

Converting Group Key Agreement Protocol into Password-Based Setting – Case Study

Ratna Dutta

Cryptography and Security Department, Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613.
Email: dratna@i2r.a-star.edu.sg

Abstract— Converting a secure group key agreement protocol into password-based setting is not a trivial task. The security of a password-based scheme hinges on preventing dictionary attack that enables an adversary to take advantage of the low entropy of human memorable passwords. In this paper, we make a case study considering the constant round group key agreement protocol DB proposed by Dutta-Barua, which is a simple variant of Burmester-Desmedt protocol BD. We present a simple and elegant contributory group key agreement in password-based setting derived from the protocol DB. We discuss several issues regarding the difficulties of converting this DB protocol into password-based setting. We have made several modifications in the protocol DB and then enhance it to password-based setting. The security of the proposed scheme against dictionary attacks is in the ideal cipher model under Decision Diffie-Hellman (DDH) assumption. We analyze our protocol in the security framework formalized by Bellare *et al.* Our proposed scheme achieves efficiency over the work of Bresson *et al.* in terms of both computation and communication power.

Index Terms— password-based group key agreement, DDH problem, dictionary attack

I. INTRODUCTION

Password-Based Group Key Agreement: A password-based group key agreement (GKA) protocol allows a group of users, who share only a low-entropy human-memorable password and communicating over an insecure public network, to agree upon a high-entropy session key among themselves. This session key can later be used to implement multi-cast sessions.

It is common knowledge that a user cannot remember long random numbers and he has to store a large secret, if necessary, into his system. This imposes an extra security burden and thus introduces a new point of weakness. Password-based setting seems to be a more realistic scenario and convenient to use as they do not require the use of more specialized hardware for storing or generating secret keys. The users simply remember a relatively low-entropy easily memorable password, say a 4 digit PIN number or a 6 character string. Also password-based GKA enables to establish a session key in the absence of public key infrastructures or pre-distributed symmetric keys and greatly simplifies the setup of distributed applications,

for instance, in a setup where the networking of all the devices are attached to a human.

There are several mission-critical applications of password-based group key agreement protocols in military and emergency rescue operations. Also these protocols are suitable for commercial applications like conferencing, meeting and personal networking.

On the other hand, password-based protocols are always subject to password-guessing attacks due to the low entropy of the passwords, where the attacker recovers the password and uses it to impersonate the legitimate user. In case of a failure, he can try with a different password guess and erase the previous guess. In *on-line guessing attack*, the adversary needs to be present and interact with the system during his attempt to impersonate a user. Such attacks cannot be completely ruled out for a real world adversary, but the damage caused by such attacks can be mitigated. The goal of password-based protocols is to restrict the adversary's capability to the on-line case by limiting the the number of failed login attempts or imposing a minimum time interval between failed attempts.

A more serious threat against password-based protocols is *off-line exhaustive search attack*, also known as *dictionary attack*. Since the passwords are drawn from a relatively small and easily generated dictionary, the attacker may run through all the possible passwords in order to obtain partial information and to maximize his success probability in the off-line case. By off-line exhaustive search, the attacker may significantly reduce the number of password candidates. Low entropy feature of passwords makes the off-line exhaustive search attack very damaging as the attacker may get non-negligible chance of winning. A secure password-based protocol should resist at least this attack.

Our Contribution: The design of key agreement in password-based setting has been devoted much less attention in *group* setting and there is certainly a need for significant theoretical progress. It seems to be a non-trivial task to convert any GKA to password-based setting and establish strong provable security guarantees without making idealized assumptions.

In this paper, we address the problem of converting a provably authenticated group key agreement into a password-based group key agreement. We make a

This paper is based on "Multi-Party Key Agreement in Password-Based Setting," by Ratna Dutta, which appeared in the Proceedings of First Asia International Conference on Modelling and Simulation (AMS'07), March 2007. © 2007 IEEE.

case study considering the GKA protocol proposed by Dutta-Barua in [15]. They presented a constant round group key agreement protocol (DB), similar to Burmester-Desmedt [14], [22] group key agreement (BD). We enhance the DB protocol into a password based GKA. We discuss several issues regarding the difficulties of converting this DB protocol in password-based setting. The straight-forward conversion of DB protocol by replacing the signature scheme by a symmetric encryption scheme using the password as secret key does not enable the protocol to resist off-line dictionary attacks due to presence of several redundancies in the communication. We analyze this naive approach in a length. To remove these redundancy, we make several modifications in the unauthenticated (version of DB) protocol presented in [15] and then apply encryption-based authentication mechanism using the password as a part of secret key to transform it into a secure password-based protocol that withstand dictionary attack. This yields our password-based scheme with security against dictionary attack in the ideal cipher model under DDH assumption.

On a more positive note, proposed protocol is efficient in terms of both computation and communication power as compared to the protocol of Bresson *et al.* [13]. The number of rounds required for our protocol is only 2. Each group member in the protocol performs at most 3 modular exponentiations, $2n$ multiplications and/or divisions, 2 encryptions and $n+1$ decryptions. Our proposed scheme being of constant round, the cost of computation does not linearly grow with the increase in the number of participants. A conference version of this work has already been published in preliminary form in [17]. The current manuscript is an extended version.

II. SURVEY ON PREVIOUS WORK

The area of password-based authenticated key exchange protocols has been the subject of a vast amount of work in the last few years due to its practical aspects. While the majority of the work deals with different aspects of 2-party key exchange [6], [9], [11], [18]–[20], only a few take into account the group key establishment with $n > 2$ participants, and there is a need for significant theoretical progress.

In the 2-party case, perhaps the most well known Diffie-Hellman variant is the family of Encrypted Key Exchange (EKE) protocols by Bellare and Merritt [7] that deals with off-line dictionary attacks for the first time. This work is very influential and became the basis for much future work in the area. Following it, a number of protocols for password-based key exchange have been proposed [20], [21], security of which are based on heuristic arguments. There are two proposals of formal models for password-based key exchange. One by Bellare, Pointcheval and Rogaway [6] which is based on [4], [5] and a second by Boyko, MacKenzie and Patel [9] following [26]. Bellare *et al.* proposed a 2-party password-based protocol in [6]. Bresson *et al.* [12] examined the security of 2-party AuthA password-authenticated key exchange

protocol standardized by IEEE P1363 Study Group. The security of all these schemes are both in the random oracle and the ideal cipher model. Using the multi-party simulatability technique, Boyko *et al.* [9] and MacKenzie *et al.* [25] presented protocols which are secure in the random oracle model. Goldrich and Lindell [19] provided a protocol based on general assumptions and prove its security in the standard security model of [9]. Independent of this work, Katz, Ostrovsky and Yung [23] suggested a highly efficient password-authenticated key exchange protocol in the common reference string model. Gennaro and Lindell [18] further extended this work using more general smooth projective hash function primitives.

An extension of the 2-party case in [6] to multi-party setting is presented by Bresson *et al.* [13]. In [10], they proposed another password-based group key exchange protocol. These constructions are secure in the ideal cipher model and the total number of rounds is linear in the number of players, making the schemes impractical for large groups. Recently, several constant round password-based group key exchange protocols have been proposed in the literature by [2], [16], [24] which are quite efficient and are based on the Burmester-Desmedt [14], [22] group key agreement protocol, but their security analysis usually require the random oracle and/or ideal cipher models. Recently, two new constant round password-based group key exchange protocols have been proposed independently and concurrently by Abdalla *et al.* [3] and Bohli *et al.* [8] with proof of security in the standard model. More recently, Abdalla *et al.* [1] described a compiler that transforms any provably secure 2-party key establishment into a provably secure authenticated group key establishment with 2 or more rounds of communication in the common reference string model.

III. SECURITY MODEL

Now we briefly describe the formal security model of Bellare *et al.* [6] as standardized by Bresson *et al.* [12], [13] and refer the reader to [6], [12], [13] for more details.

A protocol P for password-based group key agreement assumes that there is a set $\mathcal{P} = \{U_1, U_2, \dots, U_n\}$ of n users (n is fixed), who share a low entropy secret password pw drawn uniformly from a small dictionary of size N . The adversary is given control over all communication in the external network. We assume that users do not deviate from the protocol and adversary never participates as a user in the protocol. This adversarial model allows concurrent execution of the protocol among n users. The interaction between the adversary \mathcal{A} and the protocol participants occur only via oracle queries, which model the adversary's capabilities in a real attack. These queries are as follows (Π_U^i denotes the i -th instance of user U and sk_U^i denotes the session key after execution of the protocol by Π_U^i):

- $\text{Send}(U, i, m)$: The adversary can carry out an active attack by this query. The output of the query is the reply (if any) generated by the instance Π_U^i upon receipt of message m . The adversary is allowed to prompt the

unused instance Π_U^i to initiate the protocol by invoking $\text{Send}(U, i, \text{"Start"})$.

– $\text{Execute}(\{(U_1, i_1), \dots, (U_n, i_n)\})$: This query reflects the adversary's ability to passively eavesdrop on honest execution of password-based group key agreement protocol among unused instances $\Pi_{U_1}^{i_1}, \dots, \Pi_{U_n}^{i_n}$ and outputs the transcript of the execution. A transcript consists of the messages that were exchanged during the honest execution of the protocol.

– $\text{Reveal}(U, i)$: This unconditionally outputs session key sk_U^i that has previously been accepted by instance Π_U^i . If the session key is not accepted, a value NULL is returned.

– $\text{Corrupt}(U)$: \mathcal{A} gets back from this query pw, but does not get any internal data on U .

– $\text{Test}(U, i)$: A bit $b \in \{0, 1\}$ is chosen uniformly at random. The adversary is given sk_U^i if $b = 1$, and a random session key if $b = 0$.

We define the session identity sid_U^i for instance Π_U^i to be $\text{sid}_U^i := S = \{(U_1, i_1), \dots, (U_n, i_n)\}$ such that $(U, i) \in S$ and $\Pi_{U_1}^{i_1}, \dots, \Pi_{U_n}^{i_n}$ are involved in the session to agree upon a common key. The session identity uniquely identifies a session. Note that all the instances involved in a session have same session identity. We also assume that an instance of a user participates in at most one session. This means that the session identities of an instance for different sessions are mutually disjoint. We also define a boolean function acc_U^i which is set to be 1 by Π_U^i upon normal termination of the session and 0 otherwise.

Definition 3.1: The correctness of the protocol P means that if the instances $\Pi_{U_1}^{i_1}, \dots, \Pi_{U_n}^{i_n}$ having same session identity accept, then the instances must conclude with the same session key. In other words, if $\text{sid}_{U_j}^{i_j} = \text{sid}_{U_k}^{i_k}$ for $1 \leq j, k \leq n$ such that $\text{acc}_{U_j}^{i_j} = 1$ for $1 \leq j \leq n$, then $\text{sk}_{U_j}^{i_j} = \text{sk}_{U_k}^{i_k}$ for $1 \leq j, k \leq n$.

The Send, Execute, Reveal and Corrupt queries are asked several times, but Test query is asked only once and on a fresh instance. We say that an instance Π_U^i is fresh if

(a) no $\text{Corrupt}(V)$ query ($V \in \text{pid}_{U'}^i$) before a query of the form $\text{Send}(U, i, *)$ or $\text{Send}(U', j, *)$ where $U' \in \text{pid}_{U'}^i$, and

(b) Π_U^i has computed a session key and neither Π_U^i nor its partners have been asked a Reveal query (i.e. the adversary queried neither $\text{Reveal}(U, i)$ nor $\text{Reveal}(U', j)$ with $(U', j) \in \text{sid}_U^i, U' \in \mathcal{P}$).

Finally, adversary outputs a guess bit b' . Such an adversary is said to win the game if $b = b'$ where b is the hidden bit used by the Test oracle. Let Succ denote the event that the adversary \mathcal{A} wins the game for a protocol P . We define $\text{Adv}_{\mathcal{A}, P} := |2 \text{Prob}[\text{Succ}] - 1|$ to be the advantage of the adversary \mathcal{A} in attacking the protocol P where the probability space is over all the random coins of the adversary and all the oracles. We denote by $\text{Adv}_P^{\text{AKA}}(t, q_E, q_S)$ the maximum advantage of any adversary attacking protocol P , running in time t and making q_E calls to the Execute oracle and q_S calls to the

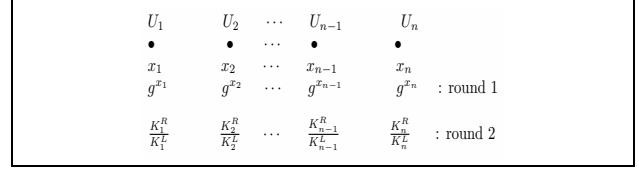


Figure 1. An execution of the protocol DB among n users; $K_i^L = g^{x_{i-1}x_i}$, $K_i^R = g^{x_i x_{i+1}}$; U_i transmits g^{x_i} in round 1 and $\frac{K_i^R}{K_i^L}$ in round 2; the session key is $\text{sk} = K_1^R K_2^R \dots K_n^R = g^{x_1 x_2 + x_2 x_3 + \dots + x_n x_1}$

Send oracle.

We adopt the following definition given by Gennaro-Lindell [18] for security of password-based key agreement protocols against dictionary attack:

Definition 3.2: A password based group key agreement protocol P is secure against dictionary attack if for every dictionary \mathcal{D} ,

$$\text{Adv}_P^{\text{AKA}}(t, q_E, q_S) \leq \frac{q_S}{|\mathcal{D}|} + \epsilon(l)$$

where ϵ is a negligible function in the security parameter l of the system.

IV. PROTOCOL

Suppose a set of $n \geq 3$ users $\mathcal{P} = \{U_1, U_2, \dots, U_n\}$ share a low entropy secret password pw drawn uniformly from a small dictionary of size N and wish to establish a high entropy common session key among themselves. We consider the users U_1, \dots, U_n participating in the protocol are on a ring and U_{i-1}, U_{i+1} are respectively the left and right neighbors of U_i for $1 \leq i \leq n$ with $U_0 = U_n, U_{n+1} = U_1$ and U_{n+i} is taken to be U_i . Quite often we identify a user U_i with his instance $\Pi_{U_i}^{d_i}$ (for some unique integer d_i that is session specific) during a protocol execution. We denote by $A|B$ the concatenation of A, B .

Let $G = \langle g \rangle$ be a multiplicative group of some large prime order q and $\overline{G} = G \setminus \{1\}$. Then $\overline{G} = \{g^x | x \in \mathbb{Z}_q^*\}$. The password pw shared among the members of the group is used as a part of key for encryption/decryption. We use a secure block cipher $(\mathcal{E}_k, \mathcal{D}_k)$ where \mathcal{E}_k is a keyed permutation over a set \mathcal{S} , \mathcal{D}_k is the inverse of \mathcal{E}_k and k is the key for encryption/decryption drawn from a key space of size N . The set \mathcal{S} is of the form $\mathcal{S} = \{U|d|t|X : \text{User } U \text{ with instance number } d \text{ sends } t^{\text{th}} \text{ message } X \in \overline{G}\}$.

A. DB : Unauthenticated GKA of [15]

The unauthenticated protocol DB presented in [15] involves two rounds and a key computation phase. The protocol is executed as follows among n users U_1, \dots, U_n .

1. (Round 1) Each user $U_i = \Pi_{U_i}^{d_i}$ chooses randomly a private key $x_i \in \mathbb{Z}_q^*$, computes $X_i = g^{x_i}$ and sends X_i to his neighbors U_{i-1}, U_{i+1} .

2. (Round 2) User U_i on receiving X_{i-1} from U_{i-1} and X_{i+1} from U_{i+1} computes his left key $K_i^L = X_{i-1}^{x_i}$,

right key $K_i^R = X_{i+1}^{x_i}$, $Y_i = K_i^R/K_i^L$ and sends Y_i to the rest of the users.

3. (Key Computation) Finally in the key computation phase, U_i computes $\overline{K}_{i+1}^R, \overline{K}_{i+2}^R, \dots, \overline{K}_{i+(n-1)}^R$ as follows making use of his own right key K_i^R : $\overline{K}_{i+1}^R = Y_{i+1}K_i^R$, $\overline{K}_{i+2}^R = Y_{i+2}\overline{K}_{i+1}^R, \dots, \overline{K}_{i+(n-1)}^R = Y_{i+(n-1)}\overline{K}_{i+(n-2)}^R$.

Then U_i verifies if $\overline{K}_{i+(n-1)}^R$ is same as that of his left key $K_i^L (= K_{i+(n-1)}^R)$. If verification fails, then U_i aborts. Otherwise, U_i has correct right keys of all the users. U_i computes the session key $sk_{U_i}^{d_i} = \overline{K}_1^R \overline{K}_2^R \dots \overline{K}_n^R$ which is equal to $g^{x_1x_2+x_2x_3+\dots+x_nx_1}$. Figure 1 illustrates the protocol.

Complexity : Each user computes 3 exponentiations (one in round 1 and two in round 2) and at most $2n - 2$ multiplications ($n - 1$ multiplications for recovery of all right keys and $n - 1$ multiplications for session key computation).

B. KeyAgree : Modified Version of Protocol DB

The unauthenticated protocol KeyAgree described in this section is the basis of our password-based scheme (presented in the subsequent subsection) and is obtained by introducing certain modifications to the protocol DB.

Like DB protocol, this protocol also involves two rounds and a key computation phase. An honest execution of the protocol proceeds as follows among user instances $\Pi_{U_1}^{d_1}, \dots, \Pi_{U_n}^{d_n}$. Figure 2 illustrates the protocol.

Let $str_i^{(l)} = U_i|d_i|l, 1 \leq i \leq n, l = 1, 2$.

1. (Round 1) At the start of the session, each user $U_i = \Pi_{U_i}^{d_i}$ chooses randomly a private key $x_i \in Z_q^*$, computes $X_i = g^{x_i}$ and sends $T_i = str_i^{(1)}|X_i$ to his neighbors U_{i-1}, U_{i+1} . In addition, user U_n sends $T_n = str_n^{(1)}|X_n$ to other participants involved in the session.

2. (Round 2) After this communication round is over, U_i receives $T_{i-1} = str_{i-1}^{(1)}|X_{i-1}$ from U_{i-1} , $T_{i+1} = str_{i+1}^{(1)}|X_{i+1}$ from U_{i+1} and $T_n = str_n^{(1)}|X_n$ from U_n . User U_i aborts the protocol if any two of X_{i-1}, X_i and X_{i+1} are same which take place with negligible probability. Else U_i extracts d_n and computes his left key $K_i^L = X_{i-1}^{x_i}$ and right key $K_i^R = X_{i+1}^{x_i}$. Each user U_i for $1 \leq i \leq n - 1$ computes $\overline{X}_i = K_i^R/K_i^L$ and sends $\overline{T}_i = str_i^{(2)}|\overline{X}_i$ to the rest of the users while user U_n remains passive (*i.e.* sends nothing) in the second round. We note that the right key of U_i is same as the left key of U_{i+1} .

3. (Key Computation) Finally in the key computation phase, U_i computes $K_1^R, K_2^R, \dots, K_n^R$ as follows making use of his own left key K_i^L and right key K_i^R :

$$K_{i+j}^R = \overline{X}_{i+j}K_{i+j-1}^R \text{ for } 1 \leq j \leq n - i - 1$$

$$K_{i-j}^L = K_{i-j+1}^L/\overline{X}_{i-j} \text{ for } 1 \leq j \leq i - 1$$

Point to note that $K_{i-j}^L = K_{i-j-1}^R$ and $K_1^L = K_n^R$. Thus U_i recovers all the right keys K_n^R, K_j^R for $1 \leq j \leq n - 1, j \neq i - 1, i$ and U_i himself knows $K_{i-1}^R (= K_i^L)$

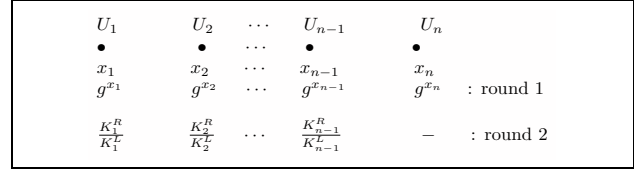


Figure 2. An execution of the protocol KeyAgree among n users, $K_i^L = g^{x_{i-1}x_i}$, $K_i^R = g^{x_i x_{i+1}}$; U_i transmits $str_i^{(1)}|g^{x_i}$ in round 1 and $str_i^{(2)}|\frac{K_i^R}{K_i^L}$ in round 2; U_i aborts if any two of $g^{x_{i-1}}, g^{x_i}, g^{x_{i+1}}$ are same; U_n sends nothing in round 2; the session key is $sk = K_1^R K_2^R \dots K_n^R = g^{x_1x_2+x_2x_3+\dots+x_nx_1}$

and K_i^R . Then U_i computes the session key $sk_{U_i}^{d_i} = K_1^R K_2^R \dots K_n^R$ which is equal to $g^{x_1x_2+x_2x_3+\dots+x_nx_1}$. User U_i also extracts instance numbers d_1, \dots, d_n from the received messages and sets his session identity $sid_{U_i}^{d_i} = \{(U_1, d_1), \dots, (U_n, d_n)\}$.

Complexity : Each user computes 3 exponentiations (1 in round 1 and 2 in round 2) and at most $2n$ multiplications and/or divisions (including n multiplications for session key computation).

We now explore certain redundancies in the transmitted messages present in the protocol DB [15] which we removed in the protocol KeyAgree by incorporating certain check steps and thereby aborting the protocol if such a situation occurs. Presence of any such redundancy in KeyAgree will cause the password-based scheme (described in the subsequent section) vulnerable to off-line dictionary attacks.

(a) Unlike DB protocol, each user U_i in KeyAgree, after first round communication, checks if any two of X_{i-1}, X_i and X_{i+1} are same and if so aborts the protocol.

Case (i): $X_{i-1} = X_{i+1}$: This implies $K_i^L = K_i^R$ which in turn yields a redundancy $\overline{X}_i = 1$.

Case (ii): $X_{i-1} = X_i$: This implicitly defines $x_{i-1} = x_i$. Then $\overline{X}_{i-1} = \frac{K_{i-1}^R}{K_{i-1}^L} = \frac{g^{x_{i-1}x_i}}{g^{x_{i-2}x_{i-1}}}$ and $\overline{X}_i = \frac{K_i^R}{K_i^L} = \frac{g^{x_i x_{i+1}}}{g^{x_i x_{i-1}}}$. Now if in addition, it happens to be the case that $X_{i-2} = X_{i+1}$ (*i.e.* $x_{i-2} = x_{i+1}$, probability of which is very small for two honest users U_{i-2} and U_{i+1}), then $\overline{X}_{i-1}\overline{X}_i = 1$, another possible redundancy in the transmitted messages of the protocol DB. An active adversary may interleave the messages during the protocol execution and manipulate the transmitted messages to create such a redundancy as follows: adversary simply replaces X_{i+1} by a copy of X_{i-2} (that he obtains by interleaving the protocol transmission) in the first round communication.

Case (iii): $X_i = X_{i+1}$: Arguing in a similar way as in case (ii), if $X_{i-1} = X_{i+2}$ (*i.e.* $x_{i-1} = x_{i+2}$, probability of which is negligible), then one obtains the redundancy relation $\overline{X}_i\overline{X}_{i+1} = 1$. This situation can be created by an active adversary by replacing X_{i+2} by a copy of X_{i-1} at hand in the first round communication.

(b) In the second round of KeyAgree, the n -th user remains passive, *i.e.* does not participate in the communication, whereas in DB protocol, all users $U_i, 1 \leq i \leq n$, sends \overline{X}_i to the rest of the participants yielding a relation

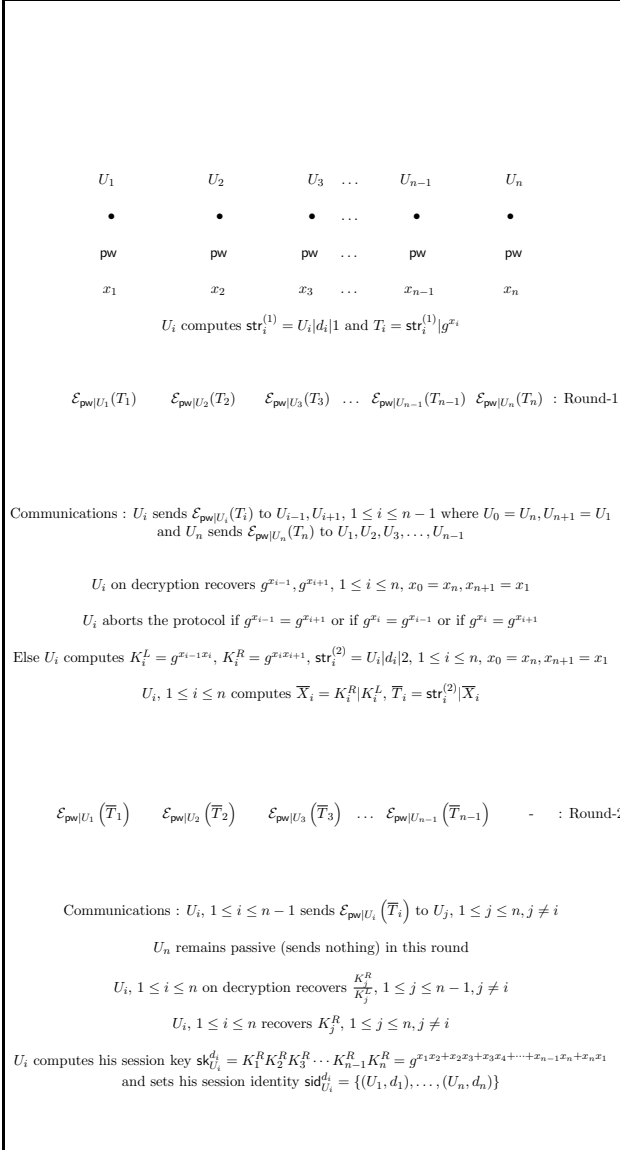


Figure 3. An execution of the protocol P – KeyAgree among n users, $K_i^L = g^{x_{i-1}x_i}, K_i^R = g^{x_i x_{i+1}}; U_i$ transmits encryption of $\text{str}_i^{(1)} |g^{x_i}$ in round 1 and encryption of $\text{str}_i^{(2)} | \frac{K_i^R}{K_i^L}$ in round 2 under the encryption key $\text{pw}|U_i; U_i$ aborts if any two of $g^{x_{i-1}}, g^{x_i}; g^{x_{i+1}}$ are same; U_n sends nothing in round 2; the session key is $\text{sk} = K_1^R K_2^R \dots K_n^R = g^{x_1 x_2 + x_2 x_3 + \dots + x_n x_1}$

$\bar{X}_1 \bar{X}_2 \dots \bar{X}_n = 1$. We are essentially removing this redundancy from the transmitted messages in the protocol KeyAgree by restricting U_n from communicating in the second round.

An active adversary, manipulating the messages exchanged during protocol execution, may be able to create other kind of redundancies. So while transforming the unauthenticated protocol DB into password-based setting, we should pay careful efforts to make an active adversary unable to get any advantage in guessing the password offline. This is a non-trivial task.

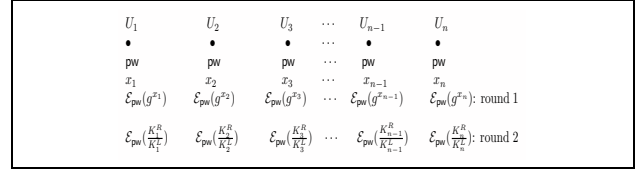


Figure 4. Naive password-based approach: a straight-forward conversion of the protocol DB to password-based setting among n users; $K_i^L = g^{x_{i-1}x_i}, K_i^R = g^{x_i x_{i+1}}; U_i$ transmits g^{x_i} in round 1 and $\frac{K_i^R}{K_i^L}$ in round 2; the session key is $\text{sk} = K_1^R K_2^R \dots K_n^R = g^{x_1 x_2 + x_2 x_3 + \dots + x_n x_1}$

C. P – KeyAgree : Password-Based GKA

We now present our password-based group key agreement protocol P – KeyAgree that is obtained by modifying the protocol KeyAgree introducing encryption-based authentication mechanism with the password pw as a part of the secret key. The protocol proceeds as follows among n instances $\Pi_{U_1}^d, \dots, \Pi_{U_n}^d$. An execution of the protocol among n users is shown in Figure 3.

Let $\text{str}_i^{(l)} = U_i |d_i|l$ for $1 \leq i \leq n, l = 1, 2$.

1. (Round 1) In the first round, each user U_i chooses randomly a private key $x_i \in_R \mathcal{Z}_q^*$, computes $X_i = g^{x_i}$, encrypts $T_i = \text{str}_i^{(1)} |X_i$ using $\text{pw}|U_i$ as secret key to obtain $Y_i = \mathcal{E}_{\text{pw}|U_i}(T_i)$ and sends Y_i to his neighbors U_{i-1}, U_{i+1} . In addition, user U_n sends Y_n to other participants involved in the session.

2. (Round 2) In the second round, each user U_i on receiving Y_{i-1}, Y_{i+1} from his neighbors, decrypts Y_{i-1}, Y_{i+1} with the respective decryption keys $\text{pw}|U_{i-1}, \text{pw}|U_{i+1}$ and obtains $T_{i-1} = \text{str}_{i-1}^{(1)} |X_{i-1}, T_{i+1} = \text{str}_{i+1}^{(1)} |X_{i+1}$ respectively. U_i aborts the protocol if any two of X_{i-1}, X_i and X_{i+1} are same which occurs with negligible probability. Else U_i extracts d_n and computes his left key $K_i^L = X_{i-1}^{x_i}$ and right key $K_i^R = X_{i+1}^{x_i}$. Each user U_i for $1 \leq i \leq n-1$ computes $\bar{X}_i = K_i^R / K_i^L, \bar{T}_i = \text{str}_i^{(2)} | \bar{X}_i, \bar{Y}_i = \mathcal{E}_{\text{pw}|U_i}(\bar{T}_i)$ and sends \bar{Y}_i to the rest of the users in the second round. U_n remains passive (i.e. sends nothing) in this round. We note that right key of U_i is same as the left key of U_{i+1} .

3. (Key Computation) Each user U_i on receiving \bar{Y}_j from $U_j, 1 \leq j \leq n-1$, decrypts \bar{Y}_j using $\text{pw}|U_j$ as the decryption key to recover \bar{X}_j and extracts instance number d_j . User U_i then computes $K_1^R, K_2^R, \dots, K_n^R$ exactly as in the protocol KeyAgree (cf. section IV-B) making use of his own left key K_i^L and right key K_i^R .

V. REDUNDANCIES AND SECURITY ISSUES

The procedure P – KeyAgree is obtained by modifying the unauthenticated protocol KeyAgree by introducing encryption-based authentication mechanism. Note that each user U_i uses $\text{pw}|U_i$ as encryption key. Only those users who have the knowledge of pw, would be able to decrypt the encrypted messages.

If instead, we apply encryption-based mechanism with pw as the encryption key for user U_i directly to the protocol DB of [15] (see Figures 3, 4), then this does

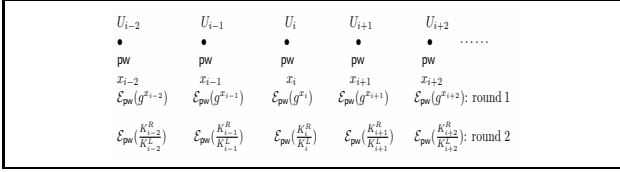


Figure 5. Dictionary attack to the password-based protocol in Figure 4; U_i can make use of the following redundancies for off-line dictionary attack where the transmitted messages over the public network are encryptions of $X_i = g^{x_i}$ and $\bar{X}_i = \frac{K_i^R}{K_i^L} = \frac{g^{x_i x_{i+1} + 1}}{g^{x_i x_{i-1} - 1}}$ under the password pw: (a) $\bar{X}_1 \bar{X}_2 \dots \bar{X}_n = 1$, (b) if $X_{i-1} = X_{i+1}$, then $\bar{X}_i = 1$, (c) if $X_{i-1} = X_i$ and the adversary makes $X_{i-2} = X_{i+1}$ to U_i , then $\bar{X}_{i-1} \bar{X}_i = 1$, (d) if $X_{i+1} = X_i$ and the adversary makes $X_{i+2} = X_{i-1}$ to U_i , then $\bar{X}_{i+1} \bar{X}_i = 1$

not yield a secure password-based protocol and one can mount an off-line dictionary attack as follows: Observe that $\bar{X}_1 \bar{X}_2 \dots \bar{X}_n = 1$. This redundancy enables an adversary to make the protocol vulnerable to dictionary attacks by guessing the password off-line and verifying whether the product of decrypted values in the second round communication is 1. If so, the adversary's guess for password is correct. To prevent such attacks, we remove the redundancy in protocol KeyAgree by keeping U_n inactive in this round. As a result, the key computation is appropriately modified. These are the points where the procedure KeyAgree differs from the unauthenticated scheme DB of [15].

Note that the protocol KeyAgree is aborted if $X_{i-1} = X_{i+1}$ which in turn implies $K_i^L = K_i^R$. This step is essential to disable an adversary from mounting off-line password guessing attack, because if the protocol proceeds with $K_i^L = K_i^R$ for some i , then $\bar{X}_i = 1$ (see Figure 4) and the corresponding publicly transmitted value is simply the encryption of a constant string. By maintaining a list, the attacker can exhaustively search for the password.

Also the protocol KeyAgree is aborted if $X_i = X_{i-1}$ or $X_i = X_{i+1}$. As discussed earlier in Section IV-B, an active adversary may manipulate the transmitted messages during protocol execution by Send query so that $\bar{X}_{i-1} \bar{X}_i = 1$ (adversary simply replaces in the first round the ciphertext of U_{i+1} by the ciphertext of U_{i-2} in the same round) or $\bar{X}_i \bar{X}_{i+1} = 1$ (adversary replaces the ciphertext of U_{i+2} in the first round by the ciphertext of U_{i-1} in the same round). Now if the encryption-based mechanism uses pw as the encryption key instead of $\text{pw}|U_i$ for user U_i (see Figure 4), then an active adversary in the on-line phase may manipulate the messages as described above, search exhaustively for the password by checking $\bar{X}_{i-1} \bar{X}_i = 1$ or $\bar{X}_i \bar{X}_{i+1} = 1$ and thus mounts an off-line dictionary attack. To resist these type of manipulation-based redundancies, the encryption by user U_i is done using $\text{pw}|U_i$ as the secret key instead of pw.

VI. SECURITY ANALYSIS

Theorem 6.1: [15] *The unauthenticated protocol UP described in subsection IV-B is secure against passive ad-*

versary under DDH assumption, achieves forward secrecy and satisfies the following:

$$\text{Adv}_{\text{UP}}^{\text{KA}}(t, q_E) \leq 4 \text{Adv}_G^{\text{DDH}}(t') + \frac{8q_E}{q}$$

where $\text{Adv}_{\text{UP}}^{\text{KA}}(t, q_E)$ is the maximum advantage of any passive adversary attacking protocol UP, running in time t and making at most q_E calls to the Execute oracle, $t' = t + O(|\mathcal{P}| q_E t_{\text{exp}})$ and t_{exp} is the time required to perform an exponentiation in G .

Theorem 6.2: *The password-based encrypted key agreement protocol PAP described in subsection IV-C satisfies the following:*

$$\text{Adv}_{\text{PAP}}^{\text{AKA}}(t, q_E, q_S) \leq \text{Adv}_{\text{UP}}^{\text{KA}}(t', q_E + \frac{q_S}{n}) + \frac{q_E^2}{2|\mathcal{S}|} + \frac{q_S}{N}$$

with $t' \leq t + (|\mathcal{P}| q_E + q_S) t_{\text{PAP}}$ where t_{PAP} is the time bound of the execution of PAP by any one of the n users, N is the size of the dictionary of all possible passwords and q_E, q_S are respectively the maximum number of encryption/decryption, Execute and Send queries an adversary may ask and $\mathcal{S} = \{U|d|t|X : U \text{ with instance number } d \text{ sends } t^{\text{th}} \text{ message } X \in \bar{G}\}$.

Proof : Let \mathcal{A}' be an adversary which attacks the password-based protocol PAP. Using this we construct an adversary \mathcal{A} which attacks the unauthenticated protocol UP. Adversary \mathcal{A} uses a list tlist . It stores pairs of session IDs and transcripts in tlist . Adversary \mathcal{A} draws a password pw from the set of all possible passwords to initialize the real attack and gives all the instances of users to \mathcal{A}' in order to cover concurrent execution of the protocol PAP.

Suppose Encrypt is the event that \mathcal{A}' makes a Send query on a message that is encrypted by \mathcal{A}' itself by guessing the password. \mathcal{A} simulates the encryption and decryption queries of \mathcal{A}' by maintaining two lists elist and dlist (initially empty) as follows, each of size at most q_E and all the other oracle queries of \mathcal{A}' are answered by \mathcal{A} using its own queries to the Execute oracle.

-Encryption query: For an encryption query $\mathcal{E}_{k|U}(X)$ such that a record $(\delta, k|U, X, \Lambda, Y)$ appears in elist (δ is either 0 or 1 and Λ is either \mathcal{E} or \mathcal{D}), the answer is Y . Otherwise, a random ciphertext $Y \in \mathcal{S}$ of length $|X|$ is chosen and the record $(\delta, k|U, X, \mathcal{E}, Y)$ is added to elist where $\delta \in \{0, 1\}$. If the query comes from the simulator, δ is set to be 0; else the query is directly asked by the adversary and δ is set to be 1.

-Decryption query: For a decryption query $\mathcal{D}_{k|U}(Y)$ such that a record $(\delta, k|U, X, \Lambda, Y)$ appears in elist (δ is either 0 or 1 and Λ is either \mathcal{E} or \mathcal{D}), the answer is X . Otherwise, $X \in \mathcal{S}$ of length $|Y|$ is chosen randomly. The record $(k|U, Y, \mathcal{D}, X)$ is added to dlist and the record $(0, k|U, X, \mathcal{D}, Y)$ is added to elist . Note that a record $(k|U, Y, \mathcal{D}, X)$ appears in dlist if and only if $\mathcal{D}_{k|U}(Y)$ is asked first before the corresponding encryption query.

Now suppose there exists a record of the form $(1, \text{pw}|U_i, X, *, Y)$ in elist such that \mathcal{A}' makes a Send query on Y . This means that the encryption query $\mathcal{E}_{\text{pw}|U_i}(X)$ is asked first before the corresponding decryption query (otherwise, elist will contain a record of

the form $(0, \text{pw}|U_i, X, *, Y)$ instead). Consequently, the record $(\text{pw}|U_i, Y, \mathcal{D}, X)$ does not appear in dlist . This implies \mathcal{A}' gets the encryption capability by guessing the password and the event **Encrypt** occurs.

Suppose **Collision** is the event that the permutation property (bijection) of block cipher $(\mathcal{E}_{k|U}, \mathcal{D}_{k|U})$ does not hold. Now since the encryption and decryption lists are at most of size $q_{\mathcal{E}}$, the probability that the permutation property of block cipher does not hold is at most $\frac{q_{\mathcal{E}}^2}{2|S|}$. Hence $\text{Prob}[\text{Collision}] \leq \frac{q_{\mathcal{E}}^2}{2|S|}$.

Let $\text{Bad} = \text{Encrypt} \vee \text{Collision}$. If ever the event **Bad** occurs, adversary \mathcal{A} aborts and outputs a random bit. Otherwise, \mathcal{A} outputs whatever bit is eventually output by \mathcal{A}' . Note that since the password pw is selected by \mathcal{A} , it can detect occurrence of the event **Encrypt**. We provide details of oracle simulations below.

–*Execute queries:* Suppose \mathcal{A}' makes a query $\text{Execute}(\{(U_1, d_1), \dots, (U_n, d_n)\})$. It receives as output a transcript T of an execution of UP. It appends (S, T) to tlist . Adversary \mathcal{A} then expands the transcript T for the unauthenticated protocol UP into a transcript T' for the password-based protocol PAP according to the modification described in Section IV-C. \mathcal{A} is able to do this since the password pw is chosen by \mathcal{A} itself. \mathcal{A} then returns T' to \mathcal{A}' .

–*Send queries:* The first send query that \mathcal{A}' makes to an instance is to start a new session. We will denote such queries by Send_0 queries. To start a session among unused instances $\Pi_{U_1}^{d_1}, \dots, \Pi_{U_n}^{d_n}$, the adversary has to make the send queries in any order: $\text{Send}_0(U_1, d_1, \text{“start”}), \dots, \text{Send}_0(U_n, d_n, \text{“start”})$. When all these queries have been made, \mathcal{A} sets $S = \{(U_1, d_1), \dots, (U_n, d_n)\}$ and makes an **Execute** query to its own **execute** oracle. It receives a transcript T in return and stores (S, T) in the list tlist . Let us denote the send queries of first and second round by Send_1 and Send_2 respectively. Note that Send_0 and Send_2 queries have no output. Assuming that the event **Bad** does not occur, any subsequent **Send** query (i.e., after a Send_0 query) to an instance Π_U^i is on a properly encrypted message. Note since **Encrypt** does not occur, any attempt by \mathcal{A}' to encrypt and make a **Send** query will be detected and the protocol aborted. Adversary \mathcal{A}' would not be able to copy a correct message of a user U_k and make a **Send** query on behalf of U_l ($l \neq k$) as this would also be detected. So the **Send** query to Π_U^i is on a properly encrypted message.

\mathcal{A} now performs the action to be done by Π_U^i in the password-based protocol. This is done in the following manner: \mathcal{A} first finds the unique entry (S, T) in tlist such that $(U, i) \in S$. Such a unique entry exists for each instance by assumption. \mathcal{A} decrypts the current incoming messages. Note that \mathcal{A} can decrypt as he knows the password pw . From the transcript T , adversary \mathcal{A} finds the appropriate message, encrypts it to form the next public information with appropriate secret key of the form $\text{pw}|U$ to be output by Π_U^i and returns it to \mathcal{A}' .

–*Reveal/Test queries :* Suppose \mathcal{A}' makes the query $\text{Reveal}(U, i)$ or $\text{Test}(U, i)$ to an instance Π_U^i for which

TABLE I.
PROTOCOL COMPARISON

Protocol	Communication			Computation			
	R	BL	PTP	Exp	Mul + Div	Enc	Dec
BCP [13]	n	$n e $	$2n - 2$	$2n$	-	1	2
Our prot.	2	$2 e $	$2n - 2$	3	$2n$	2	$n + 1$

$\text{acc}_{\mathcal{U}}^i = 1$. At this point the transcript T' in which Π_U^i participates has already been defined. Now \mathcal{A} finds the unique pair (S, T) in tlist such that $(U, i) \in S$. Assuming that the event **Bad** does not occur, T is the unique unauthenticated transcript which corresponds to the transcript T' . Then \mathcal{A} makes the appropriate **Reveal** or **Test** query to one of the instances involved in T and returns the result to \mathcal{A}' .

As long as **Bad** does not occur, the above simulation for \mathcal{A}' is perfect. Whenever **Bad** occurs, adversary \mathcal{A} aborts and outputs a random bit. So $\text{Prob}_{\mathcal{A}', \text{PAP}}[\text{Succ}|\text{Bad}] = \frac{1}{2}$. Using this, we obtain $\text{Adv}_{\mathcal{A}, \text{UP}} \geq \text{Adv}_{\mathcal{A}', \text{PAP}} - \text{Prob}[\text{Bad}]$.

The total number of **Execute** queries made by \mathcal{A} is at most $q_{\mathcal{E}} + q_S/n$, where $q_{\mathcal{E}}$ is the number of **Execute** queries made by \mathcal{A}' . Also since $\text{Adv}_{\mathcal{A}, \text{UP}} \leq \text{Adv}_{\text{UP}}^{\text{KA}}(t', q_{\mathcal{E}} + q_S/n)$ by assumption, we obtain:

$$\text{Adv}_{\text{PAP}}^{\text{AKA}} \leq \text{Adv}_{\text{UP}}^{\text{KA}}(t', q_{\mathcal{E}} + q_S/n) + \text{Prob}[\text{Encrypt}] + \text{Prob}[\text{Collision}].$$

The event **Encrypt** occurs whenever a $\text{Send}(U, i, Y)$ query is asked by the adversary \mathcal{A}' where a record of the form $(1, \text{pw}|U, X, *, Y)$ appears in elist . We note that \mathcal{A}' may ask at most q_S **Send** queries and so is able to build at most q_S data (encryptions) by itself. Hence at most q_S passwords it might have tried in the on-line guessing. Moreover, as long as the exponents from Z_q^* chosen by the users are random, the plaintexts to be encrypted are completely indistinguishable from random plaintexts drawn from the domain of the encryption function from the point of view of \mathcal{A}' . Hence an off-line exhaustive search on passwords will not get any bias on the actual password pw . So we have $\text{Prob}[\text{Encrypt}] \leq \frac{q_S}{N}$. This yields the statement of the theorem. ■

VII. Efficiency

Table 1 compares our protocol and Bresson *et al.*'s password-based group key agreement protocol (BCP) [13] where the following notations are used– n and R are the total number of users in a group and total number of rounds respectively; BL is the maximum bit length of encrypted messages sent per user, PTP, Exp, Mul, Div respectively stands for maximum number of point-to-point communications, modular exponentiations, modular multiplications and divisions computed per user; Enc, Dec denote respectively the maximum number of symmetric key encryptions and decryptions performed per user; and $|e|$ is the maximum size of an encrypted plaintext.

Our protocol achieves efficiency in both communication and computation aspects as compared to other multi-party password-based key agreement protocol [13]. Our

protocol requires only 2 rounds which makes our protocol efficient from communication point of view. Each user sends one message in each round. The maximum bit length of messages sent during protocol execution is at most $2|e|$. Each group member performs at most 3 modular exponentiations, $2n$ multiplications and/or divisions, 1 division, 2 encryptions and $n + 1$ decryptions.

VIII. CONCLUSION

We address the problem of designing password-based authenticated key establishment protocols. A case study is made regarding the difficulties of converting a GKA protocol into password-based setting considering the DB protocol [15] – a variant of the constant round GKA protocol BD [14], [22]. The protocol DB is finally enhanced into a password-based GKA which is proven to be secure against dictionary attack under DDH assumption in the ideal cipher model. We complement these results with a security analysis in the formal security model of Bellare *et al.* [6] following the work of Bresson *et al.* [13].

REFERENCES

- [1] M. Abdalla, J. -M. Bohli, M. I. G. Vasco and R. Steinwandt. *(Password) Authenticated Key Establishment: From 2-Party To Group*. In proceedings of TCC 2007, LNCS 4392, pp. 499–514, Springer-Verlag, 2007.
- [2] M. Abdalla, E. Bresson, O. Chevassut and D. Pointcheval. *Password-based Group Key Exchange in a Constant Number of Rounds*. In proceedings of PKC 2006, LNCS 3958, pp. 427–442, Springer-Verlag, 2006.
- [3] M. Abdalla and D. Pointcheval, *A Scalable Password-based Group Key Exchange Protocol in the Standard Model*. In proceedings of ASIACRYPT 2006, LNCS 4284, pp. 332–347, Springer-Verlag, 2006.
- [4] M. Bellare and P. Rogaway. *Entity Authentication and Key Distribution*. In proceedings of Crypto 1993, LNCS 773, pp. 231–249, Springer-Verlag, 1994.
- [5] M. Bellare and P. Rogaway. *Provably Secure Session Key Distribution: The Three-party Case*. In proceedings of STOC 1995, pp. 57–66, ACM Press, 1995.
- [6] M. Bellare, D. Pointcheval, and P. Rogaway. *Authenticated Key Exchange Secure Against Dictionary Attacks*. In proceedings of Eurocrypt 2000, LNCS 1807, pp. 139–155, Springer-Verlag, 2000.
- [7] S. M. Bellovin and M. Merritt. *Augmented Encrypted Key Exchange : A Password-based Protocol Secure against Dictionary Attacks and Password File Compromise*. In proceedings of ACM CCS 1993, pp. 244–250, ACM Press, 1993.
- [8] J. -M. Bohli, M. I. G. Vasco and R. Steinwandt. *Password-Authenticated Constant-Round Group Key Establishment with a common reference string*. Available at <http://eprint.iacr.org/2006/214>.
- [9] V. Boyko, P. MacKenzie and S. Patel. *Provably Secure Password-authenticated Key Exchange using Diffie-Hellman*. In proceedings of Eurocrypt 2000, LNCS 1807, pp. 156–171, Springer-Verlag 2000.
- [10] E. Bresson, O. Chevassut and D. Pointcheval. *A Security Solution for IEEE 802.11's Ad-Hoc Mode: Password Authentication and Group Diffie-Hellman Key Exchange*. International Journal of Wireless and Mobile Computing (IJWMC), 2(1):413, 2007.
- [11] E. Bresson, O. Chevassut and D. Pointcheval. *New Security Results on Encrypted Key Exchange*. In proceedings of PKC 2004, LNCS 2947, pp. 145–158, Springer-Verlag, 2004.
- [12] E. Bresson, O. Chevassut and D. Pointcheval. *Proof of Security for Password-based Key Exchange (IEEE P1363 AuthA Protocol and Extensions)*. In proceedings of ACM CCS 2003, pp. 241–250, ACM Press.
- [13] E. Bresson, O. Chevassut and D. Pointcheval. *Group Diffie-Hellman Key Exchange Secure Against Dictionary Attack*. In proceedings of Asiacrypt 2002, LNCS 2501, pp. 497–514, Springer-Verlag, 2002.
- [14] M. Burmester and Y. Desmedt. *A Secure and Efficient Conference Key Distribution System*. In proceedings of Eurocrypt 1994, LNCS 950, pp. 275–286, Springer-Verlag, 1995.
- [15] R. Dutta and R. Barua. *Constant Round Dynamic Group Key Agreement*. In proceedings of ISC 2005, LNCS 3650, pp. 74–88, Springer-Verlag, 2005.
- [16] R. Dutta and R. Barua. *Password-Based Encrypted Group Key Agreement*. In the International Journal of Network Security (IJNS), Vol.3, No.1, pp. 23–34, 2006.
- [17] R. Dutta. *Multi-Party Key Agreement in Password-Based Setting*. In proceeding of First Asia International Conference on Modelling & Simulation, (AMS'07), pp. 133–138, IEEE Computer Society Press, 2007.
- [18] R. Gennaro and Y. Lindell. *A Framework for Password-based Authenticated Key Exchange*. In proceedings of Eurocrypt 2003, LNCS 2656, pp. 524–543, Springer-Verlag, May 2003.
- [19] O. Goldreich and Y. Lindell. *Session-key Generation using Human Memorable Passwords only*. In proceedings of Crypto 2001, LNCS 2139, pp. 408–432, Springer-Verlag, 2001.
- [20] S. Halevi and H. Krawczyk. *Public Key Cryptography and Password Protocols*. In ACM Transactions on Information and System Security, pp. 524–543, ACM Press, 1999.
- [21] D. P. Jablon. *Strong Password-only Authenticated Key Exchange*. In SIGCOMM Computer Communication Review, 26(5), pp.5–26, 1996.
- [22] J. Katz and M. Yung. *Scalable Protocols for Authenticated Group Key Exchange*. In proceedings of Crypto 2003, LNCS 2729, pp. 110–125, Springer-Verlag, 2003.
- [23] J. Katz, R. Ostrovsky and M. Yung. *Efficient Password-authenticated Key Exchange using Human-Memorable Passwords*. In proceedings of Eurocrypt 2001, LNCS 2045, pp. 475–494, Springer-Verlag, 2001.
- [24] S. M. Lee, J. Y. Hwang and D. H. Lee. *Efficient Password-Based Group Key Exchange*. In proceedings of TrustBus 2004, LNCS 3184, pp. 191–199, Springer-Verlag, 2004.
- [25] P. MacKenzie, S. Patel and R. Swaminathan. *Password-authenticated Key Exchange Based on RSA*. In proceedings of Asiacrypt 2000, LNCS 1976, pp. 599–613, Springer-Verlag, 2000.
- [26] V. Shoup. *On Formal Models for Secure Key Exchange*. In IBM Technical Report RZ 3120, 1999.

Ratna Dutta has completed her B.Sc (Mathematics) in 1996 and M.Sc. (Applied Mathematics) in 1998 from University of Calcutta, India. She received her PhD degree (Computer Science) from Indian Statistical Institute, India in 2006. Currently she is a Research Fellow at the Institute for Infocomm Research, Singapore. She has worked on public key cryptography, elliptic curve cryptosystem and pairings, dynamic group key agreement, password-based protocols, word-based public key cryptography and wireless sensor networks. Her current research interests include key management problem of broadcast encryption, private information retrieval from database and certificateless public key encryption. She is a member of IEEE Computer Society.