An Improved Dynamic Password based Group Key Agreement against Dictionary Attack

Wei Yuan
Department of Computer Science and Technology, Jilin University, Changchun, China
Email: yuanwei1@126.com

Liang Hu
Department of Computer Science and Technology, Jilin University, Changchun, China
Email: hul@mails.jlu.edu.cn

Hongtu Li
Department of Computer Science and Technology, Jilin University, Changchun, China
Email: li_hongtu@hotmail.com

Jianfeng Chu
Department of Computer Science and Technology, Jilin University, Changchun, China
Corresponding Author, Email: chujf@jlu.edu.cn

Yuyu Sun
College of Computer Science and Technology, Jilin University, Changchun 130012, China,
Software Institue, Changchun University, Changchun 130022, China
E-mail: sunyy@ccu.edu.cn

***Abstract*—Key exchange protocol is fundamental for establishing secure communication channels over public networks. Password-based key exchange protocols allow parties to share secret key in an authentic manner based on an easily memorizable password. Recently, a password-based group key agreement based on Joux's tripartite key agreement is proposed to improve the performance when users join or leave the group. In this paper, we employ an online dictionary attack on this protocol to show that such kind of modification cannot achieve the basic security of password based group key agreement. With this method, an adversary can test several passwords in one session, which leads the key space reduces greatly to the potential adversaries. To fill the gaps, we propose an improved protocol, which can avoid this attack. Finally, we prove the security of our protocol under the random oracle and ideal cipher model.**

***Index Terms*—Password-based, Group key agreement, Cryptanalysis, Random oracle model, Ideal-cipher model, MDDH**

I. INTRODUCTION

The group key agreement [1,2,3] can be regarded as the generalization of two-party key agreement. It allows a group of parties to exchange information among them over an insecure network and agree on a common key. Thereafter, the key can be used to some desirable security services, encryption and data integrity.

Based on the number of the users participating into the key agreement, these protocols are sorted into three kinds: two-party case, three party case, and group case. The first key agreement protocol was proposed by Diffie-Hellman [4] in 1976; it is the two-party case. Then many papers have extended this two-party protocol to the three-party setting [5] and group setting [6]. However, the Diffie-Hellman protocol and its extensions do not provide the authentication mechanism, and therefore suffer from the man-in-the-middle attack [7] easily. To solve this issue, over the past years, bulks of key agreement protocols [8, 9, 10] with authentication function have been developed.

The password-based key agreement protocols [11,12] require users only to remember a human-memorable low-entropy password, which is rather simple to users. Password-based key agreement protocols are widely used for user authentication and secure communications in real applications, such as internet banking and remote user authentication. The problem of designing a secure password-based key agreement protocol was proposed by Bellovin and Merritt [13] in 1992, and has since been studied extensively.

In 2005, Abdalla and Pointcheval [14] proposed a simple two-party password-based authenticated key protocol, which has been proven secure on the basis of the Diffie-Hellman problem. In this scheme, a user, A, sends only one ciphertext, X^*, to B, and B only needs to return one ciphertext, Y^*, to A as well. Then they can work out a common session key, and it is quite simple. The main function of this protocol is equivalent to agree a high-entropy session key with a low-entropy password. However, this protocol cannot be applied to the practical multi-party communications, since it requires each pair of potential communicating parties to share a password. And this leads that a large number of parties result in an even larger number of passwords to be shared. It is due to this

doi:10.4304/jsw.7.7.1524-1530

problem, in 2007, Rongxing Lu and Zhenfu Cao [15] proposed a simple three-party password-based key agreement protocol, which is developed from Abdalla and Pointcheval's protocol. In this protocol, each client first shares a human-memorable password with a trusted server, and then when two clients want to agree on a session key, they resort to the trusted server to authenticate each other. This model is more compatible with the Internet.

In 2007, reference [16] proposed a password-based group key agreement based on Joux's tripartite key agreement [17]. In this paper, we make an online dictionary attack on this protocol. With this method, an adversary can test several passwords in one session. Then we propose an improved protocol to fix this gap. Finally, we prove the security of our protocol under the random oracle and ideal cipher model.

II. PRELIMINARIES

A. Bilinear pairing

Let G_1 be a cyclic additive group generated by P, whose order is a prime q, and G_2 be a cyclic multiplicative group with the same order q. A bilinear pairing is a map $e: G_1 \times G_1 \to G_2$ with the following properties:

1. Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1, a, b \in Z_q$.
2. Non-degenerative: There exists $P, Q \in G_1$ such that $e(P, Q) \neq 1$.
3. Computable: There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

B. Computational problems

Let G_1 and G_2 be two groups of prime order q, let $e: G_1 \times G_1 \to G_2$ be a bilinear pairing and let P be a generator of G_1.

- Discrete Logarithm Problem (DLP)

Let G_1 and G_2 be two groups of prime order q, let $e: G_1 \times G_1 \to G_2$ be a bilinear pairing and let g be a generator of G_1. The discrete logarithms (DL) problem can be expressed as follows.

Given $P, g \in G_1$, to find $n \in Z_q$ such that $P = g^n$.

- Decisional Diffie-Hellman problem (DDH)

Let G be a finite cyclic group of prime order q. Given $\Gamma_{real} = \{g, g^x, g^y, g^{xy}\}$ and $\Gamma_{rand} = \{g, g^x, g^y, g^z\}$ where $x, y, z \in Z_q$, it is difficult to distinguish between g^z and g^{xy}. Formally, define the advantage function $Adv_G^{DDH}(\mathcal{A}) = |\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{A}(Y) = 1]|$ where $X \in \Gamma_{real}, Y \in \Gamma_{rand}$, we say that the DDH problem is hard in group G if $Adv_G^{DDH}(\mathcal{A})$ is negligible for any probabilistic polynomial time adversary $\mathcal{A}$. $Adv_G^{DDH}(t)$ is the maximum value of $Adv_G^{DDH}(\mathcal{A})$ running in time at most t.

- Multi-Decisional Diffie-Hellman problem (MDDH)

Given $\Pi_{real} = \{g, \{g^{x_i}\}_{1 \le i \le n}, g^{x_1 \cdot x_2 \cdot \cdots \cdot x_n}\}$ and $\Pi_{rand} = \{g, \{g^{x_i}\}_{1 \le i \le n}, g^y\}$, where $x_1, \ldots, x_n, y \in Z_q$, it is difficult to distinguish between $g^{x_1 \cdot x_2 \cdot \cdots \cdot x_n}$ and g^y. Define the advantage function $Adv_G^{MDDH}(\mathcal{A}) = |\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{A}(Y) = 1]|$, where $X \in \Gamma_{real}, Y \in \Gamma_{rand}$, the DDH problem is hard in group G if $Adv_G^{MDDH}(\mathcal{A})$ is negligible for any probabilistic polynomial time adversary $\mathcal{A}$. $Adv_G^{MDDH}(t)$ is the maximum value of $Adv_G^{MDDH}(\mathcal{A})$ running in time at most t.

C. Security definitions

In 2005, reference [12, 14] proposed the real-or-random (ROR) model instead of the find-and-guess model of Bellare and Rogaway [18] to prove a three-party password-based authenticated key exchange protocol. This model seems more suitable for the password-based setting and we shall prove our scheme under this model.

A player may have numerous instances, called oracles, of distinct concurrent executions of the protocol. We denote the j-th instance of U_i by U_i^j. The interaction between the adversary $\mathcal{A}$ and players occurs only via oracle queries, which describe the capabilities of $\mathcal{A}$. In the ROR model, Reveal queries are replaced by Test queries; Execute queries are introduced to model passive attack and can easily be simulated with the Send queries. The Send query and Test query are described as follows:

Send (U_i^j , m): This query models an active attack. $\mathcal{A}$ can intercept a message and then either modify it or create a new one to the intended player. The output of this query is the response generated by the instance U_i^j upon receipt of the message m according to the execution of protocol P. The adversary can initiate the execution of P by sending a query (U_i^j , start).

Test (U_i^j): This query models the indistinguishability of the real session key from a random string. Once the instance U_i^j has accepted a session key, the adversary can attempt to distinguish it from a random key as the basis of determining security of the protocol. A random bit b is chosen and if b=1 then the real session key is returned while if b=0 then a random key is return. Adversary outputs a guess bit b'. If $b' = b$, where b is the hidden bit used by U_i^j, $\mathcal{A}$ wins the game.

III. THE PROPOSED IMPROVED DYNAMIC PASSWORD SCHEME

A. Review of the original protocol

The algorithm for odd users is different from that for even users in reference [16]. Since the even-user

algorithm is based on the odd-user one, they firstly describe the odd-user algorithm. Assume that the system pre-distributes a common password pw for each user. There exists a secure cipher scheme (E, D), where E is an encrypt algorithm, and D is the matching decrypt algorithm. $H:\{0,1\}^{*}\to\{0,1\}^{l}$ is a hash function to generate the encryption key. The basic scheme for odd-user is as follows:

Step1. The user $U_i\in\{U_1,\ldots,U_n\}$ computes the key $k_i=H(U_i\,\|\,pw\|\,i)$, choose $x_i\in Z_q^{*}$ randomly, computes $X_i=g^{x_i}$, and then sends $X_i^{*}=E_{k_i}(X_i)$ to his neighbors U_{i-2} , U_{i-1} , U_{i+1} , U_{i+2} . Assume $U_{n+1}=U_1$, $U_{n+2}=U_2$, and $Q=e(g,g)$.

Step2. After receiving X_j^{*} , the user U_i computes $k_j=H(U_j\,\|\,pw\|\,j)$ and $X_j=D_{k_j}(X_j^{*})$, and then computes U_i^R and U_i^L respectively.

$U_1^R=e(X_2,X_3)^{x_1}=Q^{x_1x_2x_3}$,

$U_1^L=e(X_{n-1},X_n)^{x_1}=Q^{x_{n-1}x_nx_1}$

$\cdots$,

$U_i^R=e(X_{i+1},X_{i+2})^{x_i}=Q^{x_ix_{i+1}x_{i+2}}$,

$U_i^L=e(X_{i-2},X_{i-1})^{x_i}=Q^{x_{i-2}x_{i-1}x_i}$

$\cdots$,

$U_n^R=e(X_1,X_2)=Q^{x_1x_2x_n}$,

$U_n^L=e(X_{n-2},X_{n-1})^{x_n}=Q^{x_{n-2}x_{n-1}x_n}$

Step3. The user U_i computes $K_i=U_i^R/U_i^L$ and broadcasts the value to all the members in the group over the Internet. After this phase, the user U_i should have the following values

$K_1=U_1^R/U_1^L=Q^{x_1x_2x_3-x_{n-1}x_nx_1}$,

$\cdots$,

$K_i=U_i^R/U_i^L=Q^{x_ix_{i+1}x_{i+2}-x_{i-2}x_{i-1}x_i}$

$\cdots$,

$K_n=U_n^R/U_n^L=Q^{x_nx_1x_2-x_{n-2}x_{n-1}x_n}$

Step4. Before computing a common session key, the user U_i should verify the values got from other group members. Since $U_i^R=U_{i+2}^L$, the user U_i can compute as follows

$U_{i+2}^{R'}=K_{i+2}\cdot U_{i+2}^{L'}=K_{i+2}\cdot U_{i+2}^{R}$,

$\cdots$,

$U_i^{L'}=U_{i-2}^{R'}=K_{i-2}\cdot U_{i-2}^{L'}$

and verify the equation

$U_i^{L'}\overset{?}{=}U_i^L$

If the above equation holds, the values that U_i gets from his neighbors are correct. Otherwise, stop the protocol and output error message.

Step5. When the user U_i decides the above steps are finished, he will compute each U_i^R . Since $U_i^R=U_{i+2}^L$, each user can compute $U_{i+2}^R=K_{i+2}\cdot U_i^R$. Then the common session key is computed.

$sk_{odd}=U_1^RU_2^R\cdots U_n^R$

In the case of even users, the users should implement the following additional steps.

Step6. Suppose that there exists a user set $\{U_1,U_2,\ldots,U_n,U_{n+1}\}$ and the user $\{U_1,U_2,\ldots,U_n\}$ have negotiated a common session key by above steps. User $\{U_1,U_n\}$ send X_1^{*} and X_n^{*} to U_{n+1} respectively. User U_{n+1} computes the key $k_{n+1}=H(U_{n+1}\,\|\,pw\|\,n+1)$, chooses $x_{n+1}\in Z_q^{*}$ uniformly at random, computes $X_{n+1}=g^{x_{n+1}}$, and then sends $X_{n+1}^{*}=E_{k_{n+1}}(X_{n+1})$ to U_1 and U_n . Upon receiving X_{n+1}^{*} , users $\{U_1,U_n\}$ compute k_{n+1} and $X_{n+1}=D_{k_{n+1}}(X_{n+1}^{*})$ respectively. Then the users U_1,U_n and U_{n+1} compute $Z_{n+1}=Q^{x_1x_nx_{n+1}}$ as their temporary session key.

Step7. The users U_1,U_n encrypt the common session key Z_{n+1} using sk_{odd} as the secret key, and broadcast the corresponding ciphertext to other user. Thereby, any user who obtains the session key Z_{n+1} and sk_{odd} can computes the updated session key

$sk=H(sk_{odd},Z_{n+1})$.

Thereafter, users U_1 and U_n encrypt sk using Z_{n+1} as the secret key, and send the corresponding ciphertext to user U_{n+1} . After implementing above steps, all the users share the updated common session key sk.

B. Cryptanalysis of the original protocol

Since our online dictionary attack is for the basic odd-user protocol, the other part of their protocol is left out of this paper. Meanwhile, the attacking process is similar with the even user setting.

The attack is described as follows. Let k be the number of honest players. The adversary starts a session in which all the honest players have indices of the form $3(i-1)+3$ for $i=1,\ldots,k$. The adversary plays the role of player $3(i-1)+1$ and $3(i-1)+5$ .we also assume $U_{n+1}=U_1$. There are 3k players in all. Then, let $\{pw_1,\ldots,pw_m\}$ be a list of candidate passwords that an adversary wants to try. The adversary gets out k candidate passwords to test in this message.

1. He computes $k_{3(i-1)+1}=H(U_{3(i-1)+1}\,\|\,pw_i\,\|\,3(i-1)+1)$ and $k_{3(i-1)+5}=H(U_{3(i-1)+5}\,\|\,pw_i\,\|\,3(i-1)+5)$, $i=1,\ldots,k$. Then he chooses $x_1,x_5,\ldots,x_{3(k-1)+1},x_{3(k-1)+5}\in Z_q$, computes corresponding $X_i=g^{x_i}$, and broadcasts $X_i^{*}=E_{k_i}(X_i)$.

2. He decrypt $X_{3(i-1)+1}=D_{k_{3(i-1)+1}}(X_{3(i-1)+1}^{*})$ and $X_{3(i-1)+5}=D_{k_{3(i-1)+5}}(X_{3(i-1)+5}^{*})$ with the guessed pw_i , tests whether $z_{3(i-1)+5}^L/z_{3(i-1)+1}^R=K_{3(i-1)+3}$ hold.

Therefore, the Adversary can test k candidate passwords from the list with one session.

C. the proposed password based group key agreement protocol

Assume that the system pre-distributes a common password pw for each player and $U_{n+1} = U_1$. There exists a secure cipher scheme (E,D), where E is an encrypt algorithm. $H : \{0,1\}^* \to \{0,1\}^{l_H}$ is a hash function to generate the encryption key and $H_1 : \{0,1\}^* \to \{0,1\}^{l_{H_1}}$ is a hash function to update the session key. e is a bilinear pairing, $Q = e(g, g)$ is public.

Step 1. The player $U_i \in \{U_1,\ldots,U_n\}$ computes the key $k_i = H(U_i \| pw \| i)$, choose $x_i \in Z_q^*$ randomly, computes $X_i = g^{x_i}$, and then sends $X_i^* = E_{k_i}(X_i \| k_i)$ to his neighbors $U_{i-2}, U_{i-1}, U_{i+1}, U_{i+2}$.

Step 2. After receiving X_j^*, the player U_i computes $k_j = H(U_j \| pw \| j)$ and decrypts $X_j \| k_j = D_{k_j}(X_j^*)$ with it. Then he check whether the k_j decrypted from X_j^* is identical to the k_j computed from pw and the user index j. After this verification, he computes $U_i^L = e(X_{i-2}, X_{i-1})^{x_i} = Q^{x_{i-2}x_{i-1}x_i}$, $U_i^R = e(X_{i+1}, X_{i+2})^{x_i} = Q^{x_i x_{i+1}x_{i+2}}$, $K_i = U_i^R / U_i^L$ and broadcast K_i to all the members in the group over the network.

Step 3. Since $U_i^R = U_{i+2}^L$, U_i computes $U_{i+2}^R = K_{i+2}U_i^R$ and gets all U_i^R s. Then he computes the session key $sk = U_1^R U_2^R \cdots U_n^R$ if n is an odd number. If n is an even number, the Steps4 and Step5 are with the same function as the Step 6 and Step 7 of the original protocol.

Step 4. Suppose that there exists a user set $\{U_1,U_2,\ldots,U_n,U_{n+1}\}$ and the user $\{U_1,U_2,\ldots,U_n\}$ have negotiated a common session key by above steps. User $\{U_1,U_n\}$ send X_1^* and X_n^* to U_{n+1} respectively. User U_{n+1} computes the key $k_{n+1} = H(U_{n+1} \| pw \| n+1)$, chooses $x_{n+1} \in Z_q^*$ uniformly at random, computes $X_{n+1} = g^{x_{n+1}}$, and then sends $X_{n+1}^* = E_{k_{n+1}}(X_{n+1} \| k_{n+1})$ to U_1 and U_n. Upon receiving X_{n+1}^*, users $\{U_1,U_n\}$ compute k_{n+1} and $X_{n+1} \| k_{n+1} = D_{k_{n+1}}(X_{n+1}^*)$ respectively. Then they check whether the two k_{n+1} are identical. Finally, the users U_1, U_n and U_{n+1} compute $Z_{n+1} = Q^{x_1 x_n x_{n+1}}$ as their temporary session key if the two k_{n+1} are identical.

Step 5. The users U_1, U_n encrypt the common session key Z_{n+1} using sk_{odd} as the secret key, and broadcast the corresponding ciphertext to other users. Thereby, any user who obtains the session key Z_{n+1} and sk_{odd} can computes the updated session key

$sk = H_1(sk_{odd}, Z_{n+1})$.

Thereafter, users U_1 and U_n encrypt sk using Z_{n+1} as the secret key, and send the corresponding ciphertext to user U_{n+1}. After above steps, all the users share the updated common session key sk.

IV. SECURITY ANALYSIS OF OUR IMPROVED PROTOCOL

The following Theorem 1 presents the main security result of the proposed password-based group key agreement protocol

Theorem 1. Let P denote the proposed protocol in which the password is chosen in a dictionary of size N. For any adversary $\mathcal{A}$ running in time t, that makes at most q_{active} attempts within at most $q_{session}$ sessions, his advantage in breaking the semantic security of the session key, in the ideal-cipher model, is upper-bounded by:

$$\begin{aligned} Adv_P^{ror-aka}(\mathcal{A}) \le\ & 2q_D q_H + 2q_H^2 / 2^{l_H} + 2q_{H_1}^2 / 2^{l_{H_1}} \\ & + 2nq_{session}q_E + 2q_{H_1} + (q_E + q_D)^2 / |G| \\ & + 2\frac{nq_{ession}Adv_G^{ddh}(t) + q_{H_1}/|G| + q_{active}/N}{2^{l_H}} \\ & + 2nq_{ession}Adv_G^{ddh}(t) \end{aligned}$$

where q_H, q_{H_1}, q_E, q_D denote the number of oracle queries the adversary is allowed to make to the random oracles H and H_1, and to the ideal-cipher oracles E and D, respectively.

This theorem shows the advantage of the adversary essentially grows linearly with the number of active attempts that the adversary makes and the passive attacks are essentially negligible because an honest transcript does not help a computationally bounded adversary in guessing the password.

Proof. We incrementally define a sequence of games from the game Game0 to Game7. In each game, various adversary behaviors are simulated and the advantages of an adversary $\mathcal{A}$ are upper-bounded. At the end of the games, we measure the probability $|\Pr[Suc_i] - \Pr[Suc_{i-1}]|$ between $Game_i$ and $Game_{i-1}$. Finally, we get the result of the Theorem 1 by using the difference of the probability.

Game0. This experiment simulates the real attack. The advantage of $\mathcal{A}$ in this protocol is defined as $Adv_P^{ror-aka}(\mathcal{A}) = 2\Pr[Suc_0] - 1$

Game1. In this game, we simulate the random oracles H and H_1 by maintaining the list L_H and L_{H_1}, respectively.

If $\mathcal{A}$ asks a H query of the form (U_i, i, pw) such that a record (U_i, i, pw, r) exists in the list L_H, then r is returned. Otherwise, r is chosen randomly from $\{0,1\}^{l_H}$, and (U_i, i, pw, r) is recorded to L_H. Define the collision event in the output of H by Col_H. Then the probability of that bad event is upper-bounder by $q_H^2 / 2^{l_H}$. Similarly, the probability of the collision event Col_{H_1} in the output

of H_1 is upper-bounder by $q_{H_1}^2 / 2^{l_{H_1}}$. Game0 and Game1 are perfectly indistinguishable unless that the bad events Col_H and Col_{H_1} occur simultaneously. Thus, we have

$$|\Pr[Suc_1] - \Pr[Suc_0]| \leq q_H^2 / 2^{l_H} + q_{H_1}^2 / 2^{l_{H_1}} .$$

Game2. This game simulates the ideal-cipher oracles E and D by maintaining a list $L_{E,D}$, which keeps track of the previous queries-answers and links each query to a specific player. $L_{E,D}$ has the form (U_i, i, e, type, k, X, X*), where type ∈ {enc, dec}. Such record means that $X^* = \mathcal{E}_k(X \| k)$, and type indicates which kind of queries generated the record. The index i indicates which player is associated with the key k, while U_i indicates the user. These values are all set to null if k does not come from a H query of the form (U_i, i, *) with $i \in \{1,\ldots,n\}$. The e will be explained in the next game. E and D can be simulated as follows:

Encryption query: For an encryption query $\mathcal{E}_k(X \| k)$, if a record (• , • , • , • , k, X, *) exists in the list $L_{E,D}$, the element * is returned. Otherwise, a random value $X^* \in G$ is returned and (null, null, null, enc, k, X,*) is added into $L_{E,D}$.

Decryption query: For a decryption query $D_k(X^*)$, if a record (• , • , • , • , k, *, X*) exists in the list $L_{E,D}$, the element * is returned. Otherwise, if k has been returned to a hash query of the form (U_i, i, *), we choose X randomly in $G \setminus \{0\}$ and update the list $L_{E,D}$ with (U_i, i, • , dec, k, X, X*); otherwise, we choose X randomly in $G \setminus \{0\}$ and update the list $L_{E,D}$ with (• , • , • , dec, k, X, X*). Finally, X and is returned.

The above simulation is perfect, except for the following two bad events. First, collisions may appear that contradict the permutation property of the ideal-cipher. The probability can be upper-bounded by $(q_E + q_D)^2 / 2|G|$. Second, in the case of the decryption query simulation, one will abort executions if the value k involved in a decryption query is output by H. The probability is at most $q_H / 2^{l_H}$ for each decryption query. For any k involved in a decryption query, if k comes from an H query, we know the corresponding pair (U_i, i). Thus, we have

$$|\Pr[Suc_2] - \Pr[Suc_1]| \leq (q_E + q_D)^2 / 2|G| + q_D q_H / 2^{l_H}$$

Game3. In this game, we change the simulation of the decryption queries, and make use of our challenger to embed an instance of the MDDH problem in the protocol simulation. Let the challenger output tuples $(\gamma_1, \gamma_2, \ldots, \gamma_n, \lambda_1, \lambda_2, \ldots, \lambda_n)$. We use these tuples to properly simulate the decryption queries. More preciously, we make a new tuple each time a new session appears in a decryption query. However, if several queries are asked with the same session, the challenger outputs the same tuple.

Given a tuple outputted by the challenger, for any randomly chosen $(e_1, e_2, \ldots, e_n)$, the tuple $(\gamma_1^{e_1}, \gamma_2^{e_2}, \ldots, \gamma_n^{e_n}, \lambda_1^{e_1 e_2}, \lambda_2^{e_2 e_3}, \ldots, \lambda_n^{e_n e_1})$ has the same distribution as the original tuple. We make this property as follows, by modifying the sub-case previously considered for new decryption queries in the Game3.

Decryption query: For a decryption query $D_k(X^*)$ such that $k = H(U_i, i, *)$ was previously obtained from H for some valid index i, we query challenger for getting a tuple $(\gamma_1, \gamma_2, \ldots, \gamma_n, \lambda_1, \lambda_2, \ldots, \lambda_n)$. Then we choose $e \in Z_q^*$ randomly, add the record (U_i , i, e, dec, k, $X = \gamma^e$, X*)into the list $L_{E,D}$, and return X.

The list $L_{E,D}$ whose records are of the form (U_i, i, e, type, k, X, X*) has been defined. Above change of simulation on the decryption queries does not modify the view of the adversary. So:

$$\Pr[Suc_3] = \Pr[Suc_2]$$

Game4. In this game, we simulate the Send query in the first and the second round. When the session starts, player i computes the symmetric keys as $k_j = H(U_j, j, pw)$, for all player j. Thus, we are working with the tuple $(\gamma_1, \gamma_2, \ldots, \gamma_n, \lambda_1, \lambda_2, \ldots, \lambda_n)$.

In the Step1, U_i randomly chooses a value $X_i^* \in G$ to be broadcasted, and asks $D_{k_i}(X_i^*)$ with the simulation in Game3. The simulation leads to add e_i to the list $L_{E,D}$, unless X_i^* already appeared as an encryption result. But the latter event can not happen with probability greater than $q_E / |G|$.

In the Step2, U_i recovers all $X_j \| k_j = D_{k_j}(X_j^*)$, and checks whether $k_j = H(U_j, j, pw)$. If X_j^* has been simulated according to the above simulation of the Step1, one gets e_{i-1} and e_{i+1} in the list $L_{E,D}$ such that $X_j = \gamma_j^{e_j}$. Otherwise, one of the X_j^* has been previously answered by the encryption oracle in response to an attacker query $E_k(X \| k)$, where $k = H(U_j, j, pw)$ is the correct key for player U_j. We denote such an event by Encrypt. In such a case, the simulation is terminated and the adversary wins. Thus, one gets $X_{i-2} = \gamma_{i-2}^{e_{i-2}}$, $X_{i-1} = \gamma_{i-1}^{e_{i-1}}$, $X_i = \gamma_i^{e_i}$, $X_{i+1} = \gamma_{i+1}^{e_{i+1}}$, $X_{i+2} = \gamma_{i+2}^{e_{i+2}}$ correctly computes $U_i^L = \lambda_{i-1}^{e_{i-2} e_{i-1} e_i}$, $U_i^R = \lambda_i^{e_{i-2} e_i e_{i+1}}$, then broadcasts $K_i = U_i^R / U_i^L$. After this round, each player can compute the session key as before. The simulation is still perfect, unless the above bad events happen. Therefore, we get

$$|\Pr[Suc_4] - \Pr[Suc_3]| \leq q_{passive} \cdot q_E / |G| + \Pr[Encrypt_1] / 2^{l_H}$$
$$\leq n q_{session} q_E / |G| + \Pr[Encrypt_1] / 2^{l_H}$$

Game5. Since it is clear that the security of the above protocol still relies on the DDH assumption, let the

challenger output tuples $(\gamma_1, \gamma_2, \ldots, \gamma_n, \lambda_1, \lambda_2, \ldots, \lambda_n)$ according to the Π_{rand} distribution. We have

$$|\Pr[Suc_5] - \Pr[Suc_4]| \le q_{ession} Adv_G^{mddh}(t) \le nq_{ession} Adv_G^{ddh}(t)$$

$$|\Pr[Encrypt_2] - \Pr[Encrypt_1]| \le q_{ession} Adv_G^{mddh}(t)$$
$$\le nq_{ession} Adv_G^{ddh}(t)$$

Game6. Since the session key is computed from U_i^R and K_i, the collision has been upper-bounded by above games. In this game, we derive the session keys using a private random oracle $H_1^{'}: sk = H_1^{'}(sk_{odd}, Z_{n+1})$ to simulate the updated session key. After the modification of the derivation of the session key, the probability for the adversary to see the difference between the current and the previous games is to query $H_1: sk = H_1(sk_{odd}, Z_{n+1})$. From the previous game, we know no information has been leaked about sk and these queries are identical inside each session: the probability of such an event can also be upper-bounded by $q_{\mathcal{G}} / |G|$. Thus, we have

$$|\Pr[Suc_6] - \Pr[Suc_5]| \le q_{H_1} / |G|$$

$$|\Pr[Encrypt_3] - \Pr[Encrypt_2]| \le q_{H_1} / |G|$$

Since the private oracle $\mathcal{G}^{'}$ is private to the simulator, it is clear that

$$\Pr[Suc_6] = 1/2.$$

Game7. One can note that the password pw is only used in the simulation of the first and second rounds to compute k_i with the element γ_i. But only K_i which is computed from λ_{i-1} and λ_i is outputted. In this game, we can simplify the simulation of the second and third rounds as follows: In the first round, U_i randomly chooses $X_i^* \in G$, and sends it with no decryption. In the second round, U_i simply computes and sends $K_i = \lambda_i / \lambda_{i-1}$. This simulation is perfect since one does not need anymore to compute the session key. Thus, the probability of the Encrypt event is less than the number of first flows manufactured by the adversary. We have

$$\Pr[Encrypt_3] \le q_{active} / N$$

In the above, the collisions in the output have been eliminated in previous games and we can get the Theorem 1.

V. CONCLUSION

Password-based group key agreement is an important research field in key agreement protocol and plays an outstanding role in distributed networks. In this paper, we make an online dictionary attack on this protocol. With this method, an adversary can test several passwords in one session. Then we propose an improved protocol to fix this gap. Finally, we prove the security of our protocol under the random oracle and ideal cipher model.

ACKNOWLEDGMENT

The authors would like to thank the editors and anonymous reviewers for their valuable comments. This work is supported by the National Natural Science Foundation of China under Grant No. 60873235 and 60473099, the National Grand Fundamental Research 973 Program of China (Grant No. 2009CB320706), Scientific and Technological Developing Scheme of Jilin Province (20080318), the National High Technology Research and Development Program 863 of China under Grant No. 2011AA010101. and Program of New Century Excellent Talents in University (NCET-06-0300).

REFERENCES

[1] Xianfeng Guo, Jiashu Zhang, Secure group key agreement protocol based on chaotic Hash, Information Sciences 2010, 180: 4069-4074.
[2] Song Han, Security of a key agreement protocol based on chaotic maps, Chaos, Solitons and Fractals 2008, 38: 764-768.
[3] Ming-Hui Zheng, Hui-Hua Zhou, Jun Li, Guo-Hua Cui, Efficient and provably secure password-based group key agreement protocol, Computer Standards & Interfaces 31 (2009) 948–953.
[4] W. Diffie and M.E. Hellman, New Directions in Cryptography, IEEE Trans. Information Theory 1976, IT-22 (6): 644-654.
[5] H. Y. Chen, T. C. Wu, Provably Secure Password-Based Three-Party Key Exchange With Optimal Message Steps 52 (6) (2009) 646-655.
[6] M. K. Boyarsky, Public-key cryptography and password protocols: The multi-user case. ACM CCS 99: 6th Conference on Computer and Communications Security, 63-72.
[7] R. C. W. Phan, W. C. Yau, B. M. Goi, Cryptanalysis of simple three-party key exchange protocol, Information Sciences 178 (2008) 2849–2856.
[8] J. Katz, M. Yung, Scalable protocols for authenticated group key exchange, Crypto 2003, Lecture Notes in Computer Science (2729) (2003) 100-125.
[9] M. Cagalj, S. Capkun, Key Agreement in Peer-to-Peer Wireless Networks, Proceedings of the IEEE, 94 (2) (2006) 467-478.
[10] E. Bresson, O. Chevassut, D. Pointcheval, Group Diffie-Hellman key exchange secure against dictionary attack, ASIACRYPT 2002, Lecture Notes in Computer Science, vol. 3386, 2005, pp.65-84.
[11] M. Abdalla, E. Bresson, O. Chevassut, D. Pointcheval, Password-based group key exchange in a constant number of rounds, PKC 2006, Lecture Notes in Computer Science, vol. 3958, 2006, pp.427-442.
[12] M. Abdalla, P. A. Fouque, D. Pointcheval, Password-based Authenticated Key Exchange in the Three-Party Setting, PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography, Lecture Notes in Computer Science 3386 (2005) 65-84.
[13] S. M. Bellovin and M. Merritt, Encrypted key exchange: Password-based protocols secure against dictionary attacks, 1992 IEEE Symposium on Security and Privacy, 1992, 72-84.
[14] M. Abdalla, D. Pointcheval, Simple password-based authenticated key protocols, Topics in Cryptology- CT-RSA 2005, Lecture Notes in Computer Science, vol. 3376, 2005, pp.191-208.
[15] R. Lu, Z. F. Cao, Simple three-party key exchange protocol, Computers and Security 26 (1) (2007) 94-97.

[16] C. B. Ma, J. Ao, J. H. Li, Password-based Dynamic Group key Agreement, 2007 International Conference on Network and Parallel Computing, pp.203-208.
[17] A. Joux, A One Round Protocol for Tripartite Diffie-Hellman, Lecture Notes in Computer Science 1838 (2000) 385-394.
[18] M. Bellare, D. Pointcheval, and P. Rogaway, Authenticated key exchange secure against dictionary attacks, Advances in Cryptology-EUROCRYPT 2000, Lecture Notes in Computer Science (1807) (2000) 139-155.

Wei Yuan was born in Chengde of Hebei province of China in 1984. He began the study of computer science at Jilin University in 2003 and got his bachelor degree in 2007. Then he continued his research on information security and received his master degree in 2010. Now he is a PhD candidate of the college of computer science and technology of Jilin University.

His main research interests include cryptography and information security. he have participated in several projects include two National Natural Science Foundations of China and one National Grand Fundamental Research 973 Program of China and published more than 10 research papers from 2007.

Liang Hu was born in 1968. He has his BS degree on Computer Systems Harbin Institute of Technology in 1993 and his PhD on Computer Software and Theory in 1999. Currently, he is the professor and PhD supervisor of College of Computer Science and Technology, Jilin University, China.

His main research interests include distributed systems, computer networks, communications technology and information security system, etc. As a person in charge or a principal participant, Dr Liang Hu has finished more than 20 national, provincial and ministerial level research projects of China.

Hongtu Li was born in Siping of Jilin, China on Mar. 17 1984. In 2002, Li Hongtu began the study of computer science at Jilin University in Jilin, Changchun, China. And in 2006, Li Hongtu got bachelor's degree of computer science. In the same year, Li Hongtu began the master's degree study in network security at Jilin University. After 3 years study, Li Hongtu got his master's degree in 2009. From then on, Li Hongtu began the doctor's degree in the same field of study at the same University.

From 2009, he has got a fellowship job. He worked in grid and network security laboratory as an ASSISTANT RESEACHER at Jilin University. From 2006 to now, he has published several papers. The list of published articles or books is as follows:

"Identity -Based Short Signature Without Random Oracles Model", International Conference of ICT Innovation and Application-ICIIA2008, Guangzhou, China, 2008.

"Registration and private key distribution protocol based on IBE", the 5th International Conference on Frontier of Computer Science and Technology-FCST2010, Changchun, China, 2010.

"Certificateless authenticated key agreement protocol against KCI and KRA", The 2011 International Conference on Network Computing and Information Security-NCIS'11 and the 2011 International Conference on Multimedia and Signal Processing-CMSP'11, Guilin, China, 2011.

Expect network security, he also interested in grid computing, wireless networks, intrusion detection and so on. From 2006 to now, he have participated in or led several projects include two National Natural Science Foundations of China and one National Grand Fundamental Research 973 Program of China.

Jianfeng Chu, born in 1978, Ph.D. , Now he is the teacher of the College of Computer Science and Technology, Jilin University, Changchun, China. He received the Ph.D. degree in computer structure from Jilin University in 2009. His current research interests focus on information security and cryptology.

An important objective of the projects is to probe the trend of network security, which can satisfy the need of constructing high-speed, large-scale and multi-services networks. Various complex attacks can not be dealt with by simple defense. And to add mechanisms to network architecture results in decreasing performance. In a word, fundamental re-examination of how to build trustworthy distributed network should be made.

Yuyu Sun, female, born in 1977,Lecturer, Ph.D. of Jilin University. She graduated from the Department of Computer Science and Technology of Jilin University in 2005,and obtained an MA degree. From 2008,she began to start her doctorate in computer in Jilin University, now she is working in Changchun University. Her current research interests include network and information security. She mainly engaged in Teaching and research on information security and Application software development. She has participated in one National Natural Science Foundation of China, one Major Project of Chinese National Programs for Fundamental Research and Development (973 Program), five Science and technology support key project plan of Jilin Provincial Science and technology Department , three S&T plan projects of Jilin Provincial Education Department .She has Wrote 4 textbooks as yet. She has published 14 academic articles in English and Chinese, four of that has been retrieved by EI.