Forward Secure Password-Based Authenticated Key Distribution in the Three-Party Setting*

Shuhua Wu and Yuefei Zhu

Department of Networks Engineering,
Zhengzhou Information Science Technology Institute,
Zhengzhou 450002, China
wushuhua726@sina.com.cn

Abstract. Key establishment protocols are used for distributing shared keying material in a secure manner. In 1995, Bellare and Rogaway presented a three-party server-based key distribution (3PKD) protocol. But the protocol was recently found insecure and then was fixed by Raymond Choo et al.. But forward-secrecy is not considered in the revised protocol. In this paper, we demonstrate that it is not forward secure indeed. We then revise the protocol to be a password-based authenticated key distribution in the three-party setting and prove our protocol is forward secure in the random-oracle and ideal-cipher models under the Password-based Chosen-basis Gap Diffie-Hellman assumption. Our protocol is quite simple and elegant, and rather efficient when compared to previous solutions.

Keywords: password, forward-secure, three-party.

1 Introduction

The need for authentication is obvious when two entities communicate on the Internet. The password-based mechanism is useful for user authentication in computer network systems. It allows users to be authenticated by remote computer systems via easily memorable passwords and in the absence of public-key infrastructures or pre-distributed symmetric keys. However, since people like to choose simply-guessed strings (e.g. personal identity, nickname, birth day, etc.) as their passwords, many password-based systems are vulnerable to replay attack or dictionary attacks [1]. Designing a secure password-based system is a precise task that has attracted many cryptographers. Bellovin and Merritt [1] proposed the encrypted key exchange (EKE) protocol in 1992. The EKE protocol enables two communication entities to authenticate each other and to establish a session key for securing later transmissions via a weak password. Since then, numerous two-party password-based authenticated key exchange (2PAKE) protocols have been proposed to improve security and performance. However, only a few take

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

J. Cao et al. (Eds.): NPC 2008, LNCS 5245, pp. 105–115, 2008.

into account the 3-party scenario, e.g., [2,3,4,5,6,7,8], where each communication entity shares a password with a trusted server and any two communication entities can be achieved mutual authentication and secure communication through the server's assistance. Moreover, to the best of our knowledge, with the exception of the protocols proposed in [6,7,8], none of the proposed the three-party password-based authenticated key exchange(3PAKE) enjoys provable security. However, the protocols in [7,8] were subsequently shown insecure in [9] and [10] respectively. As for the protocol proposed in [6], the security was proved in a model with no *Corrupt* oracle and thus the forward security for it was still unknown. Other protocols, such as the symmetric-key-based key distribution scheme of Bellare and Rogaway [11], do consider the 3-party setting, but not in the password-based scenario. Recently, the protocol [11] was found insecure and fixed by by Raymond Choo et al. in [12]. Yet, forward-secrecy is not considered in the revised protocol.

In this paper, we demonstrate that it is not forward secure indeed. We then revise the protocol to be a password-based authenticated key distribution in the three-party setting. One should remark that adding authentication services to a key establishment protocol is a not trivial since redundancy in the flows of the protocol can open the door to different forms of attacks [13]. Fortunately, we can prove our protocol is forward secure in the random-oracle [14] and ideal-cipher models [15] under the Password-based Chosen-basis Gap Diffie-Hellman assumption (see section 4). Our protocol is quite simple and elegant and rather efficient when compared to previous solutions. In particular, the costs for each participant of the new 3-party protocol are comparable to those of a 2-party password-based key exchange protocol. Besides, a three party password-based key distribution protocol is the underlying primitive of the generic construction in [6]. We hope one will leverage our work to obtain tighter and more meaningful security measurements for the forward-secrecy of the protocol.

The remainder of this paper is organized as follows. In Section 2, we introduce the formal model of security for for 3-party key exchange. Next, in Section 3, we recall the computational assumptions upon which the security of our protocol is based upon. Section 4 describes the 3PKD revised by Raymond Choo et al. and demonstrates that the revised protocol is not forward secure indeed. Section 5 then presents the improved protocol— our 3-party password-based key distribution protocol— along with its security claims and rigorous proof. In the last section, We conclude this paper.

2 Security Model for Three-Party Key Exchange

In this section, we introduce the formal security models which will be used in next section when we show that our protocol is secure in the random-oracle model. The model was proposed in 2000 by Bellare, Pointcheval and Rogaway [15], hereafter referred to as the BPR2000 model.

2.1 The Security Model

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 (see literature for more details [15,16].) The types of oracles available to the adversary are as follows:

- $Execute(U_1^{i_1}, S^j, U_2^{i_2})$: This query models passive attacks in which the attacker eavesdrops on honest executions among the client instances $U_1^{i_1}$ and $U_2^{i_2}$ and trusted server instance S^j . The output of this query consists of the messages that were exchanged during the honest execution of the protocol.
- $SendClient(U^i, m)$: This query models an active attack, in which the adversary may intercept a message and then modify it, create a new one, or simply forward it to the intended client. The output of this query is the message that client instance U^i would generate upon receipt of message m.
- $SendServer(S^j, m)$: This query models an active attack against a server. It outputs the message that server instance S^j would generate upon receipt of message m.
- $Reveal(U^i)$: If a session key is not defined for instance U^i or if a Test query was asked to either U^i or to its partner, then return $\perp$. Otherwise, return the session key held by the instance U^i.

2.2 Security Notions

In order to define a notion of security for the key exchange protocol, we consider a game in which the protocol $\mathcal{P}$ is executed in the presence of the adversary $\mathcal{A}$. In this game, we first choose the long-lived keys for each participant, provide coin tosses and oracles to $\mathcal{A}$, and then run the adversary, letting it ask any number of queries as described above, in any order.

Forward Security. In order to model the forward secrecy (semantic security) of the session key, we consider a game $Game^{ake-fs}(\mathcal{A}, \mathcal{P})$, in which two additional oracles are available to the adversary: the $Test(U^i)$ and $Corrupt(U)$: oracle.

- $Test(U^i)$: This query tries to capture the adversary's ability to tell apart a real session key from a random one. In order to answer it, we first flip a (private) coin b and then forward to the adversary either the session key sk held by U^i (i.e., the value that a query $Reveal(U^i)$ would output) if $b = 1$ or a random key of the same size if $b = 0$.
- $Corrupt(U)$: This query returns to the adversary the long-lived key (e.g. passwords pw_U in the password-based scenario) for participant U. As in [15], we assume the weak corruption model in which the internal states of all instances of that user are not returned to the adversary.

The $Test$-oracle can be queried at most once by the adversary $\mathcal{A}$ and is only available to $\mathcal{A}$ if the attacked instance U^i is FS-Fresh, which is defined to avoid cases in which adversary can trivially break the security of the scheme. In this setting, we say that a session key sk is FS-Fresh if all of the following hold:

(1) the instance holding sk has accepted, (2) no *Corrupt*-query has been asked since the beginning of the experiment; and (3) no *Reveal*-query has been asked to the instance holding sk or to its partner (defined according to the session identification). In other words, the adversary can only ask $Test$-queries to instances which had accepted before the *Corrupt* query is asked. Let **Succ** denote the event in which the adversary successfully guesses the hidden bit b used by $Test$ oracle. The FS-AKE advantage of an adversary $\mathcal{A}$ is then defined as $Adv_{\mathcal{P}}^{ake-fs}(\mathcal{A}) = 2Pr[\mathbf{Succ}] - 1$. The protocol $\mathcal{P}$ is said to be (t, ε)-FS-AKE-secure if $\mathcal{A}$'s advantage is smaller than ε for any adversary $\mathcal{A}$ running with time t. The definition of time-complexity that we use henceforth is the usual one, which includes the maximum of all execution times in the experiments defining the security plus the code size [17].

In the password-based scenario, key exchange 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 $O(n/|\mathcal{D}|) + \epsilon(k)$ where $|\mathcal{D}|$ is the size of the dictionary $\mathcal{D}$, n is the number of active sessions and $\epsilon(k)$ is a negligible function depending on the security parameter k.

Note 1. In the original security models, $\mathcal{A}$ was required to output the guess bit of b immediately after making a $Test$ query. However, such a requirement is not strong enough to guarantee security for certain applications(see section 4). Therefore, this restriction has been removed in the current models.

3 Algorithmic Assumptions

The arithmetic is in a finite cyclic group $G = \langle P \rangle$ of order a k-bit prime number q, where the operation is denoted addictively.

3.1 GDH-Assumption

A $(t, \varepsilon) - CDH_{P,G}$ attacker, in a finite cyclic group G of prime order q with P as a generator, is a probabilistic machine Δ running in time t such that its success probability $\mathbf{Succ}_{P,G}^{cdh}(\mathcal{A})$, given random elements xP and yP to output xyP, is greater than ε:

$$\mathbf{Succ}_{P,G}^{cdh}(\mathcal{A}) = Pr[\Delta(xP, yP) = xyP] \geq \varepsilon.$$

We denote by $\mathbf{Succ}_{P,G}^{cdh}(t)$ the maximal success probability over every adversaries running within time t. The CDH-Assumption states that $\mathbf{Succ}_{P,G}^{cdh}(t) \geq \varepsilon$ for any t/ε not too large.

A $(t, n, \varepsilon) - GDH_{P,G}$ attacker is a $(t, \varepsilon) - CDH_{P,G}$ attacker, with access to an additional oracle: a DDH-oracle, which on any input (xP, yP, zP) answers whether $z = xy \bmod q$. Its number of queries is limited to n. As usual, we denote by $\mathbf{Succ}_{P,G}^{gdh}(n, t)$ the maximal success probability over every adversaries running within time t. The GDH-Assumption states that $\mathbf{Succ}_{P,G}^{gdh}(n, t) \geq \varepsilon$ for any t/ε not too large [18].

3.2 PCGDH-Assumption

The so-called Password-based Chosen-basis CDH (PCCDH) problem is a variation of the computational Diffie-Hellman that is more appropriate to the password-based setting: Let $\mathcal{D} = \{1, \cdots, |\mathcal{D}|\}$ be a dictionary containing $|\mathcal{D}|$ equally likely password values. Now let us consider an adversary $\mathcal{A}$ that runs in two stages. In the first stage, the adversary is given as input two random elements U and V in G as well as the dictionary $\mathcal{D}$ and it outputs an element M in G (the chosen-basis). Next, we choose a password $pw \in \mathcal{D}$ randomly and give it to the adversary. The goal of the adversary in this second stage is to output $K = CDH(M+pwU, V)$. We denote by $\mathbf{Succ}^{pccdh}_{P,G,\mathcal{D}}(t)$ the maximal success probability over every adversaries $\mathcal{A}$ running within time t. An $(t, \varepsilon) - PCCDH_{P,G,\mathcal{D}}$ attacker is a probabilistic machine running in time t such that its success probability $\mathbf{Succ}^{pccdh}_{P,G,\mathcal{D}}(\mathcal{A})$ is greater than $1/|\mathcal{D}| + \varepsilon$. The PCCDH-Assumption states that $\mathbf{Succ}^{pccdh}_{P,G,\mathcal{D}}(t) \geq 1/|\mathcal{D}| + \varepsilon$ for any t/ε not too large. Fortunately, the new assumption is not stronger than the CDH-Assumption [19,20]. Similarly, we can define the PCGDH-Assumption.

4 Rmarks on Raymond Choo's protocol

In this section, we revisit Raymond Choo's protocol and demonstrate that the revised protocol is not forward secure indeed.

1. $A \longrightarrow B : R_A$
2. $B \longrightarrow S : R_A, R_B$
3a. $S \longrightarrow A : \{SK_{AB}\}_{K^{enc}_{AS}}, [A, B, R_A, R_B, \{SK_{AB}\}_{K^{enc}_{AS}}]_{K^{MAC}_{AS}}, R_B$
3b. $S \longrightarrow B : \{SK_{AB}\}_{K^{enc}_{BS}}, [A, B, R_A, R_B, \{SK_{AB}\}_{K^{enc}_{BS}}]_{K^{MAC}_{BS}}$

Fig. 1. An execution of Raymond Choo's protocol

As illustrated in Fig.1., Raymond Choo's protocol involves three parties, a trusted server S and two principals A and B who wish to establish communication. The security goal of this protocol is to distribute a session key between two communication principals (i.e. the key establishment goal), which is suitable for establishing a secure session. In the protocol, the notation $\{message\}_{K^{enc}_{AS}}$ denotes the encryption of some message under the encryption key K^{enc}_{AS} and the notation $[message]_{K^{MAC}_{AS}}$ denotes the computation of MAC digest of some message under the MAC key K^{MAC}_{AS}. K^{enc}_{AS} is the encryption key shared between A and B, and K^{MAC}_{AS} is the MAC key shared between A and B. Both keys, K^{enc}_{AS} and K^{MAC}_{AS}, are independent of each other.

The protocol begins by having A randomly select a k-bit challenge R_A and send it to the B with whom she desires to communicate. Upon receiving the message R_A from A, B also randomly selects a k-bit challenge R_B and sends R_B together with R_A as a message (R_A,R_B) to the server S. S, upon receiving the message (R_A,R_B)

from B, runs the session key generator to obtain a session key SK_{AB}, which has not been used before. S then encrypts SK_{AB} with K_{AS}^{enc} and K_{BS}^{enc} to obtain ciphertexts α_A and α_B, and computes the MAC digests β_A and β_B of the strings $(A, B, R_A, R_B, \{SK_{AB}\}_{K_{AS}^{enc}})$ and $(A, B, R_A, R_B, \{SK_{AB}\}_{K_{BS}^{enc}})$ under the keys K_{AS}^{MAC} and K_{BS}^{MAC} respectively. S then sends messages (α_A, β_A, R_B) and (α_B, β_B) to A and B respectively in Steps 3a and 3b of the protocol.

Unfortunately, forward-secrecy is not considered in the protocol. Indeed the revised protocol is not forward secure since any adversary who knows the long-lived encryption keys K_{AS}^{enc} or K_{BS}^{enc} certainly can obtain the session key by decrypting α_A and α_B respectively. Now we describe the attack in the BPR2000 mode and illustrate that it is wrong to make the restriction that the $Test$ query be the adversary's last. It is especially important to understand the security proof in Section 5.2. We assume a malicious adversary $\mathcal{A}$ runs the game simulation $Game$ as follows. As a preliminary step, $\mathcal{A}$ eavesdrops on honest executions among the client instances $U_1^{i_1}$ and $U_2^{i_2}$ and trusted server instance S^j and obtains the messages α_A, α_B. When the session is accepted, $\mathcal{A}$ makes a $Test$ oracle query to the client instance $U_1^{i_1}$ or $U_2^{i_2}$. We should note that the session is still fresh at this moment. $\mathcal{A}$ continues making a $Corrupt$ oracle query to the principal and knows its long-lived key K_{AS}^{enc} or K_{BS}^{enc} and thus the session key and the bit b involved in the $Test$ oracle. Eventually, $\mathcal{A}$ terminates the game simulation and outputs the value of b correctly. Our attack demonstrates that the protocol is not forward secure in the BPR2000 model. However, if $\mathcal{A}$ was required to output the guess bit of b immediately after making a $Test$ query, the attack described above would have not been captured. Therefore, removal of this restriction is quite important to guarantee security.

5 Our Three-Party Password-Based Protocol

As we mentioned in Section 1, the original key distribution scheme of Raymond Choo et al. [12] is not in the password-based scenario. In this section, we revised it to be a password-based authenticated key distribution protocol and provide the rigorous proof of forward-security for it based on the hardness of the Password-based Chosen-basis Gap Diffie-Hellman problem. The security proof is in the random oracle model and the ideal-cipher model. It assumes that the clients willing to establish a common secret session key share passwords with a common server and the latter is a trusted server.

5.1 Description

As illustrated on Fig.2. (with an honest execution of the 3PAKD protocol), the protocol runs between two clients A, B and a server S, and the session-key sk is a random value chosen by S and distributed to the clients. Client and server initially share a low-quality password PW, uniformly drawn from the dictionary $\mathcal{D}$. In Fig.2, by $U_2 \xleftarrow[send]{message} U_1$ we mean that user U_1 sends $message$ to user

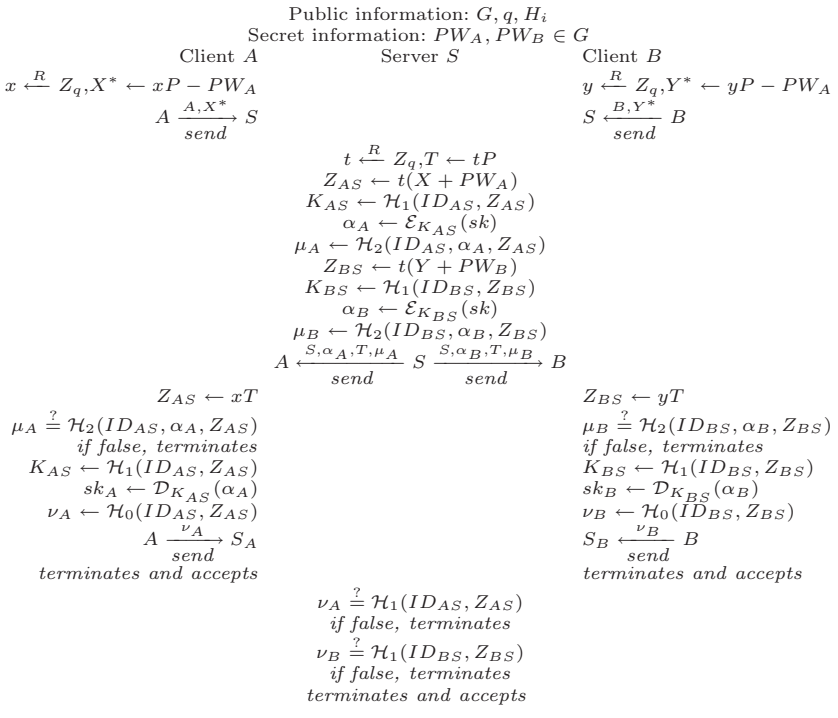

Fig. 2. The password-based authenticated key distribution

U_2. Hash functions from $\{0,1\}^*$ to $\{0,1\}^l$ are denoted $\mathcal{H}_i$ for $i = 0, 1, 2$. A block cipher is denoted $(\mathcal{E}_K, \mathcal{D}_K)$ where K is its private key.

The protocol consists of three flows. First, each client chooses an ephemeral public key by choosing a random element in Z_q and raising P to the that power, encrypts it using his password, and sends it to the server. Upon receiving a message from each client, the server decrypts these messages to recover each client's ephemeral public key, chooses a random index $t \in Z_q$, exponentiates each of the ephemeral public keys to the t-th power as the Diffie-Hellman keys Z, and at the same time raises P to the that power as his ephemeral public key. Then the server computes the private keys K for the block cipher via a hash function H_0 using as input ID and Z, and encrypts the session key sk to be distributed subsequently as the encrypted value α using the block cipher $\mathcal{E}$ with private key K. In the end, the server computes the authenticators μ via a hash function H_1 using as input ID, α and Z. Here, ID represents the string consisting of the transcript of the conversation among the clients and the server and the password. More specifically, ID_{AS} is A, B, S, PW_A, X^*, T and ID_{BS} is A, B, S, PW_B, Y^*, T. This is just for simplicity.

In the second round of messages, the server sends to each client his identity S, the encrypted values α, his ephemeral public key T and the authenticators μ. Upon receiving a message from the server, each client computes the

Diffie-Hellman key Z, and the authenticators μ. Then he checks the authenticator received is valid. If it is invalid, he simply abolishes and terminates the execution of the protocol. Otherwise, he proceeds to compute the private keys K for the block cipher and to recover the session key sk. In addition, he also computes his authenticator ν via a hash function $\mathcal{H}_2$.

In the third round of messages, the client sends his authenticator ν to the server S and accepts and terminates the execution of the protocol. Upon receiving the authenticator from the two clients, the server S checks the authenticators received —ν_A andν_B— are valid. If both of them are valid, accepts and terminates the execution of the protocol.

Note 2. One should remark that the last round of messages is necessarily included so that the servers can detect online dictionary attacks as pointed out in [21]. For 3-party PAKE protocols, only adding mutual authentication between two communicating clients in the end can not enhance those protocols to be resistant to undetectable on-line dictionary attacks. Unlike 2-party protocols, malicious attacker can play the legal role of client users and interacts with trusted servers to guess the value of passwords.

Our protocol is quite efficient, only requiring a small amount of computation by each user. In what concerns block cipher computations, hash computations, each client only has to perform 1 block cipher computation, and 3 hash computations; and the server only has to perform 2 block cipher computations, and 6 hash computations. All these can be done efficiently and their computational complexity can be neglected. The most expensive part of our protocol is the number of scalar multiplication, which entails the highest computational complexity. Since each client needs to perform 2 scalar multiplications and the server 3 scalar multiplications, our protocol has a per-user computational cost that is comparable to that of the underlying two-party encrypted key exchange. When compared to previous solution in [6], our protocol requires at least one less scalar multiplication for each participant and thus certainly more efficient.

5.2 Security

As the following theorem states, our 3PAKD is a forward-secure 3-party password-based key distribution protocol as long as the Password-based Chosen-basis Gap Diffie-Hellman problem is hard in G . The specification of this protocol is found on Fig.2.

Theorem 1. *Let $\mathcal{D}$ be a uniformly distributed dictionary of size $|\mathcal{D}|$. Let $\mathcal{P}$ describe the 3-party password-based authenticated key distribution protocol associated with these primitives as defined in Fig.2. Then,*

$$Adv_{\mathcal{P}}^{ake-fs}(\mathcal{A}) \leq \frac{(2q_p+q_s)^2}{q} + \frac{2q_{\mathcal{E}}^2}{q} + \frac{q_h^2}{2^l} + \frac{2q_s}{|\mathcal{D}|} + \frac{4q_s+2q_p}{2^l} + 4\boldsymbol{Succ}_{P,G,\mathcal{D}}^{pcgdh}(q_h, t+2\tau),$$

where q_s denotes the number of active interactions with the parties (Send-queries); q_p denotes the number of passive eavesdroppings (Execute-queries);

q_h denotes the number of hash queries to $\mathcal{H}_i$; $q_\mathcal{E}$ denotes the number of encryption/decryption queries; and τ denotes the computational time for an exponentiation in G.

Due to the limitation of the paper length, the complete proof is to be included in the full version of this paper.

Note 3. The ideal-cipher model is very strong (even stronger than the ideal-hash model) and yet there are natural and apparently-good ways to instantiate an ideal cipher for use in practical protocols (see [22]). Working in this model does not render trivial the goals that this paper is interested in, and it helps make for protocols that achieve provably forward security. We can only prove the proposed scheme is sematic secure but forward secure if we do not assume ideal cipher model. There seems to be some collisions with some technique that is used in our proof when we attempts to reduce an adversary against forward security of the protocol to an adversary against the classical security definition of the encryption scheme.

Rationale for the scheme. At first thought, you may wonder how we can make the original protocol forward-secure by adding password-authentication services. Now let us reconsider the attack in the section 4. In that case, the adversary $\mathcal{A}$ that eavesdrops on honest executions and then corrupts any player of the target session can compute the ephemeral public keys but should not be able to compute the Diffie-Hellman key and thus the private key and the session key. Therefore,we can prove our protocol is forward-secure in the BRP2000 model.

6 Conclusion

We have shown Raymond Choo's protocol is not forward-secure in the BPR2000 model. Following that, we have presented a 3-party password-based authenticated key distribution protocol by adding password-authenti- cation services to Raymond Choo's protocol. Furthermore, we have proved the forward-security for our protocol under the Password-based Chosen-basis Gap Diffie-Hellman assumption in the BPR2000 model. When compared with previous solutions in the password-based scenario, our protocol is efficient. The costs for each participant of the new 3-party protocol are comparable to those of a 2-party encrypted key exchange protocol.

References

1. Bellovin, S.M., Merritt, M.: Encrypted key exchange: password-based protocols secure against dictionary attacks. In: Proc. 1992 IEEE Computer Society Symp. on Research in security and Privacy, May 1992, pp. 72–84 (1992)
2. Byun, J.W., Jeong, I.R., Lee, D.H., Park, C.-S.: Password-authenticated key exchange between clients with different passwords. In: Deng, R.H., Qing, S., Bao, F., Zhou, J. (eds.) ICICS 2002. LNCS, vol. 2513, pp. 134–146. Springer, Heidelberg (2002)

3. Lin, C.-L., Sun, H.-M., Hwang, T.: Three-party encrypted key exchange: Attacks and a solution. ACM SIGOPS Operating Systems Review 34(4), 12–20 (2000)
4. Wang, S., Wang, J., Xu, M.: Weaknesses of a password-authenticated key exchange protocol between clients with different passwords. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 414–425. Springer, Heidelberg (2004)
5. Yeh, H.-T., Sun, H.-M., Hwang, T.: Efficient three-party authentication and key agreement protocols resistant to password guessing attacks. Journal of Information Science and Engineering 19(6), 1059–1070 (2003)
6. Abdalla, M., Fouque, P.-A., Pointcheval, D.: Password-based authenticated key exchange in the three-party setting. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 65–84. Springer, Heidelberg (2005)
7. Abdalla, M., Pointcheval, D.: Interactive Diffie-Hellman Assumptions with Applications to Password-based Authentication. In: Patrick, S., Yung, A. (eds.) FC 2005. LNCS, vol. 3570, pp. 341–356. Springer, Heidelberg (2005), http://www.di.ens.fr/~pointche/pub.php
8. Wen, H.-A., Lee, T.-F., Hwang, T.: Provably secure three-party password-based authenticated key exchange protocol using Weil pairing. IEE Proceedings — Communications 152(2), 138–143 (2005)
9. Nam, J., Kim, S., Won, D.: Security Weakness in a Three-Party Password-Based Key Exchange Protocol Using Weil Pairing. In: Cryptology ePrint Archive, Report (2005), http://eprint.iacr.org/2005/269.ps
10. Choo, K.-K.R., Boyd, C., Hitchcock, Y.: Examining Indistinguishability-Based Proof Models for Key Establishment Protocols. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 585–604. Springer, Heidelberg (2005)
11. Bellare, M., Rogaway, P.: Provably Secure Session Key Distribution: The Three Party Case. In: 27th ACM Symposium on the Theory of Computing, pp. 57–66. ACM Press, New York (1995)
12. Choo, K.-K.R., Boyd, C., Hitchcock, Y., Maitland, G.: On Session Identifiers in Provably Secure Protocols—The Bellare-Rogaway Three-Party Key Distribution Protocol Revisited. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 352–367. Springer, Heidelberg (2005)
13. Abdalla, M., Bresson, E., Chevassut, O., Pointcheval, D.: Password-based Group Key Exchange in a Constant Number of Rounds. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 427–442. Springer, Heidelberg (2006)
14. Bellare, M., Rogaway, P.: Optimal asymmetric encryption: How to encrypt with RSA. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995), http://www-cse.ucsd.edu/users/mihir
15. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure Against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
16. Bresson, E., Chevassut, O., Pointcheval, D.: New security results on encrypted key exchange. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 145–158. Springer, Heidelberg (2004)
17. Abdalla, M., Bellare, M., Rogaway, P.: The oracle Diffie-Hellman assumptions and an analysis of DHIES. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 143–158. Springer, Heidelberg (2001)
18. Okamoto, T., Pointcheval, D.: The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992. Springer, Heidelberg (2001)

19. Abdalla, M., Pointcheval, D.: Simple Password-Based Encrypted Key Exchange Protocols. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 191–208. Springer, Heidelberg (2005)
20. Abdalla, M., Bresson, E., Chevassut, O., Möller, B., Pointcheval, D.: Provably Secure Password-Based Authentication in TLS. In: Proc. of at AsiaCCS 2006, Taipei, Taiwan, March 21-24 (2006)
21. Ding, Y., Horster, P.: Undetectable On-line Password Guessing Attacks. ACM Operating Systems Review 29(4), 77–86 (1995)
22. Black, J., Rogaway, P.: Ciphers with Arbitrary Finite Domains (manuscript, 2000)