifies the received $n-2$ authenticators $Auth_i$ $(i \neq j)$.

In [12], Bresson et al. design a generic transformation which transforms a group key exchange protocol into a protocol with mutual authentication by using the authenticator structure. They also proved that the transformation is secure in the random-oracle model. That is, authenticator transformation for mutual authentication preserves the indistinguishability security of the original session key. So, in the paper, we do not consider a security of mutual authentication.

3.3 Security of N-Party EKE-U

In this section we prove that the proposed N-party EKE-U protocol is secure under the group computational *Diffie-Hellman* and multi-decisional *Diffie-Hellman* assumptions in the ideal hash and ideal cipher paradigms. We show that N-party EKE-U satisfies that an advantage of an adversary $\mathcal{A}$ in attacking session key **Indistinguishability** security is bounded above by $\mathcal{O}(q_s/|\mathcal{D}|) + \varepsilon(k)$, for some negligible function $\varepsilon(\cdot)$. The first term is an advantage from on-line guessing attacks. As mentioned in Section 2, the on-line attacks can not be avoided and hence the success probability of the on-line attacks may be considered as a lower bound of an advantage of any adversary. By **Theorem 3.1** bellow, we show that the best thing any adversary can do is only on-line password-guessing attacks.

Theorem 3.1 *P is an N-party EKE-U protocol of Fig. 1, and passwords are chosen from a finite dictionary of size* $|\mathcal{D}|$. *Let* $\mathcal{A}$ *be a probabilistic polynomial time adversary which asks* q_s *send,* q_h *hash, and* $q_{\mathcal{E}}$ *encryption/decryption queries. Then*

$$\mathbf{Adv}_P^{neke}(\mathcal{A}) \leq \frac{q_{\mathcal{E}}^2}{(q-1)} + \frac{q_s}{|\mathcal{D}|} + 4n \cdot \mathbf{Adv}_{\mathcal{D}}^{mddh}(T_{\mathcal{D}}) + 2q_h \cdot \mathbf{Adv}_{\Delta}^{gcdh}(T_{\Delta}).$$

where $T_{\mathcal{D}}$ *and* T_{Δ} *is polynomial running time of* **MDDH** *adversary algorithm* $\mathcal{A}_{\mathcal{D}}$ *and* **GCDH** *adversary algorithm* $\mathcal{A}_{\Delta}$ *such that* $T_{\Delta} \leq T + (q_s + q_{\mathcal{E}})(\tau_G + \tau_{\mathcal{E}})$ *and* $T_{\mathcal{D}} \leq T + (q_s + q_{\mathcal{E}})(\tau_G + \tau_{\mathcal{E}})$*, respectively.* τ_G *and* $\tau_{\mathcal{E}}$ *are computational time for exponentiation and encryption.* q *is the prime order of* G.

Theorem 3.2 *An N-party EKE-U protocol P provides a forward secrecy under the group computational Diffie-Hellman assumption.*

The proofs of the theorems will appear in the full version of the paper.

4 N-Party EKE-M Protocol

In this section we design an N-Party EKE-M protocol in the multicast channel. We assume that a server S keeps clients' passwords pw_1,..,pw_{n-1} and all entities know participating parties in a session in advance. Our N-party EKE-M protocol consists of two rounds. In the first round we run a well-known 2-party password-authenticated key exchange scheme to set up secure channels between all clients $\in$ *Clients* and a server. In the second round, the server distributes a common keying value on the secure channel. Finally all clients generate a group session key using the common keying value. In Fig. 4, we illustrate a general framework of our construction.

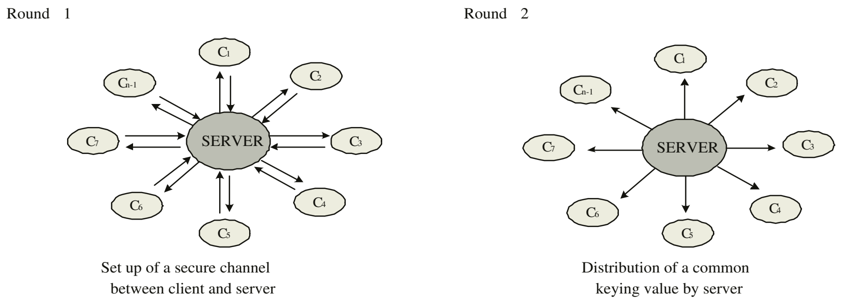
Round 1
C1
C2
C3
C4
C5
C6
C7
Cn-1
SERVER
Set up of a secure channel between client and server
Round 2
C1
C2
C3
C4
C5
C6
C7
Cn-1
SERVER
Distribution of a common keying value by server

Fig. 4. Framework of N-party EKE-M.

Concretely, to set up secure channels, we use a 2-party encrypted key exchange scheme BPR [4]. The scheme assumes the ideal cipher, hence our whole security results are naturally based on the ideal cipher model. If we adopt a 2-party password-authenticated key exchange scheme such as KOY [28] whose security is in the standard model, then our construction does not need the ideal cipher model in the proofs. However, as compared with BPR protocol, KOY protocol has disadvantages in computational and communicational aspects while it has advantage in the security assumption. For efficiency we adopt the BPR protocol to set up secure channels.

4.1 Description of N-Party EKE-M

Notations. Let G=$\langle g \rangle$ be cyclic group of prime order q. $sk_i(= \mathcal{H}_1(sid'||g^{x_i s_i}))$ is an ephemeral key generated between S and client C_i in the first round, where $sid' = \mathcal{E}_{pw_1}(g^{x_1})||\mathcal{E}_{pw_2}(g^{x_2})||...||\mathcal{E}_{pw_{n-1}}(g^{x_{n-1}})$. A common group key between clients is $sk = \mathcal{H}_2(SIDS||N)$, where $SIDS = sid'||sk_1 \oplus N||sk_2 \oplus N||...||sk_{n-1} \oplus N$ and N is a random value chosen from $[1, q-1]$.

Descriptions. In the first round, the single server S sends $\mathcal{E}_{pw_i}(g^{s_i})$ to $n-1$ clients concurrently. Simultaneously each client C_i, $1 \leq i \leq n-1$, also sends

$\mathcal{E}_{pw_i}(g^{x_i})$ to the single-server concurrently in the first round. After the first round finished S and C_i, $1 \leq i \leq n-1$, share an ephemeral *Diffie-Hellman* key, $sk_i = \mathcal{H}_1(sid' || g^{x_i s_i})$.

In the second round, S selects a random value N from $[1, q-1]$ and hides it by exclusive-or operation with the ephemeral key sk_i. S sends $N \oplus sk_i$ to C_i, $1 \leq i \leq n-1$, concurrently. After the second round finished all clients can get a random secret N using its sk_i, and generate a common session key, $sk = \mathcal{H}_2(SIDS || N)$. To add the mutual authentication (key confirmation) to N-party EKE-M protocol, we can use the additional *authenticator* structure described in [12], as mentioned in Section 3.

4.2 Security Theorem of N-Party EKE-M

We prove that an advantage of session key that an adversary $\tilde{\mathcal{A}}$ tries to get is negligible under the computational *Diffie-Hellman* assumption in both the ideal cipher and random oracle models. In the proof of **Theorem 4.1** we define sequences of experiments and get the advantage of the adversary in each experiment. At the end of proof we show the advantage of adversary in the real experiment is negligible. The security result is similar to the result of N-party EKE-U protocol except that the term about advantage of **GCDH** is replaced with the term about advantage of **CDH**. This result means that the best thing any adversary does in the multicast channel is also on-line guessing attacks. The security theorem is as follows.

Theorem 4.1 *P' is an N-party EKE-M protocol of Fig. 5, and passwords are chosen from a finite dictionary of size $|\mathcal{D}|$. Let $\tilde{\mathcal{A}}$ be a probabilistic polynomial time adversary which asks $\tilde{q}_s$ send, $\tilde{q}_h$ hash, and $\tilde{q}_{\mathcal{E}}$ encryption/decryption queries. Then*

$$\mathbf{Adv}_{P'}^{neke}(\tilde{\mathcal{A}}) \leq \frac{\tilde{q}_{\mathcal{E}}^2}{(q-1)} + \frac{2\tilde{q}_s}{|\mathcal{D}|} + 2\tilde{q}_h \cdot \mathbf{Adv}_{\tilde{\Delta}}^{cdh}(T_{\tilde{\Delta}})$$

where $T_{\tilde{\Delta}}$ is polynomial running time of **CDH** *adversary algorithm $\mathcal{A}_{\tilde{\Delta}}$ such that $T_{\tilde{\Delta}} \leq T + (\tilde{q}_s + \tilde{q}_{\mathcal{E}})(\tilde{\tau}_G + \tilde{\tau}_{\mathcal{E}})$. $\tilde{\tau}_G$ and $\tilde{\tau}_{\mathcal{E}}$ are computational time for exponentiation and encryption. q is the prime order of G.*

	S	C_1	C_2	...	C_{n-1}
Round 1	$s_i \leftarrow [1, q-1]$	$x_1 \leftarrow [1, q-1]$	$x_2 \leftarrow [1, q-1]$	...	$x_{n-1} \leftarrow [1, q-1]$
	$\mathcal{E}_{pw_i}(g^{s_i})$	$\mathcal{E}_{pw_1}(g^{x_1})$	$\mathcal{E}_{pw_2}(g^{x_2})$	...	$\mathcal{E}_{pw_{n-1}}(g^{x_{n-1}})$
	$N \leftarrow [1, q-1]$				
Round 2	$sk_1 \oplus N \| ... \| sk_{n-1} \oplus N$				

Fig. 5. N-party EKE-M.

Theorem 4.2 *An N-party EKE-M protocol P' provides a forward secrecy under the computational Diffie-Hellman assumption.*

The proofs of the theorems will appear in the full version of the paper.

5 Conclusion and Further Research

In this paper we first presented two secure N-party EKE protocols in which clients are able to generate a common group session key only with their different passwords, and proved its security based on the group computational *Diffie-Hellman*, computational *Diffie-Hellman*, and multi-desional *Diffie-Hellman* assumptions. Further works may be to design N-Party EKE protocols in dynamic scenario and prove our schemes in the standard assumption.

Acknowledgement

We thank Dr. Ik Rae Jeong for helpful discussions about the security proofs and useful comments on an early version of this paper. We also thank anonymous referees for their valuable comments.

References

1. M. Abdalla, M. Bellare, and P. Rogaway, “The oracle diffie-hellman assumptions and an analysis of DHIES”, *In proceedings of CT-RSA 2001*, LNCS Vol. 2020, pp. 143-158, Springer-Verlag, 2001.
2. M. Abdalla, P. Fouque, and D. Pointcheval, “Password-based authenticated key exchange in the three-party setting”, *In proceedings of PKC’05*, LNCS Vol. 3386, pp. 65-84, Springer-Verlag, 2005.
3. R. Beraldi and R. Baldoni, “Unicast routing techniques for mobile ad hoc networks”, ISBN:0-8493-1322-5, CRC Press, Inc. 2003.
4. M. Bellare, D. Pointcheval, and P. Rogaway, “Authenticated key exchange secure against dictionary attacks”, *In proceedings of Eurocrypt’00*, LNCS Vol.1807, pp. 139-155, Springer-Verlag, 2000.
5. M. Bellare and P. Rogaway, “Random oracles are practical: a paradigm for designing efficient protocols”, *In proceedings of the First ACM Conference on Computer and Communications Security, ACM*, 1995
6. M. Bellare and P. Rogaway, “Entity authentication and key distribution”, *In proceedings of Crypto’93*, LNCS Vol. 773, pp. 232-249. Springer-Verlag, 1994.
7. S. Bellovin and M. Merrit, “Encrypted key exchange: password based protocols secure against dictionary attacks”, *In proceedings of the Symposium on Security and Privacy*, pp.72-84, IEEE, 1992.
8. S. Blake-Wilson, D. Jhonson, and A. Menezes, “Key agreement protocols and their security analysis”, *In proceedings of IMA international conference*, LNCS Vol. 1361, pp. 30-45, Springer-Verlag, 1997.
9. C. Basile, M-O. Killijian, and D. Powell, “A survey of dependability issues in mobile wireless networks”, Technical Report, LAAS CNRS Toulouse, France, February, 2003. Availabe at http://www.crhc.uiuc.edu/∼basilecl/papers/mobile.ps

10. C. Boyd and A. Mathuria, “Key establishment protocols for secure mobile communications : a selective survey”, *In proceedings of ACISP 98'*, LNCS Vol. 1438, pp. 344-355, Springer-Verlag, 1998.
11. E. Bresson, O. Chevassut, and D. Pointcheval, “Group diffie-hellman key exchange secure against dictionary attacks”, *In proceedings of Asiacrypt'02*, LNCS Vol. 2501, pp. 497-514, Springer-Verlag, 2002.
12. E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater, “Provably authenticated group diffie-hellman key exchange”, *In proceedings of 8th ACM Conference on Computer and Communications Security*, pp. 255-264, 2001.
13. E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater, “Provably authenticated group diffie-hellman key exchange in the dynamic case”, *In proceedings of Asiacrypt'01*, LNCS Vol. 2248, pp. 290-309, Springer-Verlag, 2001.
14. E. Bresson, O. Chevassut, and D. Pointcheval, “Dynamic group diffie-hellman key exchange under standard assumptions”, *In proceedings of Eurocrypt'02*, LNCS Vol. 2332, pp. 321-336, 2002.
15. E. Bresson, O. Chevassut, and D. Pointcheval, “The group diffie-hellman problems”, *In proceedings of SAC'02*, LNCS Vol. 2595, pp. 325-338, Springer-Verlag, 2002.
16. J. Black and P. Rogaway, “Ciphers with arbitrary finite domains”, *In proceedings of CT-RSA conference*, LNCS Vol. 2271, pp. 114-130, Springer-Verlag, 2001.
17. V. Boyko, P. MacKenzie, and S. Patel, “Provably secure password-authenticated key exchange using diffie-hellman”, *In proceedings of Eurocrypt'00*, LNCS Vol. 1807, pp. 156-171, Springer-Verlag, 2000.
18. M. Burmester and Y. Desmedt, “A secure and efficient conference key distribution system”, *In proc. of Eurocrypt'94* LNCS VOL. 950, pp. 275-286, Springer-Verlag, 1994.
19. J. Byun, I. Jeong, D. Lee, and C. Park, “Password-authenticated key exchange between clients with different passwords”, *In proceedings of ICICS'02*, LNCS Vol. 2513, pp. 134-146, Springer-Verlag, 2002.
20. L. Chen, “A Weakness of the Password-Authenticated Key Agreement between Clients with Different Passwords Scheme”, ISO/IEC JTC 1/SC27 N3716.
21. C. Cordeiro and D. Agrawal, “Mobile ad hoc networking”, *Tutorial/Short Course in 20 th Brazilian Symposium on Computer Networks*, pp. 125-186, May, 2002.
22. D. Denning and G. Sacco, “Timestamps in key distribution protocols”, *In Communications of the ACM*, Vol. 24, No. 8, pp. 533-536, 1981.
23. Y. Ding and P. Horster, “Undetectable on-line password guessing attacks”, *In ACM Operating Systems Review*, Vol. 29, No. 4, pp. 77-86, 1995.
24. O. Goldreich and Y. Lindell, “Session-key generation using human passwords only”, *In proceedings of Crypto'01*, LNCS Vol. 2139, pp. 408-432, Springer-Verlag, 2001.
25. S. Halevi and H. Krawczyk, “Public-key cryptography and password protcols”, *In proceedings ACM Conference on Computer and Communications Security*, ACM press, pp. 63-72, 1999.
26. IEEE P802.11i/D10.0, “Wireless medium access control (MAC) and physical layer (PHY) specifications : medium access control (MAC) security enhancements”, April 2004.
27. D. Jablon, “Strong password-only authenticated key exchange”, *In Computer Communication Review*, Vol.26, No.5, pp. 5-26, 1996.
28. J. Katz, R. Ostrovsky, and M. Yung, “Efficient password-authenticated key exchange using human-memorable passwords”, *In proceedings of Eurocrypt'01*, LNCS Vol. 2045, pp. 475-494, Springer-Verlag, 2001.

29. A. Kashyap, H. Nishar, and P. Agarwal, "Survey on unicast routing in mobile ad hoc networks", 2001. This paper is available at http://www.cs.unibo.it/people/faculty/bononi/Sim2003/Papers/surveyrouting..pdf
30. J. Kim, S. Kim, J. Kwak, and D. Won, "Cryptanalysis and Improvements of Password Authenticated Key Exchange Scheme between Clients with Different Passwords", *In Proceedings of ICCSA 2004*, LNCS Vol. 3044, pp. 895-902, Springer-Verlag, 2004.
31. P. Kuosmanen, "Classification of ad hoc routing protocols", 2003. Available at `http://eia.udg.es/~lilianac/docs/classification-of-ad-hoc.pdf`
32. C. Lin, H. Sun, and T. Hwang, "Three-party encrypted key exchange: attacks and a solution", *In ACM Operating Systems Review*, Vol. 34, No. 4, pp. 12-20, 2000.
33. C. Lin, H. Sun, M. Steiner, and Tzonelih Hwang, "Three-party Encrypted Key Exchange Without Server Public-Keys", *In IEEE Communications Letters*, Vol. 5, No. 12, pp. 497-499, IEEE Press, 2001.
34. S. Lucks, "Open key exchange: how to defeat dictionary attacks without encryting public keys", *In proceedings of the security protocol workshop '97*, pp. 79-90, 1997.
35. M. Steiner, G. Tsudik, and M. Waider, "Refinement and extension of encrypted key exchange", *In ACM Operation Sys. Review*, Vol. 29, No. 3, pp. 22-30, 1995.
36. M. Steiner, G. Tsudik, and M, Waidner, "Diffie-hellman key distribution extended to groups", *In proceedings of ACM CCS'96*, ACM Press, 1996.
37. V. Shoup, "OAEP reconsidered", *In proceedings of Crypto01'*, LNCS Vol. 2139, pp. 239-259, 2001.
38. W. Tzeng, "A secure fault-tolerant conference-key agreement protocol", *In IEEE Transaction on Computers*, Vol. 51, No. 4, 2002.
39. V. Varadharajan and Y. Mu, "On the design of security protocols for mobile communications", *In proceedings of ACISP'96*, LNCS Vol. 1172, pp. 134-145, 1996.
40. S. Wang, J. Wang, and M. Xu, "Weakness of a Password-authenticated Key Exchange Protocol Between Clients with Different Passwords", *In Proceedings of ACNS 2004*, LNCS Vol. 3089, pp. 414-425, Springer-Verlag, 2004.
41. T. Wu, "Secure remote password protocol", *In proceedings of the Internet Society Network and Distributed System Security Symposium*, pp. 97-111, 1998.