Enhancements of a Three-Party Password-Based Authenticated Key Exchange Protocol

Shuhua Wu[1], Kefei Chen[2,3], and Yuefei Zhu[1]
[1]Department of Network Engineering, Information Engineering University, China
[2]Department of Computer Science and Engineering, Shanghai Jiaotong University, China
[3]Institute of China Electronic System Engineering Corporation, China

Abstract: *This paper discusses the security for a simple and efficient three-party password-based authenticated key exchange protocol proposed by Huang most recently. Our analysis shows her protocol is still vulnerable to three kinds of attacks: 1). undetectable on-line dictionary attacks, 2). key-compromise impersonation attack. Thereafter we propose an enhanced protocol that can defeat the attacks described and yet is reasonably efficient.*

Keywords: *Password-based, authenticated key exchange, three-party, dictionary attack.*

Received June 2, 2010; accepted March 1, 2011; published online March 1, 2012

1. Introduction

To communicate securely over an insecure public network, it is essential that secret keys are exchanged securely. Password-Authenticated Key Exchange (PAKE) protocol allows two parties holding a same memorable password to agree on a common secret value (a session key) over an insecure open network. It also seems more convenient since human-memorable passwords are simpler to use than, for example, having additional cryptographic devices capable of storing high-entropy secret keys.

The intrinsic problem with password-based protocols is that the memorable password, associated with each user, has low entropy, so that it is not easy to protect the password information against so-called dictionary attacks. Generally, we can divide such attacks into the following three classes [18]:

- *Off-Line Dictionary Attacks:* Only by using the eavesdropped information, an attacker guesses a password and verifies its guess in an off-line manner. No participation of the honest client or the server is required, so these attacks can not be noticed.
- *Undetectable On-Line Dictionary Attacks:* An attacker tries to verify a password guess in an on-line transaction. However, a failed guess can not be detected by the honest client or the server, since one of them is not able to distinguish a malicious request from an honest one.
- *Detectable On-Line Dictionary Attacks:* Similar to above, an attacker attempts to use a guessed password in an on-line transaction. Using the response from the honest client or the server, it verifies the correctness of its guess. But a failed guess can be detected by the honest client or the server.

Among these attacks, detectable on-line dictionary attacks are unavoidable and should be handled by taking additional precautions such as logging failed protocol attempts and invalidating the use of the password after a certain number of failures. However, both off-line and undetectable on-line dictionary attacks are serious attacks against password-based settings so that a secure password-based protocol should ideally resist the two types of attacks.

1.1. Related Works

The first PAKE protocol, known as Encrypted Key Exchange (EKE), which was suggested by Bellovin and Merritt [5]. Subsequently, many other two-party PAKE protocols have been proposed (e.g., [1, 3, 7, 14, 20]). Because two-party PAKE protocols are not suitable for the large peer-to-peer architecture (e.g., [11]), many researchers have recently begun to study the Three-Party PAKE (3PAKE) protocols (e.g., [2, 15, 16, 17, 18]), in which a Trusted Server *TS* exists to mediate between two communication parties to allow mutual authentication and each user only needs to share one password with the common server. Unfortunately, some of them are not efficient enough to be used in practice (e.g., [2, 17]), the others are not secure (e.g., [15, 16, 18]). Later, two efficient three-party password-based key exchange protocols were proposed by Abdalla and Pointcheval [4] and Lu and Cao [19] respectively. However, the two schemes were still found insecure against undetectable on-line dictionary attacks or off-line dictionary attacks in [8, 9, 10, 13, 21, 22] respectively. Problems in them

illustrates that the design of such protocols remains a hard problem despite years of research.

1.2. Contribution

Most recently, to the best of our knowledge, Huang [12] also proposed a simple 3PAKE protocol, which is more efficient than previously proposed schemes. She claimed that her protocol could resist against various dictionary attacks and was suitable for some practical scenarios. Unfortunately, we find that some security weaknesses still remain in her protocol. In this paper, we first show her protocol is still vulnerable to kinds of attacks:

1. Undetectable on-line dictionary attacks.
2. Key-compromise impersonation attack.

Thereafter we propose an enhanced protocol that can defeat the attacks described and yet is reasonably efficient. Please note, independently of and concurrently to our work, Yoon and Yoo [24] also found the security weaknesses of Huang's protocol. However, they did not suggest any countermeasures.

1.3. Organization

The remainder of this paper is organized as follows: Section 2 briefly reviews Huang's three-party password-based authenticated protocol. Section 3 then reveals three weaknesses existing in Huang's protocol. Section 4 presents an enhanced 3PAKE protocol along with its and security analysis and performance analysis. Finally, conclusions is presented in section 5.

2. Review of Huang's Protocol

This section describes the 3PAKE protocol proposed by Huang [12], starting with some notations.

2.1. Notations

The notations used in their protocol are described as in the following:

- *A, B*: Two identity of clients (users).
- *TS*: A *TS* (remote server).
- pw_A(pw_B): The password shared between user *A* (resp. *B*) and *TS*.
- *p*: A large prime number such that *p-1* has a large prime factor *q* ($q \geq 2^{256}$).
- *g*: A generator with order *q* in *GF* (*p*).
- G: The cyclic group generated by *g*.
- $\oplus$: An exclusive-or operator.
- *h*(): A public one-way hash function.

2.2. Protocol Description

There are three entities involved in the protocol: the authentication server *TS*, and two users *A* (initiator) and *B* (responder) who wish to establish a session key between them. Each user's password is assumed to be shared with the server *TS* via a secure channel. As illustrated on Figure 1, *A* and *B* authenticate each other with *TS*'s help, then *A* and *B* can share a common session key *K*. The details will be described in the following steps. Here, we just follow the description in [12].

- *Step 1.* User *A* chooses a random number *x* and computes $R_A=(g^x \bmod p)\oplus h(pw_A, A, B)$, then sends (A, R_A) to user *B*.
- *Step 2.* User *B* also selects a random number *y* and computes $R_B = (g^y \bmod p) \oplus h(pw_B, A, B)$, then forwards (A, R_A, B, R_B) to *TS*.
- *Step 3.* Upon receiving (A, R_A, B, R_B), the *TS* first uses pw_A and pw_B to compute $g^x=R_A\oplus h(pw_A, A, B)$, $g^y = R_B \oplus h(pw_B, A, B)$ respectively. Then, *TS* chooses another random number *z* and computes $a = g^{xz} \bmod p$, $b = g^{yz} \bmod p$. Finally, *TS* send (Z_A, Z_B) to user *B*, where $Z_A=b\oplus h(pw_A, g^x)$ and $Z_B=a\oplus h(pw_B, g^y)$.
- *Step 4.* When *B* receives (Z_A, Z_B), it uses its password pw_B and g^y to obtain $a=Z_B\oplus h(pw_B, g^y)$, and uses the random number *y* to compute the common session key $K=a^y=(g^{xz})^y=g^{xyz} \bmod p$ and $S_B=h(K, B)$. Next, user *B* forwards (Z_A, S_B) to user *A*.
- *Step 5.* After receiving (Z_A, S_B), user *A* also uses its password pw_A and g^x to derive $b=Z_A\oplus h(pw_A, g^x)$, and uses the random number *x* to obtain the common key $K=b^y=(g^{yz})^x=g^{xyz} \bmod p$. Then, *A* checks whether $S_B=h(K, B)$ holds or not. If it does not hold, *A* terminates the protocol. Otherwise, *A* is convinced that $K=g^{xyz}$ is a valid session key. Then, *A* computes $S_A=h(K, A)$ and sends it to user *B*.
- *Step 6.* Upon receiving S_A, user *B* verifies whether $S_A=h(K, A)$ holds or not. If it does not hold, *B* terminates the protocol. Otherwise, *K* is a valid session key. Both the users *A* and *B* can use this session key *K* for secure communication. Here, *K* is only used for one session.

3. Weaknesses of Huang's Protocol

In this section, we will show that, unfortunately, Huang's protocol [12] is vulnerable to an undetectable on-line dictionary attack. In addition, we also shows that her protocol can not offer resilience against key-compromise impersonation.

It is worth mentioning that, Yoon and Yoo [24] also found Huang's protocol is not secure to undetectable on-line dictionary attacks. Besides, they also provide two simple and efficient off-line password guessing attacks on Huang's protocol. More information about it is referred to [24].

3.1. Undetectable On-line Dictionary Attacks

In this subsection, we demonstrate Huang's protocol [12], falls to an undetectable on-line password guessing attack, by which an adversary is able to legally gain information about the password by repeatedly and indiscernibly asking queries to the authentication server.

In the following, we show any adversary $\mathcal{A}$ can mount an undetectable online dictionary attack even without knowing any password. The attack scenario is outlined in Figure 2. A more detailed description of the attack is as follows:

1. The adversary $\mathcal{A}$ first chooses a random number x' and guesses two password pw_A^* and pw_B^*. Then $\mathcal{A}$ computes $X=g^{x'}$ and simply sets $R_B=X\oplus h(pw_B^*, A, B)$ and $R_A=X\oplus h(pw_A^*, A, B)$. Finally, $\mathcal{A}$ sends (A, R_A, B, R_B) to TS. For simplicity, we denote $R_A\oplus h(pw_A, A, B)$ and $R_B\oplus h(pw_B, A, B)$ by α and β respectively. Therefore, if the guessed passwords pw_A^* and pw_B^* are the correct passwords of A and B respectively, both the α value and the β value will be identical with the X value.
2. Upon receiving (A, R_A, B, R_B), the TS first uses pw_A and pw_B to compute $\alpha=R_A\oplus h(pw_A, A, B)$, $\beta=R_B\oplus h(pw_B, A, B)$ respectively. Then, TS chooses another random number z and computes $a=\alpha^z \bmod p$ and $b=\beta^z \bmod p$. Finally, TS send (Z_A, Z_B) to user B, where $Z_A=b\oplus h(pw_A, \alpha)$ and $Z_B=a\oplus h(pw_B, \beta)$.
3. $\mathcal{A}$ intercepts the message (Z_A, Z_B) and uses X, pw_A^* and pw_B^* to obtain $a'=Z_B\oplus h(pw_B^*, X)$ and $b'=Z_A\oplus h(pw_A^*, X)$. Then it verifies whether the computed b' value and a' value are identical or not. As mentioned before, if both pw_A^* and pw_B^* are the correct passwords of A and B respectively, the α value and the β value will be identical with the X value. Accordingly, the examination of whether the computed b' value and a' value are identical will be successfully verified since $a'=a=(\alpha^z \bmod p)=(X^z \bmod p)=(\beta^z \bmod p)=b=b'$ occurs in that case. In brief, if the computed b' value and a' value are equivalent, it implies that $\mathcal{A}$ does guess the correct passwords of user A and B respectively. Otherwise, $\mathcal{A}$ repeatedly performs the steps 1, 2, and 3 while TS never detects a failure of $\mathcal{A}$'s malicious trial. Finally, once the examination is successfully verified, $\mathcal{A}$ believes that it actually guesses the correct passwords of user A and B.

Obviously, the above attack scenario shows that Hwang's scheme cannot prevent the secret password from being known even by an outsider attacker while user A and B need not be present at all. One can easily remark that if the adversary is an insider attacker, say A, he can similarly guess B's password more efficiently. A more detailed description of the attack is as follows:

1. As a preliminary step, the adversary records the first message (A, R_A) sent from A to B during an honest execution of the protocol. Later, B indeed mounts an attack by initiating a new session.
2. B first guesses a password pw_A^*, and then computes $X=R_A\oplus h(pw_A^*, A, B)$ for an unknown number in Z_q^*. Then he simply sets $R_B=X\oplus h(pw_B, A, B)$. Finally, B sends (A, R_A, B, R_B) to TS. For simplicity, we denote $R_A\oplus h(pw_A, A, B)$ and $R_B\oplus h(pw_B, A, B)$ by α and β respectively. Therefore, if the guessed password pw_A^* is A's correct password, both the α value and β value will be identical with the X value. Here we assume both α and β fall into $GF(p)$ with high probability, otherwise one can mount a partition attack as described previously.
3. Upon receiving (A, R_A, B, R_B), TS operates as specified in step 3 of the protocol. The TS first uses pw_A and pw_B to compute $\alpha=R_A\oplus h(pw_A, A, B)$ and $\beta=R_B\oplus h(pw_B, A, B)$, respectively. Then, TS chooses another random number z and computes $a=\alpha^z \bmod p$, $b=\beta^z \bmod p$. Finally, TS sends (Z_A, Z_B) to user B, where $Z_A=b\oplus h(pw_A, \alpha)$ and $Z_B=a\oplus h(pw_B, \beta)$.
4. When B receives (Z_A, Z_B), it uses its password pw_B and X to obtain $a'=Z_B\oplus h(pw_B, X)$, and on the other hand, it also uses pw_A^* and X to compute $b'=Z_A\oplus h(pw_A^*, X)$. Then it verifies whether computed b' value and a' value are identical or not. As mentioned before, if the pw_A^* value is identical to the pw_A value, both the α value and β value will be identical with the X value. Accordingly, the examination of whether the computed b' value and a' value are identical will be successfully verified since $b'=a'=(X^z \bmod p)$ occurs in that case. In brief, if the computed b' value and a' value are equivalent, it implies that B does guess the correct password of User A. Otherwise, B repeatedly performs the steps 2, 3 and 4 while TS never detects a failure of B's malicious trial. Finally, once the examination is successfully verified, B believes that it actually guesses the correct password of user A.

In our attacks, an adversary tries to break the security of a scheme by a brute-force method, i.e., it tries online all possible combinations of secret keys in a given small set of values while TS can not detect any malicious trial at all. Even though these attacks are not very effective in the case of high-entropy keys, they can be very damaging when the secret key is a password since the attacker has a non-negligible chance of winning [3]. As a result of the two attacks, the authentication mechanism of the protocol is completely compromised because the adversary can impersonate that user with the knowledge of its

password and thus nothing is guaranteed for future sessions. We will explore effective countermeasures in next section.

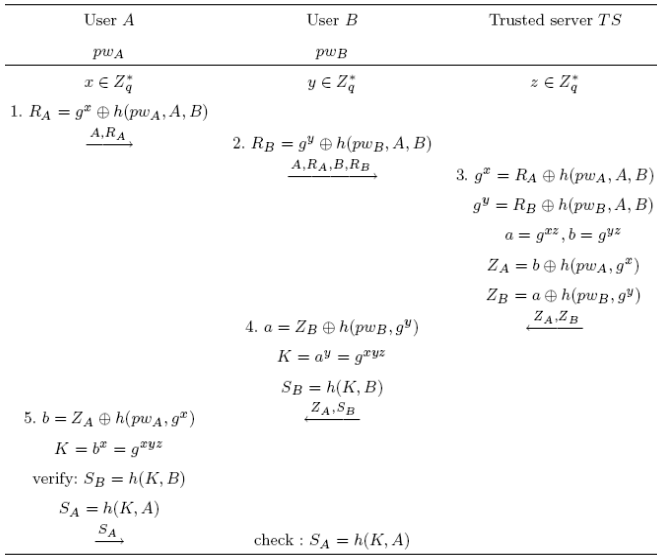

Figure 1. Huang's protocol.

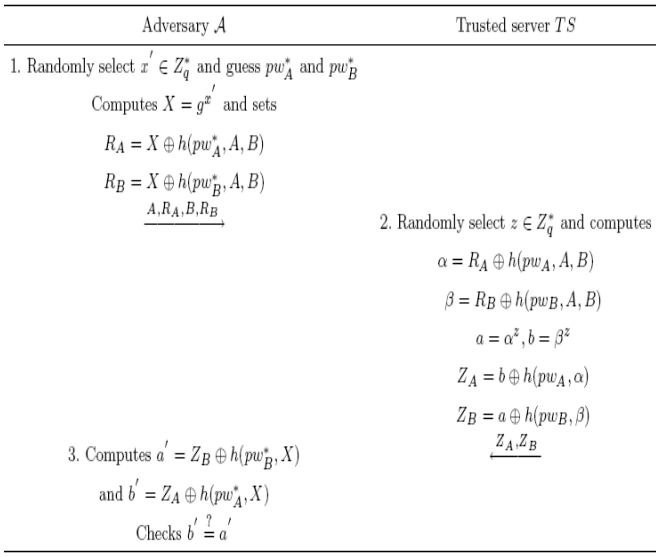

Figure 2. Undetectable on-line password guessing attack on Huang's protocol.

3.2. Key-Compromise Impersonation Attack

When the long-term key of a communicating entity is compromised, the adversary will be able to masquerade as the entity but the situation will be even worse if the adversary can also masqueraded as another communicating entity. We say a protocol offers resilience against key-compromise impersonation if it can prevent this attack. This security property is a desirable one that any given key exchange protocol is expected to possess [6]. One can easily remark that, in Huang's scheme, if A's password pw_A is compromised (this assumption is reasonable because the users may be careless or unskilled at protecting their passwords), the adversary $\mathcal{A}$ can not only masquerade as A to interact with B but also he can masquerade as B to interact with A successfully. More specifically, upon intercepting (A, R_A) sent by A in step 1, $\mathcal{A}$ uses pw_A to compute $g^x = R_A \oplus h(pw_A, A, B)$, selects a random number y and computes $Z_A = (g^y \bmod p) \oplus h(pw_A, g^x)$. Then he computes $K=(g^x)^y \bmod p$ and $S_B=h(K,B)$ and sends (Z_A, S_B) to user A as if it was originated from B. As a result, A will think $\mathcal{A}$ is B while the latter is never present at all. In other words, her scheme can not provide resilience against key-compromise impersonation. The weakness would be very damaging in some scenario, say, where B is a boss and A is his assistant. Then $\mathcal{A}$ would be able to masquerade as the boss to give commands to his assistant.

4. Enhanced Protocol

In this section, we present an enhanced protocol, which is based on the two-party protocol in [7] and then make some analysis on its performance.

4.1. Description

First, we define some notations used in our scheme. Let G be the cyclic group generated by g with large prime order q; and $\mathcal{H}: \{0,1\}^* \rightarrow \{0,1\}^l$ a secure hash function, where l is a security parameter. In our protocol, we assume the two users (or clients) A and B willing to establish a common secret session key share passwords $PW_A, PW_B \in G$ respectively with a common server TS. We also assume that the TS has a private/public key pair: (S, Q) with $Q=g^s$. The public parameters $(\mathcal{H}, q, g, G, Q)$ have been fixed in advance and are known to all parties in the network. The simplified description of the new protocol is given in Figure 3, where . denotes concatenation of bit strings. The details will be described in the following:

- *Step 1.* User A chooses two random numbers $r_A, w_A \in Z_q^*$ and computes $R_A = g^{r_A} \oplus PW_A$, $T_A = Q^{r_A}$ and $W_A = g^{w_A}$. Then it compute $\tau_{A,S} = \mathcal{H}(A \| B \| TS \| R_A \| W_A \| PW_A \| T_A)$ and sends $(A, R_A, W_A, \tau_{A,S})$ to user B.
- *Step 2.* Upon receiving $(A, R_A, W_A, \tau_{A,S})$, User B also selects two random numbers $r_B, w_B \in Z_q^*$ and computes $R_B = g^{r_B} \oplus PW_B$, $T_B = Q^{r_B}$ and $W_B = g^{w_B}$. Then it computes $\tau_{B,S} = \mathcal{H}(B \| TS \| A \| R_B \| W_B \| PW_B \| T_B)$, then forwards $(A, R_A, W_A, \tau_{A,S}, B, R_B, W_B, \tau_{B,S})$ to TS.
- *Step 3.* Upon receiving $(A, R_A, W_A, \tau_{A,S}, B, R_B, W_B, \tau_{B,S})$ the TS first computes $T_A = (R_A \oplus PW_A)^s$ and $T_B = (R_B \oplus PW_B)^s$ (should not be 0 or 1) and then uses them to check $\tau_{A,S}$ and $\tau_{B,S}$ respectively in a straight way. TS detects failure of a malicious trial

and thus terminates if either of them is valid or moves to the next phase otherwise. Finally, *TS* computes $\tau_{S,A} = \mathcal{H}(TS \| A \| B \| R_A \| W_B \| PW_A \| T_A)$ and $\tau_{S,B} = \mathcal{H}(TS \| B \| A \| R_B \| W_A \| PW_B \| T_B)$ and then sends $(\tau_{S,A}, \tau_{S,B})$ to user *B*.

- *Step 4.* Upon receiving $(\tau_{S,A}, \tau_{S,B})$, *B* first checks the validity of $\tau_{S,B}$ using T_B. If that value is invalid, *B* detects failure of a malicious trial and thus terminates the protocol. Otherwise, it computes $K = W_A^{w_B}$, and uses this value to compute the common session key $SK = \mathcal{H}(A \| B \| W_A \| W_B \| K)$ and $S_B = \mathcal{H}(SK \| B)$. Next, user *B* forwards $(W_B, \tau_{S,A}, S_B)$ to user *A*.
- *Step 5.* After receiving $(W_B, \tau_{S,A}, S_B)$, user *A* first checks the validity of $\tau_{S,A}$ using T_A. If that value is invalid, *A* detects failure of a malicious trial and thus terminates the protocol. Otherwise, it computes $K = W_B^{w_A}$, and uses this value to obtain the common key $SK = \mathcal{H}(A \| B \| W_A \| W_B \| K)$. Then, *A* also checks whether $S_B = \mathcal{H}(SK \| B)$ holds or not. If it does not hold, *A* terminates the protocol. Otherwise, *A* is convinced that *SK* is a valid session key and accepts it. Then, *A* computes $S_A = \mathcal{H}(SK \| A)$ and sends it to user *B*.
- *Step 6.* Upon receiving S_A, user *B* verifies whether $S_A = \mathcal{H}(SK \| A)$ holds or not. If it does not hold, *B* terminates the protocol. Otherwise, *SK* is a valid session key. Both the users *A* and *B* can use this session key *SK* for secure communication. Here, *SK* is only used for one session.
- *Note 1.* Now the relation for helping to guess the password in section 3.1 is not available to the intruder since each message is sent along with its authenticator so that any failure of a malicious trial can be detected. And one principal will invalidate or block the use of a password whenever a certain number of failed attempts occur. Therefore, the attack in Figure 3 is not possible either.
- *Note 2.* Although *Q* is *TS* 's public key, it can be "hard-coded" into any implementation of the protocol on the client side as other public parameters, say *g*. We can do so just because all the users resort to a single common server *TS* for authenticating each other in the three-party PAKE protocols. Therefore, our protocol still provides password-only authentication but does not require complex Public-Key Infrastructure (PKI).

The correctness of our protocol follows from the fact that, in an honest execution of the protocol, $T_A = Q^{r_A} = R_A^s$, $T_B = Q^{r_B} = R_B^s$ and $K = W_B^{r_A} = W_A^{r_B}$.

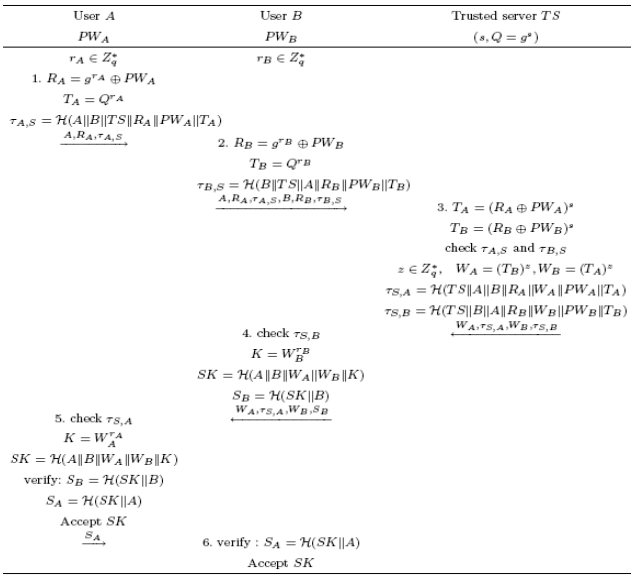

Figure 3. Our enhanced protocol.

How the Password Becomes an element in G. Since the password PW appears as an element of G in the computations for our 3PAKE, some additional function is needed to obtain this element from the password string password, as in [1]. In the protocol description, we do not care about details of the function and simply use the result PW (in group G) as the "effective password" instead: anyone knowing PW is actually able to impersonate the client or the server, and attacking the protocol reduces to finding PW. In other words, at the protocol level, PW is the password needed for authentication and password is just a way to remember it.

4.2. Security

The security of our protocol mainly relies on the Computational Diffie-Hellman (CDH) assumption: given $X=g^x$ and $Y = g^y$, where x and y are drawn randomly from Z_q^*, it is computationally infeasible to compute g^{xy} (denoted by $CDH_{g,G}(X,Y)$). Furthermore, we further assume its decision problem, i.e., DDH, is also hard.

Now we provide the intuitive understanding the security for our protocol. One can easily get the following facts:

1. The adversary can not successfully impersonate *TS* to a user, say *A*, because he can not compute $CDH_{g,G}(R_A \oplus PW_A, Q)$ when R_A generated by *A* based on the hardness of CDH problem and thus he can not reply with a valid authenticator $\tau_{S,A}$ for *A*'s challenge R_A.
2. Similarly, the adversary can not impersonate a user, say *A*, to send *TS* a new valid message $(R_A, \tau_{A,S})$ unless he guessed *A*'s correct password, based on CDH assumption.

3. The adversary may impersonate a user, say A, to replay to TS such a message $(R_A, \tau_{A,S})$ previously generated by A but he can not successfully establish a session key with B because he can not know $CDH_{g,G}(W_A, W_B)$ when W_A and W_B generated by TS based on the hardness of CDH problem and thus can not reply with a valid authenticator S_A to B.

According to facts 1 and 2 each party in our scheme can naturally detect failure of a malicious trial. On the other hand, one principal will invalidate or block the use of a password whenever a certain number of failed attempts occur. Therefore, our scheme can resist password guessing attacks. Based on facts 1, 2 and 3, one can easily remark that the attacker can not impersonate B to A even with the knowledge of the latter's password. That is, our scheme can resist against key-compromise impersonation attack. Based on the above analysis, an adversary should not be able to establish any session keys with honest legal users. For the session keys established between any two honest legal users, the attacker can not retrieve any information about them because he can not know $CDH_{g,G}(W_A, W_B)$ based on CDH assumption.

4.3. Performance

Our protocol is reasonably efficient. The efficiency is measured by the following two aspects:

- *Communication Cost:* The number of communication steps during the execution of protocol.
- *Computation Cost:* The computation complexity of a participant.

In what concerns Computation Cost, we only count the number of exponentiation, which entails the highest computational complexity, and neglect the computational complexity of all other operations such as Hash computation, which can be done efficiently. The details of comparisons between our protocol and the previously proposed efficient schemes so far as I know are shown in Table 1.

Table 1. Comparisons with related works.

Schemes	Computation Cost*		Communication Cost
	User	Server	
Abadi et al.'s Scheme[2]	4	4	10
Wang and Hu's Scheme[23]	4	4	8
Kim and Choi's Scheme[13]	5	6	5
Guo et al.'s Scheme[10]	6	10	5
Chung and Ku's Scheme[9]	4	6	5
Huang's Scheme[12]	2	2	5
Our Scheme	3	4	5

*Note Abadi *et al.'s* scheme in [2] and Wang and Hu's scheme in [23] are assumed to have been appropriately instantiated with efficient components recommended respectively.

As shown in Table 1, in one run of the enhanced protocol, each user performs three exponentiations and the TS performs four exponentiations. Although our protocol is not better than Huang's protocol, it is still more efficient than other schemes in [2, 9, 10, 13, 23]. However, Huang's protocol did not guarantee resistance against undetectable online dictionary attack or offer key-compromise impersonation resistance at all. In contrast with all these previous solutions, our scheme can resist both types of attacks. Given the better security guarantees, the performance of our scheme may be considered quite reasonable.

5. Conclusions

In this paper, we have demonstrated that Huang's three-party password-based authenticated protocol is still vulnerable to three kinds of attacks: 1). undetectable on-line dictionary attacks, and 2). key-compromise impersonation attack. Thereafter we have proposed an enhanced protocol that can defeat the attacks described and yet is reasonably efficient.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (No. 61101112) and China Postdoctoral Science Foundation (No. 2011M500775). The authors are grateful to the anonymous reviewers for valuable comments.

References

[1] Abdalla M., Bresson E., Chevassut O., Mőller B., and Pointcheval D., "Provably Secure Password-Based Authentication in TLS," *in Proceedings of the 1st ACM Symposium on Information, Computer and Communications Security*, USA, pp. 35-45, 2006.

[2] Abdalla M., Fouque P., and Pointcheval D., "Password-Based Authenticated Key Exchange in the Three-Party Setting," *in Proceedings of IEEE Information Security*, vol. 153, pp. 27-39, 2006,

[3] Abdalla M. and Pointcheval D., "Simple Password-Based Encrypted Key Exchange Protocols," *in Proceedings of the International Conference on Topics in Cryptology*, USA, vol. 3376, pp. 191-208, 2005.

[4] Abdalla M. and Pointcheval D., "Interactive Diffie-Hellman Assumptions with Applications to Password-Based Authentication," *in Proceedings of the 9th International Conference on Financial Cryptography*, Berlin, pp. 341-356, 2005.

[5] Bellovin S. and Merritt M., "Encrypted key Exchange: Password-Based Protocols Secure Against Dictionary Attacks," *in Proceedings of*

IEEE Symposium on Security and Privacy, USA, pp. 72-84, 1992.
[6] Boyd C. and Mathuria A., *Protocols for Authentication and Key Establishment*, Springer-Verlag, 2003.
[7] Bresson E., Chevassut O., and Pointcheval D., "New Security Results on Encrypted Key Exchange," *in Proceedings of Public Key Cryptography*, Berlin, pp. 145-158, 2004.
[8] Choo K., Boyd C., and Hitchcock Y., "Examining Indistinguishability-Based Proof Models for Key Establishment Protocols," *in Proceedings of the 11th International Conference on Theory and Application of Cryptology and Information Security*, Heidelberg, pp. 585-604, 2005.
[9] Chung H. and Ku W., "Three Weaknesses in a Simple Three-Party Key Exchange Protocol," *Information Science*, vol. 178, no. 1, pp. 220-229, 2008.
[10] Guo H., Li Z., Mu Y., and Zhang X., "Cryptanalysis of Simple Three-Party Key Exchange Protocol," *Computers and Security*, vol. 27, no. 1-2, pp. 16-21, 2008.
[11] Hassan M. and Abdullah A., "A New Grid Resource Discovery Framework," *The International Arab Journal of Information Technology*, vol. 8, no. 1, pp. 99-107, 2011.
[12] Huang H., "A Simple Three-Party Password-Based Key Exchange Protocol," *International Journal of Communications and Systems*, vol. 22, no. 7, pp. 857-862, 2009.
[13] Kim H. and Choi J., "Enhanced Password-Based Simple Three-Party Key Exchange Protocol," *Computers and Electrical Engineering*, vol. 35, no. 1, pp. 107-114, 2009.
[14] Kobara K. and Imai H., "Pretty-simple Password-Authenticated Key-Exchange Under Standard Assumptions," *IEICE Transactions*, vol. E85-A, no. 10, pp. 2229-2237, 2002.
[15] Lee T., Hwang T., and Lin C., "Enhanced Three-Party Encrypted Key Exchange Without Server's Public Keys," *Computers and Security*, vol. 23, no. 7, pp. 571-577, 2004.
[16] Lee S., Kim H., and Yoo K., "Efficient Verifier-Based Key Agreement For Three Parties Without Server's Public Key," *Applied Mathematics and Computation*, vol. 167, no. 2, pp. 996-1003, 2005.
[17] Lin C., Sun H. and Hwang T., "Three-Party Encrypted Key Exchange Attacks and A Solution," *ACM Operating Systems Review*, vol. 34, no. 4, pp. 12-20, 2000.
[18] Lin C., Sun H., Steiner M., and Hwang T., "Three-Party Encrypted Key Exchange Without Server's Public Keys," *IEEE Communications Letters*, vol. 5, no. 12, pp. 497-499, 2001.
[19] Lu R. and Cao Z., "Simple Three-Party Key Exchange Protocol," *Computers and Security*, vol. 26, no. 1, pp. 94-97, 2007.
[20] MacKenzie P., "The PAK Suite: Protocols for Password-Authenticated Key Exchange," *Technical Report*, DIMACS Center, Rutgers University, Contributions to IEEE P1363.2, 2002.
[21] Nam J., Paik J., Kang H., Kim U., and Won D., "An Off-Line Dictionary Attack on A Simple Three-Party Key Exchange Protocol," *IEEE Communications Letters*, vol. 13, no. 3, pp. 205-207, 3009.
[22] Phan R., Yau W., and Goi B., "Cryptanalysis of Simple Three-Party Key Exchange Protocol (S-3PAKE)," *Information Science*, vol. 178, no. 13, pp. 2849-2856, 2008.
[23] Wang W. and Hu L., "Efficient and Provably Secure Generic Construction of Three-Party Password-Based Authenticated Key Exchange Protocols," *in Proceedings of the 7th International Conference On Cryptology*, India, pp. 118-132, 2006.
[24] Yoon E. and Yoo K., "Cryptanalysis of a Simple Three-Party Password-Based Key Exchange Protocol," *International Journal of Communication Systems*, vol. 24, no. 4, pp. 532-542, 2011.

Shuhua Wu is a lecturer at Networks Engineering Department, Information Science Technology Institute, China. Currently, he is a postdoctor at the Department of Computer Science and Engineering, Shanghai Jiaotong University. His research interests include cryptology and communication protocols.

Kefei Chen he received his PhD degree in Justus Liebig University Giessen, Germany, in 1994. His main research areas are classical and modern cryptography, theory and technology of network security, etc. Since 1996, he came to Shanghai Jiaotong University and became the professor at the Department of Computer Science and Engineering. Up to now (1996-2007), he has published more than 80 academic papers on cryptology.

Yuefei Zhu is a professor of Networks Engineering Department, Information Science Technology Institute, China. His research interests include cryptology and information security.