

# Efficient Solution to Password-based Key Exchange for Large Groups

Shuhua Wu and Yuefei Zhu

Department of Networks Engineering,  
Information Science Technology Institute, Zhengzhou, China  
Email: wushuhua726@sina.com.cn

**Abstract**—This paper presents an efficient solution to key exchange problem for large groups in the password-based scenario. Our scheme can combine existing group protocols to get a hybrid group key exchange protocol which is efficient in terms of both computation and communication. And the resulting protocol allows users to securely join and leave the group at any time—the so-called dynamic case. For dynamic group communications, we propose an efficient key-updating algorithms so that the group key is only known to the members of the newly formed pool. Furthermore, our proposal does not introduce new idealizing assumptions or high entropy secrets for authentication.

**Index Terms**—password authenticated; group key establishment; provably secure

## I. INTRODUCTION

A group key exchange protocol allows a group of users to exchange information over public network to agree upon a common secret key from which a session key can be derived. This common session key can later be used to achieve desirable security goals, such as authentication, confidentiality and data integrity. Secure virtual conferences involving up to one hundred participants is an example.

Due to the usefulness of such protocols, several papers have attempted to design secure group key exchange protocols. In order to protect against an active adversary who may inject messages, impersonate one or more of the parties, or otherwise control the communication in the network, these protocols need incorporate some authentication mechanism to be authenticated ones. The most classical way to add authentication to key exchange protocols is to sign critical message flows. Unfortunately, such techniques require the use of complex infrastructures to handle public keys and certificates. One way to avoid such infrastructures is to use passwords for authentication. Humans directly benefit from this approach since they only need to remember a low-quality string chosen from a relatively small dictionary (e.g. 4 decimal digits). However, since passwords are easily-guessed strings, many password-based systems are vulnerable to replay attack or dictionary attacks [1]. To design a secure password-based system is a precise task that has attracted many cryptographers.

During the last decades, the design of 2-party password-based authenticated key establishments has been explored intensively, e.g. [2]–[5]. Nonetheless, very few group key exchange protocols have been proposed with password authentication [6]–[12]. And the situation for it is not very satisfying and there is a need for significant theoretical progress [12]. Moreover, to the best of our knowledge, none of the password-based schemes in the group scenario seems quite practical for large groups. The problems of previously-suggested protocols for password-only group key exchange can be classified as follows:

- 1) the total number of rounds significantly growing with the group size, e.g. the scheme due to Bresson et al. [6], [7], in which the total number of rounds is linear in the number of players. A scheme of this kind is obviously impractical for large groups since, as noted in [13], the efficiency of it can be severely degraded even in the case of a group where only few members have a slow network connection. Furthermore, it is clear that it is not scalable.
- 2) the most expensive part of computations for some participant significantly growing with the group size, e.g. the scheme due to Bresson et al. [7], in which one player requires to perform a large number of exponentiations and the amount is linear in a square of the number of players. This is also true for some existing password-based group key exchange protocols in a constant number of rounds, e.g. the scheme due to Abdalla et al. [8], in which each player requires to perform a large number of signature verifications and the amount is linear in the number of players.
- 3) lack of a robust way to share passwords among a large group. All of the existing schemes are subject to this problem. Intuitively, it can be assumed that all the members share a single common passwords as in [6], [8]–[11]. In that case, if there is a member corrupted, all the other members can not build secure channels any longer. Consider mission-critical applications such as emergency rescue and military operations [14]–[16], and it would be a serious problem since these applications operate in a highly mobile environment characterized by the lack of any fixed network and security infrastructure. How

This work was partially supported by a grant from the National High Technology Research and Development Program of China (863 Program) (No. 2007AA01Z471).

can a user be forced to leave when everyone in the group shares the same password? If we assume passwords are pairwise-shared among the group members, it seems to be robust enough but the number of passwords that a member would need to remember would be linear in the number of the group size, making the scheme impractical for large groups.

- 4) not allowing dynamic membership. Dynamicity in the membership may be of critical concern in practical environment. For example, it is quite important to force a corrupted member to leave and allow a newly-authorized member to join the group in the mission-critical applications mentioned above. To the best of our knowledge, none of the existing password-based authenticated group key exchange schemes enjoys dynamicity in the membership except the one in [7].

In this paper, we propose an efficient hybrid solution to key exchange problem for large groups in the password-based scenario. We deal with the problem using ideas similar to those used in the Dutta and Barua's protocol [17], which does not consider the password-based scenario. Part of our idea has appeared in [18] and this is an improved version. We assume all group users are divided into several subsets and each subset has at least a representative. Our solution consists of two stages. In the first stage, each user subset executes a pure group key exchange protocol, e.g. the scheme due to Bresson et al. [7], to establish a locally common key respectively. In the second stage, the users that represent the subsets authenticate each other and help other users to make use of keys established in the first stage to generate a new session key so that this value is known and only known to the members of the whole group. Our proposal has several advantages.

- First, our hybrid scheme can combine existing pure group protocols in the literature to get a hybrid group key agreement protocol which is efficient in terms of both computation and communication when the group size is large. In our scheme, both the computation cost and communication cost can be linear in the logarithm of the group size.
- Second, our solution eases sharing passwords among the group. In our scheme, we can deal with it hierarchically, i.e. among the users in each subset and among the representatives. Each user communicates only with the users in the same subset. And cross-subset communication is operated between the representatives. As a result, the number of passwords that a member would need to remember would be significantly reduced.
- Third, our solution allows users to securely join and leave the group at any time—the so-called dynamic case— [7].

Since executing the existing pure group protocols among all the users may be infeasible when the group size is large, our solution applies the pure group key exchange

on the subsets of small size instead. Therefore, it is quite efficient when the group size is large. Furthermore, by using our solution, a list of keys of different levels can be established so that security goals can be achieved among a varying range of user pools. This is especially attractive in some mission-critical applications such as military operations. Finally, our proposal does not introduce new idealizing assumptions or high entropy secrets for authentication. That is, if the underlying primitives do not does not impose idealizing assumptions or high entropy secrets for authentication, our scheme can yield a hybrid scheme that is password-only authenticated: users need remember only a short password and no cryptographic key(s) of any kind, and its security result does not assume the Random Oracle (RO) model [19] or the ideal cipher model.

The remainder of this paper is organized as follows. Section 2 recalls the security model for password-based key exchange. Section 3 then presents algorithmic assumptions to be used in this paper briefly. Section 4 gives a detailed description of our scheme along with its efficiency analysis and security proof. Finally, conclusion is presented in Section 5.

## II. SECURITY MODELS FOR PASSWORD-BASED KEY EXCHANGE

A secure password-based key exchange is a key exchange protocol where the parties use their passwords in order to derive a common session key  $sk$  that will be used to build secure channels. Loosely speaking, such protocols are said to be secure against *dictionary attacks* if the advantage of an attacker in distinguishing a real session key from a random key is less than  $O(q_s/|\mathcal{D}|) + \epsilon(k)$ , where  $|\mathcal{D}|$  is the size of the dictionary  $\mathcal{D}$ ,  $q_s$  is the number of active sessions and  $\epsilon(k)$  is a negligible function depending on the security parameter  $k$ .

In this section, we recall the security model we will use in the rest of the paper to define the execution of the protocol for password-based authenticated key exchange. We refer to the model newly introduced by Michel Abdalla, Pierre-Alain Fouque, and David Pointcheval(AFP) [20] as the Real-Or-Random (ROR) model and to the model introduced by Bellare, Pointcheval, and Rogaway (BPR) [21] as the Find-Then-Guess (FTG) model, following the terminology of Bellare et al. for symmetric encryption schemes [22]. As proved in [20], the Real-Or-Random (ROR) security model is actually stronger than the Find-Then-Guess (FTG) security model. In this paper, we prove our protocol is semantically secure in ROR model. This section outlines the ROR model and follows the description in [20].

### A. The Security Model

We denote by  $U_i$  a player that can participate in the key exchange protocol. The players belongs to a nonempty set  $\mathcal{U}$  of  $n$  users who can participate in the key exchange protocol  $\mathcal{P}$ . A player  $U_i$  may have several instances called oracles involved in distinct, possibly concurrent,

executions of the protocol. We denote by  $U_i^j$  the instance  $j$  of a player  $U_i$ . The players also share low-entropy secret secrets which are drawn from a small dictionary  $\mathcal{D}$ , according to the uniform distribution.

The key exchange algorithm  $\mathcal{P}$  is an interactive protocol among a group users that provides the instances of them with a session key  $sk$ . The interaction between an adversary  $\mathcal{A}$  and the protocol participants occurs only via oracle queries, which model the adversary capabilities in a real attack. The types of oracles available to the adversary are as follows:

- $Execute(\mathcal{U})$ : This query models passive attacks in which the attacker eavesdrops on honest executions of the protocol. The output of this query consists of the messages that were exchanged during the honest execution of the protocol.
- $Send(U_i^j, m)$ : This query models an active attack, in which the adversary may intercept a message and then either modify it, create a new one, or simply forward it to the intended participant. The output of this query is the message that the participant instance  $U_i^j$  would generate upon receipt of message  $m$ .

### B. Security Definitions

The aim of the adversary is to break the privacy of the session key (a.k.a., semantic security). The security notions take place in the context of executing  $\mathcal{P}$  in the presence of the adversary  $\mathcal{A}$ . One first draws passwords from  $\mathcal{D}$  according to the uniform distribution, provides coin tosses to  $\mathcal{A}$ , all oracles, and then runs the adversary by letting it ask any number of queries as described above, in any order.

**AKE Security.** In order to model the privacy (semantic security) of the session key, we consider the game  $\text{Game}_{\mathcal{P}}^{ake}$  in which an additional oracle is made available to the adversary: the  $Test(U^j)$  oracle. Let  $b$  be a bit chosen uniformly at random at the beginning of the game defining the semantic security of session keys. The  $Test$  oracle in the ROR model is defined as follows:

- $Test(U^j)$ : If no session key for instance  $U^i$  is defined, then return the undefined symbol  $\perp$ . Otherwise, return the session key for instance  $U^j$  if  $b = 1$  or a random key of the same size if  $b = 0$ . This query is only available to  $\mathcal{A}$  if the attacked instance  $U^j$  is Fresh (which roughly means that the session key is not “obviously” known to the adversary.)

As in FTG models, the  $Test$  oracle in the ROR model also tries to capture the adversary’s ability (or inability) to tell apart a real session key from a random one. The main difference is that it does so not only for a single session but for all sessions. More precisely, the adversary in the ROR model is not restricted to ask a single  $Test$  query, but it can in fact ask multiple ones. All  $Test$  queries in this case will be answered using the same value for the hidden bit  $b$  that was chosen at the beginning of the game defining the semantic security of the session keys. That

is, the keys returned by the  $Test$  oracle are either all real or all random. However, in the random case, the same random key value is returned for  $Test$  queries that are asked to be instances that belong to the same session. The goal of the adversary in the ROR model is still the same: to guess the value of the hidden bit.

Let  $\text{SUCC}$  denote the event in which the adversary is successful. The **ror-ake-advantage** of an adversary  $\mathcal{A}$  in violating the semantic security of the protocol  $\mathcal{P}$  in the ROR sense and the **advantage function** of the protocol  $\mathcal{P}$ , when passwords are drawn from a dictionary  $\mathcal{D}$ , are respectively

$$\text{Adv}_{\mathcal{P}, \mathcal{D}}^{\text{ror-ake}}(\mathcal{A}) = 2 \cdot \text{Pr}[\text{SUCC}] - 1$$

and

$$\text{Adv}_{\mathcal{P}, \mathcal{D}}^{\text{ror-ake}}(t, R) = \max_{\mathcal{A}} \{ \text{Adv}_{\mathcal{P}, \mathcal{D}}^{\text{ror-ake}}(\mathcal{A}) \},$$

where the maximum is over all  $\mathcal{A}$  with time-complexity at most  $t$  and using resources at most  $R$  (such as the number of queries to its oracles). The definition of time-complexity that we use henceforth is the usual one, which includes the maximum of all execution times in the games defining the security plus the code size [23]. Note that the advantage of an adversary that simply guesses the bit  $b$  is 0 in the above definition due to the rescaling of the probabilities.

### III. ALGORITHMIC ASSUMPTIONS

The security is proved by finding a reduction to the hardness of the Decisional Diffie-Hellman (DDH) problem over the underlying group  $\mathbb{G}$  and the security of the underlying primitives. We will briefly introduce two algorithmic assumptions in this section.

#### A. Decisional Diffie-Hellman assumption: DDH

The decisional Diffie-Hellman assumption, DDH, states, roughly, that the distributions  $(g^u, g^v, g^{uv})$  and  $(g^u, g^v, g^w)$  are computationally indistinguishable when  $u, v, w$  are drawn at random from  $\{1, \dots, |\mathbb{G}|\}$ . This can be made more precise by defining two experiments,  $\text{Exp}_{\mathbb{G}}^{\text{ddh-real}}(\mathcal{A})$  and  $\text{Exp}_{\mathbb{G}}^{\text{ddh-rand}}(\mathcal{A})$ . In both experiments, we compute two values  $U = g^u$  and  $V = g^v$  to be given to the adversary  $\mathcal{A}$ . But in addition to that, we also provide a third input, which is  $g^{uv}$  in  $\text{Exp}_{\mathbb{G}}^{\text{ddh-real}}(\mathcal{A})$  and  $g^z$  for a random  $z$  in  $\text{Exp}_{\mathbb{G}}^{\text{ddh-rand}}(\mathcal{A})$ . The goal of the adversary is to guess a bit indicating the experiment it thinks it is in. We define the advantage of  $\mathcal{A}$  in violating the DDH assumption,  $\text{Adv}_{\mathbb{G}}^{\text{ddh-rand}}(\mathcal{A})$ , as  $\text{Pr}[\text{Exp}_{\mathbb{G}}^{\text{ddh-real}}(\mathcal{A}) = 1] - \text{Pr}[\text{Exp}_{\mathbb{G}}^{\text{ddh-rand}}(\mathcal{A}) = 1]$ . The advantage function of the group,  $\text{Adv}_{\mathbb{G}}^{\text{ddh-rand}}(t)$  is then defined as the maximum value of  $\text{Adv}_{\mathbb{G}}^{\text{ddh-rand}}(\mathcal{A})$  over all  $\mathcal{A}$  with time-complexity at most  $t$ .

### B. Secure Message Authentication Scheme

A message authentication scheme is a pair of polynomial algorithms (MAC,VF). The function MAC takes a message  $m$  and a key  $k$ , and it produces a “message authentication code” (tag)  $\mu = \text{MAC}_k(m)$ . The function VF takes a message  $m$ , a tag  $\mu$  and a key  $k$ , and it returns a bit  $\text{VF}_k(m, \mu)$ , with 1 standing for accept and 0 for reject. We require that for any  $m$  output with positive probability by its tag  $\mu$ , it is the case that  $\text{VF}_k(m, \mu)$ .

For the security of the underlying MAC scheme, we consider the classical definition of existential unforgeability under chosen-message attack (CMA) due to Goldwasser, Micali, and Rivest [24]. By definition, the security level for a MAC scheme is to prevent existential forgeries, even for an adversary which has access to the tag generation and verification oracles. We define the advantage of  $\mathcal{A}$  in violating the security of the MAC scheme as  $\text{Adv}_{\text{MAC}}^{\text{euf-cma}}(\mathcal{A}) = \Pr[k \leftarrow \{0, 1\}^l, (m, \mu) \leftarrow \mathcal{A}^{\text{MAC}_k(\cdot), \text{VF}_k(\cdot, \cdot)} : \text{VF}_k(m; \mu) = 1]$ , and the advantage function of MAC,  $\text{Adv}_{\text{MAC}}^{\text{euf-cma}}(t, q_g, q_s)$  as the maximum value of the advantage  $\text{Adv}_{\text{MAC}}^{\text{euf-cma}}(\mathcal{A})$  over all  $\mathcal{A}$  that asks up to  $q_g$  and  $q_s$  queries to the tag generation and verification oracles, respectively, and with time-complexity at most  $t$ . Note that  $\mathcal{A}$  wins the above experiment only if it outputs a new valid authenticator.

## IV. OUR HYBRID PROTOCOL

In this section, we introduce our hybrid password-based group key exchange protocol. It combines a 2-party password-based key exchange, a pure password-based group key exchange and a MAC-based key exchange. Our scheme can combine existing pure group protocols to get a hybrid group key agreement protocol which is efficient in terms of both computation and communication when the group size is large.

### A. Description

We assume all group users are divided into several subsets. Each subset is identified with a unique ID and has at least a representative. If a subset has more than one representative, a spare representative can be activated when the current representative does not work. We recommend that passwords are pairwise shared among the users in each subset and among the representatives so that the password-based system can be robust enough. Our solution consists of two stages. In the first stage, each user executes a pure group key exchange among its subset. In the second stage, each user executes a MAC-based key exchange in an authenticated way under the aid of the representative of its subset to establish a new session key so that this value is known and only known to the members of the whole group. For simplicity, and without loss of generality, we assume all group users are divided into two subsets: user subset  $A$  and  $B$ . Also we assume  $U_A$  is the representative of  $A$  and  $U_B$  the representative of  $B$ . Each representative is associated with an ID to indicate

what is represented. We further assume the IDs of  $A$  and  $B$  are  $ID_A$  and  $ID_B$  respectively.

The protocol runs as follows:

- 1) Users execute a pure group key exchange  $\mathcal{P}_1$  among its subset  $A$  or  $B$  to establish a secure common agreed key  $sk_A = (x, k_1)$  and  $sk_B = (y, k_2)$  respectively, where  $x, y$  represent the higher part and  $k_1, k_2$  the rest part of  $sk_A$  and  $sk_B$  respectively. Note  $sk_A$  and  $sk_B$  are only known to the members in subset  $A$  and  $B$  respectively.
- 2) The two representatives help to execute a MAC-based key exchange between  $A$  and  $B$  in an authenticated way. At first, each player  $U$  chooses a random nonce  $N_U$  and broadcasts  $T_U = (ID, U, N_U, i)$ , where  $ID$  is used to indicate which subset  $U$  is in and  $i$  is used to indicate whether  $U$  is the representative of that subset. Then  $U_A$  and  $U_B$  execute a 2-party password-based key exchange  $\mathcal{P}_2$  to establish a secure common agreed key  $k_3$ . Afterward,  $U_A$  computes  $X = g^x$  and the MAC digest of the string  $X$  under the key  $k_3$ :  $\alpha_1 = \text{MAC}_{k_3}(TR_{AB}||X)$ , and sends  $(X, \alpha_1)$  to  $U_B$ , where  $TR_{AB}$  is the concatenation of such  $T_U$  that  $U$  is either  $U_A$  or  $U_B$ . Similarly,  $U_B$  also computes  $Y = g^y$  and  $\alpha_2 = \text{MAC}_{k_3}(TR_{AB}||Y)$  and sends  $(Y, \alpha_2)$  to  $U_A$ . Upon receiving the message from the other side, each representative verifies the the MAC digest is valid in the straight way. If  $\alpha_2$  is valid,  $U_A$  proceeds to compute the MAC digest of the string  $Y$  under the key  $k_1$ :  $\beta_1 = \text{MAC}_{k_1}(T_A||Y)$  and broadcasts  $(Y, \beta_1)$  among  $A$ , where  $T_A$  is the concatenation of all such  $T_U$  that  $U$  is a member in  $A$ . If  $\alpha_1$  is valid,  $U_B$  proceeds to compute  $\beta_2 = \text{MAC}_{k_2}(T_B||X)$  and broadcasts  $(X, \beta_2)$  among  $B$ , where  $T_B$  is the concatenation of all such  $T_U$  that  $U$  is a member in  $B$ . Upon receiving the message from the representative, each user checks that the MAC digest is valid. If it is valid, he proceeds to complete the MAC-based key exchange to establish a secret key  $g^{xy}$ . Please note,  $g^{xy}$  should be transformed as bit-strings (called session key) for subsequent use.

The solution can be easily extended to the more general case where all group users are divided into more than two subsets. Actually, we can combine two subsets as one and set a user to be their representative when the members in the two subsets have established a common agreed key. In this case, we can combine all the subsets into one by repeating the operations of the second stage after each user has executed a pure group key exchange  $\mathcal{P}_1$  among its subset. As a result, a user may be a member of several user sets of different levels at the same time. Therefore, each user has to maintain a list to record all the keys of different levels. Our proposal is essentially a solution to tree-based group key agreement in password-based setting. The procedure of combination is illustrated by an example, where all group users are divided into 8 subsets— $A, B, C, D, E, F, G, H$  with  $U_A, U_B, U_C, U_D, U_E, U_F, U_G, U_H$  as the representatives

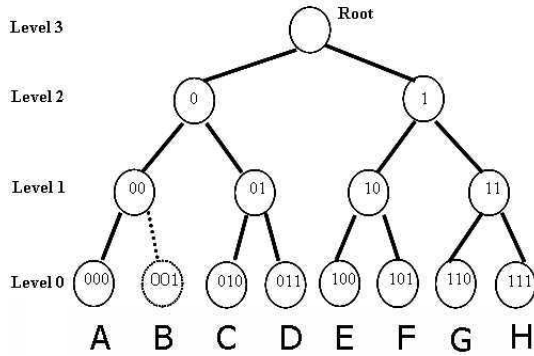


Figure 1. An example of combination of 8 subsets

respectively. All the subsets can be organized as the three in Fig 1. The procedure of combination will be performed according to the tree, where each node denotes a set that is formed by combining its two children, and the leaves are  $A, B, C, D, E, F, G, H$ . As usual, we say a pair of nodes are brothers if they root the same third node, and we call the latter as their parent( thus the two nodes are its children). For example, “000” and “001” are brothers, and their parent is “00” (thus “000” and “001” are the children of “00”). Each representative is associated with a binary string—ID, indicating which set is represented. The dotted means the subset, e.g.  $B$ (“001”), is not present at the moment. Whenever a subset updates its key locally, its representative will launch a key-updating chain as the Procedure KeyUpdate describes. At the same time, the representatives of different levels will be assigned. We should note a user can serve as several representatives of different levels, e.g.  $U_A$  is the representative of both Set “000” and Set “00” in Fig 1. Throughout the course, a request to update a key of low levels can stop updating a key of high levels. For example, a user may restart to update a key of lower levels when he is updating a key of higher levels if he receives another request.

**Procedure: KeyUpdate**

For convenience of explanation, we introduce some symbols at first. If  $\Theta$  is a set, we denote by  $U_\Theta$  the representative of  $\Theta$ ,  $K'_\Theta$  the new key of  $\Theta$ , and  $K_\Theta$  the old key of  $\Theta$ . Let  $S$  be a set,  $T$  be the brother of  $S$  and  $P$  the parent of  $S$ . We define two Boolean variables:  $N\_SecLink_{S-T}$  and  $State_T$ . We say  $N\_SecLink_{S-T}$  is true if a secure channel has not been established between  $U_S$  and  $U_T$ . And we say  $State_T$  is true if  $U_T$  is present in the network. Similarly, we can further define  $State_P$  like  $State_T$ . And  $N\_State_P$  denotes the inverse of  $State_P$ .

If  $S$  is the very set whose key has just been updated,  $U_S$  will know it because  $U_S$  is a member of  $S$ . Let  $Updt_{K_S}$  denote the event of updating the key  $K_S$ . If  $Updt_{K_S}$  occurs and  $S \neq Root$ ,  $U_S$  will do the following:

1. At first, if  $State_T$ ,  $U_S$  will do the CASE A; otherwise,  $U_S$  will do the CASE B.

1.1 CASE A:

- 1.1.1 If  $N\_SecLink_{S-T}$ ,  $U_S$  executes  $\mathcal{P}_2$  with  $U_T$  to establish a secure channel between them.
- 1.1.2  $U_S$  helps to execute the MAC-based key exchange between  $S$  and  $T$  and derive  $K'_P$  from  $Y^{x'}$  among  $S$ , where  $x'$  is the higher part of  $K'_S$ ,  $Y$  is the Diffie-Hellman public key sent by  $U_T$ .
- 1.1.3 If  $N\_State_P$ ,  $U_P$  will be determined as follows: if  $S$  is the right child of  $P$ , the user who is assigned to be  $U_S$  also serves as  $U_P$ , otherwise, the user who is assigned to be  $U_T$  will serve as  $U_P$ .

1.2. CASE B:

- 1.2.1  $U_S$  randomly chooses an element  $Y \in \mathbb{G}$  and treats as if this value is the Diffie-Hellman public key coming from the other side. And then  $U_S$  helps to derive  $K'_P$  from  $Y^{x'}$  among  $S$ . The user who is assigned to be  $U_S$  also serves as  $U_P$ .

2. Finally, in either CASE A or CASE B, both  $U_P$  and  $U_T$  will leave the remaining task to  $U_P$ . Note, when  $K_P$  is updated,  $U_P$  will proceed to perform the key-updating chain as above until it arrives the Root. Since the user who serves as  $U_P$  is the user who is assigned to be either  $U_S$  or  $U_T$  at the same time,  $U_P$  certainly knows when  $K_P$  is updated.

At first, each set of users execute  $\mathcal{P}_1$  to establish the key of the lowest level among  $A, B, C, D, E, F, G, H$  respectively. Once a fresh key is generated for a set, its representative will launch a key-updating chain so that all the keys of different levels can be established gradually. In the end, all the users in the current group will share a common fresh session key. Among  $U_A, U_B, U_C, U_D, U_E, U_F, U_G, U_H$ , the faster he computes, the more representatives he may serve as. For example, if neither  $U_C$  nor  $U_D$  has completed updating the key of Level 0 but  $U_A$  has completed updating the key of Level 1,  $U_A$  will serve as the representative of Set “0” because the representative of Set “01” has not been assigned when that of Set “00” has already been assigned to  $U_A$ . Therefore, the optimized overall efficiency can be delivered.

Our solution allows users to securely join and leave the group at any time—the so-called dynamic case— [7]. At first, we explain how a subset leaves or joins the group, which corresponds to the scenarios where a set of users can be split with the group due to network faults or malice and later the set can be properly connected to the group again. One can remark that such operations can happen as well when a representative changes. If a subset, say  $B$  in Fig 1, wants to join the group, its representative  $U_B$  just needs to launch a key-updating chain so that all the keys of different levels can be updated gradually. Namely the keys of Set “00”, “0” and the Root will be updated upward successively. In the end, all the users in the current group will share a common fresh session key. If a subset, say  $H$  in Fig 1, wants to leave the group, the tree will

break into some sets, i.e. Set “110”, “10” and “0”. Their representatives just need to launch a key-updating chain and in the end the three sets can be merged together. More specifically, after  $U_G$  helps to update the key of Level 1, he will be the representative of Set “11” and thus will also contribute to update the key of Level 2. Since the representative of Set “10” exists, the Key-updating process can lead to merging of Set “11” and “10”. Then the chain will be proceeded by the representative of Set “1” and so on. In the end, all the users in the current group will share a common fresh session key. Now, we come to explain how a user, say a user in  $B$ , leaves or joins the group, assuming that the corresponding set, i.e.  $B$ , still is maintained. If the pure group key exchange in use, i.e.,  $\mathcal{P}_1$ , allows dynamicity in the membership, the key of  $B$  can be updated efficiently. Once the key of Level 0 is updated, the representative will launch a key-updating chain so that a common fresh session key can be established among the whole group.

Finally, we note that re-running the protocol from scratch is always possible, and hence the goal of such dynamic operations is to provide an efficient means to update the existing session key into a new one. Whenever a group key can be updated as above, this value is only known to the members of the newly formed pool. We will argue it later.

*Note 1.* By using our solution, a list of keys of different levels can be established so that security goals can be achieved among a varying range of user pools. This is especially attractive in some mission-critical applications such as military operations.

### B. Efficiency

Our scheme can combine pure group protocols to get a hybrid group key agreement protocol which is efficient in terms of both computation and communication when the group size is large. Since immediate executing the existing pure group protocols among all the users may be inefficient from computation or communication point of view when the group size is large, our solution applies the pure group key exchange operation on the subsets of small size. This operation can be done by these subsets in parallel. In our case, both the computation cost and communication cost are linear in the logarithm of the group size if the size of each subset is upper bounded by a constant. Since our proposal is essentially a solution to tree-based group key agreement in password-based setting, the costs largely depend on the depth of the tree. None of the existing password-based solutions has the attractive feature. For some of them, such as [7], the total number of rounds is linear in the group size. For some of them, such as the one proposed by M. Abdalla et al. [8], the amount of the most expensive part of computation by each user is linear in the group size.

To demonstrate efficiency, we assume the 2-party password-based scheme in [3] and the group password-based scheme in [7] are used in the instantiation of  $\mathcal{P}_2$  and  $\mathcal{P}_1$  of our hybrid structure respectively. We use the

group password-based scheme in [7] to instantiate  $\mathcal{P}_1$  simply because it allows dynamicity in the membership, and accommodates pairwise-shared passwords among the members, or at least with a central server. And the other existing pure group password-based schemes in the literature do not have the two attractive features. We further assume the group is a pool of 80 users and is divided into 8 subsets uniformly. In what concerns computation cost, we only count the number of exponentiation because it is the most expensive computation in the schemes and entails the highest computational complexity. All other operations can be done efficiently and their computational complexity can be neglected. At first, we consider the case where the 80 users simply execute the group password-based scheme in [7] to establish the session key among the whole group. In that case, the number of communication rounds is  $n+2$  and some participant, which is called as the last player, requires to compute  $n^2+n-1$  exponentiations in one run, where  $n$  is the group size, i.e.  $n = 80$ . Now, we come to consider the case where our hybrid scheme is used to establish the group session key. To generate the keys of Level 0, the number of communication rounds is  $\frac{n}{8} + 2$  and the last player of each subset requires to compute  $\frac{n^2}{64} + \frac{n}{8} - 1$  exponentiations since the subset size is  $\frac{n}{8}$ . In addition, to generate the final session key, i.e. the key of Level 3, the representatives has to help to execute the MAC-based key exchange in an authenticated way at Level 1, Level 2, and Level 3 respectively. In each level, our scheme needs 5 more rounds and a representative needs to perform 4 more exponentiations, in which 2 rounds and 2 exponentiations are due to executing  $\mathcal{P}_2$ . Note, in each level, a non-representative just needs to perform one more exponentiation instead, which is due to computing the Diffie-Hellman key. Moreover, the last player of a subset may not be assigned to be a representative but we just consider the worst case in which it is assigned. For clarification, we show the efficiency analysis in Table 1. From the table, we can see that our scheme can combine a pure group protocol to get an efficient hybrid key agreement protocol for large groups.

To make the password-based system robust enough and thus allow dynamicity in the membership, the scheme in [7] assumes that passwords should be pairwise-shared among the group members. To achieve this goal, our scheme assumes that passwords are pairwise shared among the users in each subset and among the representatives. Therefore, the number of passwords that a member would need to remember would be significantly reduced in our case. To demonstrate this, we still consider the case in which the group is a pool of 80 users and is divided into 8 subsets uniformly, with a representative in each subset. In Bresson’s scheme, a member would need to remember 79 passwords. However, in our scheme, a member would need to remember 9 passwords and, if he also serves as a representative, he would need to remember 7 more passwords. In other words, a user needs to remember 16 passwords at most, which is much less than 79. Finally, we should note that, if the group size is very, very large,

**Table 1.** The efficiency analysis with  $n = 80$ 

Protocol	Computational Complexity	Communication Rounds
Bresson's protocol	$n^2 + n - 1 = 6479$	$n + 2 = 82$
Our protocol	$\frac{n^2}{64} + \frac{n}{8} - 1 + 4 * 3 = 121$	$\frac{n}{8} + 2 + 5 * 3 = 27$
Ratio: $\frac{\text{Bresson's protocol}}{\text{Our protocol}}$	53.5	3.0

we recommend that representatives are assigned to be central servers and they share high-entropy secret keys instead of passwords between them. Alternatively, we can assume passwords are pairwise shared among the representatives while a single password is shared among the users of a same subset. But that would only allow a subset of users but a individual member to join or leave the group. In the both cases, users still need to remember very few passwords but not many.

*Note 2.* Although the 2-party password-based schemes in [3] and the group password-based scheme in [7] are proved secure in the FTG mode by their authors, the security proof for them can be easily modified to meet the security goals in the ROR model with essentially the same bounds [20]. And thus the two schemes can be safely used in the instantiation of  $\mathcal{P}_2$  and  $\mathcal{P}_1$  of our hybrid structure respectively.

### C. Security

For simplicity, and without loss of generality, we assume all group users are divided into two subsets: user subset  $A$  and  $B$ . In a more general case where the group is divided into more than two subsets, we can inductively prove a group session key can also be securely established and updated so that this value is only known to the members of the current pool.

As Theorem 1 states, our hybrid password-based key authenticated protocol described in section 4.1 is secure in ROR model as long as the Decisional Diffie-Hellman assumption holds in  $\mathbb{G}$  and the underlying primitives it uses are secure.

*Theorem 1:* Let  $\mathcal{P}_2$  be a secure 2-party password-based key exchange,  $\mathcal{P}_1$  be a secure pure password-based group key exchange (in the ROR model), and  $\mathcal{MAC}$  be a secure message authentication scheme. Let  $q_{exe}$  and  $q_{test}$  represent the number of queries to *Execute* and *Test* oracles, and let  $q_s$  send, and  $q_{ake}$  represent the number of queries to the Send oracles, and the final  $\mathcal{MAC}$ -based key exchange protocol. Then,  $Adv_{\mathcal{P}, \mathcal{D}}^{ror-ake}(t, q_{exe}, q_{test}, q_s, q_{ake}) \leq 4Adv_{\mathcal{P}_1, \mathcal{D}}^{ror-ake}(t, q_{exe}, q_{exe} + q_{test}, q_s) + \frac{n^2 q_{ake}^2}{2^l} + 6q_{ake} Adv_{\mathcal{MAC}}^{euf-cma}(t, 2, 0) + 2Adv_{\mathcal{P}_2, \mathcal{D}}^{ror-ake}(t, q_{exe}, q_{exe} + q_{test}, q_s) + 2Adv_{\mathbb{G}}^{dh}(t + 6(q_{exe} + q_{ake})\tau)$ , where  $\tau$  denotes the exponentiation computational time in  $\mathbb{G}$ ,  $l$  the bit-length of a nonce and  $n$  the group size.

We can prove it using Abdalla and Pointcheval et al.'s style [20]. The complete proof is omitted here due to limitation of paper length.

Now, we come to consider key privacy with respect to the joining or leaving member, i.e. a group session key

can be updated whenever a membership changes so that this value is only known to the members of the newly formed pool.

*Theorem 2:* We still assume  $\mathcal{P}_2$  is a secure 2-party password-based key exchange,  $\mathcal{P}_1$  is a secure pure password-based group key exchange (in the ROR model), and  $\mathcal{MAC}$  is a secure message authentication scheme as above. Our scheme has key privacy with respect to the joining or leaving member as long as the Decisional Diffie-Hellman assumption holds in  $\mathbb{G}$ .

*Proof:* Let  $x$  be the higher part of the key of Level 0 for  $A$ . At first, we consider the case where a subset, say  $B$ , leaves or joins the group. If  $B$  joins the group, the new key of Level 1 among  $A$  and  $B$  is derived from  $g^{xy'}$ , where  $y'$  is the higher part of the new keys of Level 0 for  $B$ . But the old key of Level 1 among  $A$  was derived from  $g^{xy}$ , where  $y$  (or  $g^y$ ) was randomly chosen by the representative of  $A$ . One can remark that  $x$  and  $y$  are kept unknown to  $B$  all through the course. And thus the old key of Level 1 should be kept secret with respect to  $B$  as long as the DDH holds in  $\mathbb{G}$ . On the other hand, if  $B$  leaves the group, the new key of Level 1 among  $A$  is derived from  $g^{xy'}$ , where  $y'$  (or  $g^{y'}$ ) is randomly chosen by the representative of  $A$ . But the old key of Level 1 among  $A$  and  $B$  was derived from  $g^{xy}$ , where  $y$  is the higher part of the old key of Level 0 for  $B$ . One can remark that  $x$  and  $y'$  are kept unknown to  $B$  all through the course. And thus the new key of Level 1 should be secret with respect to  $B$  as long as the DDH holds in  $\mathbb{G}$ .

Now we come to consider the case where a member, say, a user  $U$  in  $B$ , leaves or joins the group but the corresponding subset  $B$  is still maintained. The new key of Level 1 among  $A$  and  $B$  is derived from  $g^{xy'}$ , where  $y'$  is the higher part of the new key of Level 0 for  $B$ . But the old key of Level 1 among  $A$  and  $B$  was derived from  $g^{xy}$ , where  $y$  is the higher part of the old key of Level 0 for  $B$ . One can remark that  $x$  and  $y$  are kept unknown to  $U$  if  $U$  joins the group. And thus the old key of Level 1 should be kept secret with respect to  $U$  as long as the DDH holds in  $\mathbb{G}$ . On the other hand, if  $U$  leaves the group,  $x$  and  $y'$  are kept unknown to  $U$ . And thus the new key of Level 1 should be kept secret with respect to  $U$  as long as the DDH holds in  $\mathbb{G}$ . ■

*Note 3.* The security analysis does not introduce new idealizing assumptions. That is, if the underlying primitives do not make use of the RO model, neither does our scheme. Hence, by using schemes whose security is in the standard model, one gets a password-based authenticated group key exchange protocol whose security is in the standard model.

## V. CONCLUSION

We have designed an efficient hybrid password-based authenticated group key exchange protocol. And the protocol also allows users to securely join and leave the group at any time—the so-called dynamic case. It is very attractive while the number of group users is large. Furthermore, our proposal does not introduce new idealizing assumptions or high entropy secrets for authentication.

## REFERENCES

- [1] S.M. Bellare, and M. Merritt. "Encrypted key exchange: password-based protocols secure against dictionary attacks," in *Proc. 1992 IEEE Computer Society Symp. on Research in security and Privacy*, pp. 72-84, May 1992.
- [2] E. Bresson, O. Chevassut, and D. Pointcheval. "Security proofs for an efficient password-based key exchange," In *Poc. ACM CCS'03*, pp. 241-250, ACM Press, Oct. 2003.
- [3] E. Bresson, O. Chevassut, and D. Pointcheval. "New security results on encrypted key exchange," In *Poc. of PKC'2004*, LNCS 2947, pp. 145-158, Springer-Verlag, Mar. 2004.
- [4] M. Abdalla and D. Pointcheval. "Simple Password-Based Encrypted Key Exchange Protocols," In *Poc. of CT-RSA'2005*, LNCS 3376, pp. 191-208, Springer-Verlag.
- [5] M. Abdalla, O. Chevassut, and D. Pointcheval. "One-time verifier-based encrypted key exchange," In *Poc. of PKC'05*, LNCS 3386, pp. 47-64. Springer-Verlag, 2005.
- [6] E. Bresson, O. Chevassut, and D. Pointcheval. "Group Diffie-Hellman key exchange secure against dictionary attacks," In *Poc. of ASIACRYPT'2002*, LNCS 2501, pp. 497-514. Springer-Verlag, Berlin, Germany, Dec. 2002.
- [7] E. Bresson, O. Chevassut, and D. Pointcheval. "A Security Solution for IEEE 802.11s Ad-hoc Mode: Password-Authentication and Group-Diffie-Hellman Key Exchange," In *International Journal of Wireless and Mobile Computing*, Volume 2, Number 1, pp. 4-13, Inderscience, 2007.
- [8] M. Abdalla and D. Pointcheval. "A Scalable Password-based Group Key Exchange Protocol in the Standard Model," In *Poc. of ASIACRYPT'2006*, LNCS 4284, pp. 332-347, Shanghai, China, December 2-6 2006. Springer-Verlag, Berlin, Germany.
- [9] M. Abdalla, E. Bresson, O. Chevassut, and D. Pointcheval. "Password-based group key exchange in a constant number of rounds," In *Poc. of PKC'2006*, LNCS 3958, pp. 427-442. Springer-Verlag, Berlin, Germany, Apr. 2006.
- [10] R. Dutta and R. Barua. "Password-based encrypted group key agreement," In *International Journal of Network Security*, 3(1):30-41, July 2006. <http://isrc.nchu.edu.tw/ijns>.
- [11] J.-M. Bohli, M. I. G. Vasco, and R. Steinwandt. "Password-authenticated constant-round group key establishment with a common reference string," Cryptology ePrint Archive, Report 2006/214, 2006. <http://eprint.iacr.org/>.
- [12] M. Abdalla, J.-M. Bohli, M. I. G. Vasco and R. Steinwandt. "(Password) Authenticated Key Establishment: From 2-Party To Group," In *Poc. of TCC'2007*, LNCS 4392, pp. 499-514, Amsterdam, Netherlands, February 21-24 2007. Springer-Verlag, Berlin, Germany.
- [13] E. Bresson and D. Catalano. "Constant Round Authenticated Group Key Agreement via Distributed Computation," In *Proc. of PKC 2004*, LNCS 2947, Springer-Verlag, pp.115-129, 2004.
- [14] K. Obraczka, G. Tsudik, and K. Viswanath. "Pushing the Limits of Multicast in Ad Hoc Networks," In *International Conference on Distributed Computing Systems*, April 2001.
- [15] C. E. Perkins. *Ad Hoc Networking*. Addison Wesley, 2001.
- [16] L. Zhou and Z. J. Haas. "Securing Ad Hoc Networks," *IEEE Network Magazine*, 13(6), 1999.
- [17] R. Dutta and R. Barua. "Dynamic Group Key Agreement in Tree-Based Setting," In *Poc. ofACISP'05*, LNCS 3574, pp. 101-112. Springer, 2005.
- [18] S.H. Wu and Y.F. Zhu. "Efficient Hybrid Password-Based Authenticated Group Key Exchange," In *Poc. of Apweb 2009*, Springer, 2009.
- [19] M. Bellare and P. Rogaway. "Random oracles are practical: A paradigm for designing efficient protocols," In *Poc. of ACM CCS 93*, pp. 62-73, Fairfax, Virginia, USA, November 3-5, 1993. ACM Press.
- [20] M. Abdalla, P.-A. Fouque, and D. Pointcheval. "Password-Based Authenticated Key Exchange in the Three-Party Setting," In *Poc. of PKC 2005*, LNCS 3386, pp. 65-84, Springer-Verlag.
- [21] M. Bellare, D. Pointcheval, and P. Rogaway. "Authenticated key exchange secure against dictionary attacks," In *Poc. of EUROCRYPT 2000*, LNCS 1807, pp. 139-155, Bruges, Belgium, May 14-18, 2000. Springer-Verlag, Berlin, Germany.
- [22] M. Bellare, A. Desai, E. Jorjani, and P. Rogaway. "A concrete security treatment of symmetric encryption," In *Poc. of 38th Annual Symposium on Foundations of Computer Science*, pp. 394-403, Miami Beach, Florida, October 19-22, 1997. IEEE Computer Society Press.
- [23] M. Abdalla, M. Bellare, and P. Rogaway. "The oracle Diffie-Hellman assumptions and an analysis of DHIES," In *Poc. of CT-RSA 2001*, LNCS 2020, pp. 143-158, San Francisco, CA, USA, April 8-12, 2001. Springer-Verlag, Berlin, Germany.
- [24] S. Goldwasser, S. Micali, and R.L. Rivest. "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," *SIAM Journal on Computing*, 17(2):281 - 308, 1988.

**Shuhua Wu** is a Ph.D. candidate at Zhengzhou Institute of Information Science Technology. His research interest is information security.

**Yuefei Zhu** is a professor of Zhengzhou Institute of Information Science Technology. His research interest is information security and cryptography.