One-Round Protocol for Two-Party Verifier-Based Password-Authenticated Key Exchange*

Jeong Ok Kwon[1], Kouichi Sakurai[2], and Dong Hoon Lee[1]

[1] Graduate School of Information Security CIST, Korea University
Anam-dong Seongbuk-Gu, Seoul, 136-701 Korea
{pitapat, donghlee}@korea.ac.kr
[2] Department of Computer Science and Communication Engineering
Kyushu University, 6-10-1 Hakozaki, Higashi-ku, Fukuoka, 812-0053 Japan
sakurai@csce.kyushu-u.ac.jp

Abstract. Password-authenticated key exchange (PAKE) for two-party allows a client and a server communicating over a public network to share a session key using a human-memorable password *only*. PAKE protocols can be served as basic building blocks for constructing secure, complex, and higher-level protocols which were initially built upon the Transport Layer Security (TLS) protocol. In this paper, we propose a provably-secure *verifier-based* PAKE protocol well suited with the TLS protocol which requires only a single round. The protocol is secure against attacks using compromised server's password file and known-key attacks, and provides forward secrecy, which is analyzed in the ideal hash model. This scheme matches the most efficient verifier-based PAKE protocol among those found in the literature. It is the first provably-secure *one-round* protocol for verifier-based PAKE in the two-party setting.

1 Introduction

Password-authenticated key exchange. To communicate securely over an insecure public network it is essential that secret keys are exchanged securely. The shared secret key may be subsequently used to achieve some cryptographic goals such as confidentiality or data integrity. Password-authenticated key exchange (PAKE) protocols in the two-party setting are used to share a secret key between a client and a server using *only* a shared human-memorable password. These password-only methods have many merits in views of mobility and

* The first and third authors were supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) and the second author was supported by grant for International Cooperative Research from National Institute of Information and Communication (Theme: "A Research on Scalable Information Security Infrastructure on Ubiquitous Networks"). This work was done while the first author visits in Kyushu University, Japan.

H. Leitold and E. Markatos (Eds.): CMS 2006, LNCS 4237, pp. 87–96, 2006.

efficiency. Naturally, they are less expensive and more convenient than smart cards and other alternatives. This password-only method can also eliminate the requirement of a public key infrastructure (PKI). Due to the merits, protocols for PAKE can be used in several environments, especially in mobile networks.

In mobile networks session key exchange for the secure communication services has to be done efficiently using relatively small resources. One of the main efficiency issues in real applications over mobile networks is how to reduce the number of rounds, the computing time, and the size of the transmitted messages since clusters of mobiles have memory and processing constraints, and the mobile networks have limited bandwidth. Especially, the number of rounds is very important a factor in case that session keys have to be exchanged frequently.

The difficulty to design a scheme for PAKE comes from the usage of a password having low entropy. For a human to easily memorize a password, a password may have low-entropy (i.e., 4 or 8 characters such as a natural language phrase). These natural language phrases are weak because they are drawn from a relatively small dictionary. So they are susceptible to dictionary attacks, also known as password guessing attacks. The fundamental security goal of PAKE is security against dictionary attacks. Usually, dictionary attacks are classified into two classes. In *on-line dictionary attacks*, an adversary attempts to use a guessed password by participating in a key exchange protocol. If the protocol run is failed, the adversary newly starts the protocol with the server using another guessed password. This attack requires participation of the adversary. In *off-line dictionary attacks*, an adversary selects a password from a dictionary and verifies his guess in off-line manner. Since the adversary uses only recorded transcripts from a successful run of the protocol, no participation of the adversary is required. So such off-line attacks are undetectable.

On-line dictionary attacks are always possible, but the attacks do not become a serious threat because they can be easily detected and thwarted by counting access failures. That is, a failed guess can be detected by the server since one can count the number how many somebody terminates the protocol with failure. However, off-line dictionary attacks are more difficult to prevent. Even if there exist tiny amounts of redundancy information in flows of the scheme, then adversaries can mount an off-line dictionary attack by using the redundancy as a verifier for checking whether a guessed password is correct or not. The main security goal of schemes for PAKE is to restrict the adversary to on-line dictionary attacks only.

In addition to dictionary attacks, a fundamental security goal of PAKE is *key secrecy*. This security level means that no computationally bounded adversary should learn anything about the session keys shared between two honest parties by eavesdropping or sending messages of its choice to parties in the protocol. Other desirable security goals are as follows (formal definitions are given in Section 2). The importance of the following attributes depends on the real applications. *Forward Secrecy* means that even with the password of the users any adversary does not learn any information about session keys which are successfully established between honest parties without any interruption. A PAKE

protocol is secure against *known-key attacks* if the following conditions hold: First, compromise of multiple session keys for sessions other than the one does not affect its key secrecy. This notion of security means that session keys are computationally independent from each other. A bit more formally, this security protects against "Denning-Sacco" attacks [10] involving compromise of multiple session keys (for sessions other than the one whose secrecy must be guaranteed). Next, an adversary cannot gain the ability to performing off-line dictionary attacks on the users' password from using the compromised session keys which are successfully established between honest users.

Two models for 2-pary PAKE. PAKE protocols for 2-party are classified into two models according to the sameness of knowledge used in authenticated-key exchange: *Symmetric model* in which a client and a server use the same knowledge related with a password to authenticate each other and establish a session key. In usually, the client and the server own a password in plaintext form. *Asymmetric* (or *verifier-based*) *model* in which a client and a server use the asymmetric knowledge related with a password to authenticate each other and establish a session key. In usually, the client memorizes a password, while a server stores an image (called a verifier) of the password under a one-way function instead of a plaintext version of the password.

Most previous 2-party PAKE protocols have been constructed in the *random oracle model*. The random oracle model is a security model, where we assume that a certain function is an "ideal" function. In the ideal hash model, we assume a hash function is a random function and in the ideal cipher model, we assume that a block cipher is a random permutation.

Many provably-secure PAKE protocols in the *symmetric model* have been suggested [5,9,20,15,16,12,6,7,3,1]. In [5], Bellare *et al.* provided a formal model for PAKE and proved the security of a protocol of [4] in the ideal cipher model. Boyko *et al.* presented PAKE protocols provably-secure in the ideal hash model [9,20]. Katz *et al.* [15,16] and Goldreich *et al.* [12] proposed PAKE protocols provably-secure in the standard model, independently. Bresson *et al.* [6] proved the security of *AuthA* which is a PAKE protocol considered for standardization by the IEEE P1363 standard working group, in the ideal hash model and the ideal cipher model. Also Bresson *et al.* reduced the number of ideal functions and proved the security of *AuthA* in the ideal hash model [7]. Recently, Abdalla *et al.* proposed PAKE protocols provably-secure in the ideal hash model [3,1]. *Verifier-based* PAKE protocols has been extensively studied in the last few years: A-EKE, B-SPEKE, SRP, GXY, SNAPI-X, AuthA, PAK-Z+, AMP, EPA, and VB-EKE [8,14,23,19,21,5,11,17,13,2].

Server compromise in symmetric model. In a protocol of symmetric model, the client and the server own a password. Hence the corruption of the server reveals the passwords themselves and an adversary that is able to access to the server's password file, can immediately masquerade as a legitimate client by using only the corrupted password without executing of any off-line dictionary attack. To better understand the damage of the server compromise in the symmetric

model, consider the protocol [3] in the symmetric model suggested by Abdalla *et al.* in Figure 1. In this protocol, it easy to see that if the server compromise occurs, an adversary who can access the compromised passwords, can immediately masquerade as a legitimate client C to the server since the adversary knows password pw of client C.

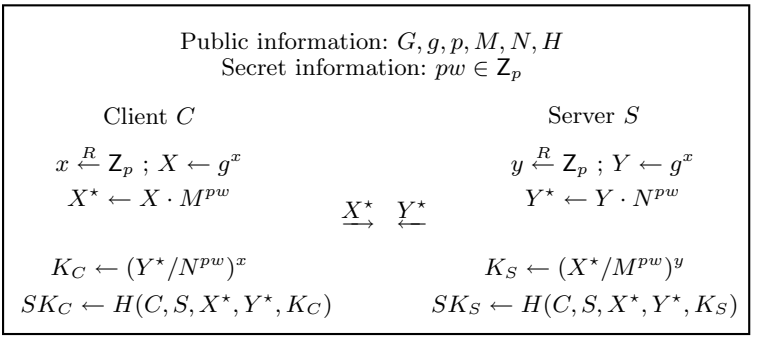

Fig. 1. A PAKE protocol in symmetric model

Server compromise in verifier-based model. PAKE protocols in verifier-based model are designed to limit the damage due to the server compromise. In a verifier-based protocol, the client owns a password, but the server owns a verifier of the password. Hence the corruption of the server just reveals the verifier not the password itself. Of course the server compromise still allows off-line dictionary attacks, but even if the password file is compromised, the attacker has to perform additional off-line dictionary attacks to find out the passwords of the clients. It will give the server system's administrator time to react and to inform its clients, which would reduce the damage of the corruption. Therefore, the main security goal of verifier-based PAKE protocols is to force an adversary who steals a password file from a server and wants to impersonate a client in the file, to perform an off-line dictionary attack on the password file. The difficulty of off-line dictionary attacks on the corrupted password file depends on the difficulty of finding the original password from the verifier.

1.1 Our Work in Relation to Prior Work

Two-party PAKE protocols can be served as basic building blocks for constructing secure, complex, and higher-level protocols which were initially built upon the Transport Layer Security (TLS) protocol [22]. In this paper we focus on designing a round-efficient verifier-based two-party PAKE protocol that can be used in the key exchange phase of the TLS protocol. In the TLS protocol, the key exchange protocol is executed right after the "hello" flows in which the first is from the client to the server, then the second is from the server to the client. To improve round-efficiency, in the paper we assume that parties can transmit messages simultaneously. Actually, in many common scenarios parties are able

to transmit messages simultaneously. By taking advantage of the communication characteristics of the network it may be possible to design protocols with improved latency. This is the focus of the present work.

Recently, a provably-secure one-round PAKE protocol in symmetric model achieving the goal is proposed by Abdalla *et al.* [3] and its forward secrecy is proved by Abdalla *et al.* in [1]. Because the protocol in [3,1] is PAKE protocol in symmetric model in which, to achieve authenticated key exchange, it must be assumed that a client and a server own information related with a password in the same form. On the other hand, in order to immune to attacks using compromised server's password file, in verifier-based PAKE, a client and a server use each other asymmetric information for a password to achieve authenticated key exchange. So the simple and novel approach in [3,1] can not be directly applied to verifier-based PAKE because of the critical assumption for possessing of the symmetric information of passwords. We note that converting a PAKE protocol in symmetric model into a PAKE protocol in verifier-based model is not easy at all. Since a mechanism converting a PAKE protocol in symmetric model to verifier-based PAKE protocol must not reveal any redundancy information that adversaries can mount an off-line dictionary attack. To solve this problem, we use an additional multiplicative function where the multiplicative function used in [13,3,1] multiplies the protocol messages by a value which is made with a password.

Table 1. Comparisons of complexity with the related verifier-based protocols

Scheme/ Resource	Round	Modular exponentiations		Communication		Security	Assumption
		Client	Server	Client	Server		
B-SPEKE [14]	3	2	1	$\|c\| + \|\tau\|$	$2\|c\| + \|p\|$	-	-
SRP [23]	3	2	1	$\|p\|$	$2\|p\|$	-	-
PAK-Z+ [11]	3	$1+E_{S.gen}$	$E_{S.ver}$	$\|p\| + \|\sigma\|$	$\|p\| + 3\|l\|$	KK&FS	Ideal hash
AMP [17]	4	1	3	$\|p\| + \|l\|$	$> \|p\| + \|l\|$	-	-
EPA [13]	2	1	2	$\|p\|$	$\|p\|$	FS	Ideal hash
VB-EKE [2]	2	1	4	$3\|p\|$	$\|p\|$	-	-
Our Scheme	1	2	1	$\|p\|$	$2\|p\|$	KK&FS	Ideal hash

$S = \{S.key, S.gen, S.ver\}$: a signature scheme, $E_{S.gen}$: the number of exponentiations in signing, $E_{S.ver}$: the number of exponentiations in verifying, $|\sigma|$: the length of a signature, $|p|$: the length of a prime p of Z_p^* , $|l|$: the length of an output of a hash function, $|c|$: the length of a symmetric encryption, $|\tau|$: the length of a message authentication code. An FS protocol is a forward-secure key exchange protocol and a KK protocol is a secure key exchange protocol against known-key attacks.

We compare the resources of our protocol with the protocols, B-SPEKE [14], SRP [23], AMP [17], and PAK-Z+ [11] submitted to the IEEE P1363.2 standard proposal for Password-Based Public Key Cryptographic Techniques, and recently proposed protocols, EPA and VB-EKE. Table 1 summarizes the comparisons of complexity and security, where communication cost is the total number of bits that a client and a server send during a protocol run. In the comparison of computation cost, we are applying pre-computation technique to the protocols to minimize on-line computational overhead. EPA requires the smallest

exponentiations and communication cost on the client side, and the smallest rounds among the previously suggested protocols. However, EPA has a type of "challenge-response" mechanism (i.e., firstly, the client sends a challenge message to the server in the first round and then the server sends a respond message generated by using the client's challenge messages in the second round. Finally, the client can compute the session key after receiving the server's message in the second round), so it is no longer possible to swap the flows by employing the advantage of simultaneous message transmission. We explore the possibility of designing a protocol for verifier-based PAKE which can be implemented in only *one-round* (assuming simultaneous message transmission). Our protocol gives a novel method to make it possible that the client and the server send independently their messages for the key exchange in a single round since the parties can add authentication to messages regardless of other parties' messages. Thus the client can compute the session key after receiving the server's message in the first round. On the other hand, our protocol is only slightly less efficient from a computational perspective on the client side than EPA. The proposed protocol is the first provably-secure verifier-based two-party *one-round* PAKE protocol providing forward secrecy in the ideal hash model.

2 Security Model

We use the standard notion of security as defined in [5] and used extensively since then. This will be necessary for proving the security about our schemes in later sections. We fix nonempty sets $\mathcal{C}$ of potential clients and $\mathcal{S}$ of potential servers. We consider a password-authenticated verifier-based key exchange protocol in which two parties, a client $C \in \mathcal{C}$ and a server $S \in \mathcal{S}$ want to exchange a session key.

Initialization. A party P may have many instances of the protocol, which is either a client or a server. An instance of P is represented by an oracle P^s, for any $s \in \mathbb{N}$. Each client $C \in \mathcal{C}$ holds a password pw_C obtained at the start of the protocol using a password generation algorithm $\mathcal{PG}(1^\kappa)$ which on input a security parameter 1^κ outputs a password pw uniformly distributed in a password space $\mathsf{Password}$ of size $\mathcal{PW}$. Each server $S \in \mathcal{S}$ holds a vector (the so called a verifier) $pw_S = [f(pw_C)]_{C \in \mathcal{C}}$ with an entry for each client, where f is a one-way function. We assume the set $\mathcal{S}$ contains a single server.

Partnering. Let sid_C^s be the concatenation of all messages that oracle C^s has sent and received. For the concatenation the messages are ordered according to the kinds of owners, e.g., the first part is the client messages and the server's messages are concatenated to them. Let a partner identifier pid_C^s for instance C^s be a set of the identities of the parties with whom C^s intends to establish a session key. pid_C^s includes C^s itself. The oracles C^s and S^t are *partnered* if $\mathsf{pid}_C^s = \mathsf{pid}_S^t$ and $\mathsf{sid}_C^s = \mathsf{sid}_S^t$.

Queries. An adversary $\mathcal{A}$ is a probabilistic polynomial-time machine that controls all the communications and makes queries to any oracle. The queries that $\mathcal{A}$ can use are as follows:

- $\mathsf{Send}(P^s, M)$: This query is used to send a message M to instance P^s (this models active attacks on the part of the adversary). When P^s receives M, it responds according to the key exchange protocol.
- $\mathsf{Execute}(C^s, S^t)$: This query models passive attacks, where the adversary gets the instances of honest executions of the protocol by C^s and S^t. (Although the actions of the Execute query can be simulated via repeated Send oracle queries, this particular query is needed to distinguish between passive and active attacks in the definition of forward secrecy.)
- $\mathsf{Reveal}(P^s)$: This query models the adversary's ability to obtain session keys, i.e., this models *known-key attacks* in the real system. The adversary is given the session key for the specified instance.
- $\mathsf{Corrupt}(P)$: This models exposure of the long-term key held by P. The adversary is assumed to be able to obtain long-term keys of parties, but cannot control the behavior of these players directly (of course, once the adversary has asked a query $\mathsf{Corrupt}(P)$, the adversary may impersonate P in subsequent Send queries.) We restrict that on $\mathsf{Corrupt}(P)$ the adversary only can get the long-term key, but cannot obtain any internal data of P.
- $\mathsf{Test}(P^s)$: This query is used to define the advantage of an adversary. This query is allowed only once by an adversary $\mathcal{A}$, and only to *fresh* oracles, which is defined later. On this query a coin b is flipped. If b is 1, the session key sk_P^s held by P^s is returned. Otherwise, a string randomly drawn from a session key distribution is returned.

Freshness. We define a notion of *freshness* considering forward secrecy which means that an adversary does not learn any information about *previously* established session keys when making a Corrupt query. We say an oracle C^s is *fresh* if the following conditions hold:

- C^s has computed a session key $sk_C^s \neq \mathrm{NULL}$ and neither C^s nor S^t have been asked for a Reveal query, where C^s and S^t are partnered.
- No $\mathsf{Corrupt}(P)$ for any $P \in \mathsf{pid}_C^s$ has been asked by the adversary before a query of the form $\mathsf{Send}(C^s, *)$.

PAKE Security. Consider a game between an adversary $\mathcal{A}$ and a set of oracles. $\mathcal{A}$ asks the above queries to the oracles in order to defeat the security of a protocol $\mathcal{P}$, and receives the responses. At some point during the game a Test query is asked to a fresh oracle, and the adversary may continue to make other queries. Finally the adversary outputs its guess b' for the bit b used by the Test oracle, and terminates. We define CG to be an event that $\mathcal{A}$ correctly guesses the bit b. The advantage of adversary $\mathcal{A}$ must be measured in terms of the security parameter k and is defined as $\mathsf{Adv}_{\mathcal{P},\mathcal{A}}(k) = 2 \cdot \Pr[\mathsf{CG}] - 1$. The advantage function is defined as $\mathsf{Adv}_{\mathcal{P}}(k, T) = \max_{\mathcal{A}}\{\mathsf{Adv}_{\mathcal{P},\mathcal{A}}(k)\}$, where $\mathcal{A}$ is any adversary with time complexity T which is polynomial in k.

Definition 1. We say a protocol $\mathcal{P}$ is a *secure password-authenticated key exchange scheme* if the following two properties are satisfied:

- Validity: if all oracles in a session are partnered, the session keys of all oracles are same.

- Key secrecy: $\mathsf{Adv}_{\mathcal{P}}(k,T)$ is bounded by $\frac{q_{se}}{\mathcal{PW}} + \epsilon(k)$, where $\epsilon(k)$ is negligible. q_{se} is the number of Send queries and $\mathcal{PW}$ is the size of the password space.

(1) We say a protocol $\mathcal{P}$ is a secure PAKE scheme if validity and privacy are satisfied when no Reveal and Corrupt queries are allowed.
(2) We say a protocol $\mathcal{P}$ is a secure PAKE-KK scheme if validity and key secrecy are satisfied when no Corrupt query is allowed.
(3) We say a protocol $\mathcal{P}$ is a secure PAKE-FS scheme if validity and key secrecy are satisfied when no Reveal query is allowed.
(4) We say a protocol $\mathcal{P}$ is a secure PAKE-KK&FS scheme if validity and key secrecy are satisfied.

3 One-Round Verifier-Based PAKE Protocol for Two-Party

We now present our protocol, $\mathcal{VB}$-$\mathcal{PAKE}$ with implicit authentication for verifier-based PAKE in the two-party setting. In this paper, we assume the parties can transmit messages simultaneously.

$\mathcal{VB}$-$\mathcal{PAKE}$ can be seen a version for verifier-based PAKE of the protocol [3,1] in symmetric model. To convert the protocol in [3,1] into the verifier-based protocol, $\mathcal{VB}$-$\mathcal{PAKE}$, secure against server compromise attacks, we use an additional verifier and ephemeral Diffie-Hellman key exchange. We can easily see that $\mathcal{VB}$-$\mathcal{PAKE}$ is secure against server compromise attacks. Even if the verifiers, v_1 and v_2, are revealed to an adversary, the adversary can not immediately masquerade as C to S without off-line dictionary attacks since the adversary does not know pw of C. Only the client C knowing pw can compute g_1^y from Z and sk_C. The description of $\mathcal{VB}$-$\mathcal{PAKE}$ follows:

Public information. A finite cyclic group $\mathbb{G}$ of order q in $\mathbb{Z}_p^*$. Two primes p, q such that $p = 2q + 1$, where p is a safe prime such that the CDH problem is hard to solve in $\mathbb{G}$. g_1 and g_2 are generators of $\mathbb{G}$ both having order q, where g_1 and g_2 must be generated so that their discrete logarithmic relation is unknown. Hash functions $\mathcal{H}_i$ from $\{0,1\}^*$ to $\{0,1\}^{l_i}$, for $i = \{0,1\}$.

Initialization. A client C obtains pw at the start of the protocol using the password generation algorithm $\mathcal{PG}(1^k)$. C sends $v_1 = g_1^{\mathcal{H}_0(C||S||pw)} mod\ p$ and $v_2 = g_2^{\mathcal{H}_0(C||S||pw)}$ $mod\ p$ which are verifiers of the password to a server S over a secure channel. Upon receiving the verifiers, S stores them in a password file with an entry for C.

Round 1. C chooses a random number $x \in \mathbb{Z}_q^*$, computes $\mathsf{X} = g_1^x \cdot v_2\ mod\ p$, and sends ($C$, X) to S. S selects random numbers $y, z \in \mathbb{Z}_q^*$, computes $\mathsf{Z} = g_1^z \cdot v_2\ mod\ p$ and $\mathsf{Y} = g_1^y \cdot v_1^z\ mod\ p$, and sends ($S$,$\mathsf{Y}$,$\mathsf{Z}$) to C.

Key computation. Upon receiving (S,Y,Z), C computes $T = (\mathsf{Z}/v_2)^{\mathcal{H}_0(C||S||pw)} mod\ p$, $K_C = (\mathsf{Y}/T)^x\ mod\ p$ and the session key $\mathsf{sk}_C = \mathcal{H}_1(C\|S\|\mathsf{sid}_C\|K_C)$, where $\mathsf{sid}_C = \mathsf{X}\|\mathsf{Y}\|\mathsf{Z}$. Upon receiving ($C$, X), S computes $K_S = (\mathsf{X}/v_2)^y\ mod\ p$ and the session key $\mathsf{sk}_S = \mathcal{H}_1(C\|S\|\mathsf{sid}_S\|K_S)$, where $\mathsf{sid}_S = \mathsf{X}\|\mathsf{Y}\|\mathsf{Z}$.

Security Analysis. We now present that under the intractability assumption of the CDH problem the proposed protocol is a secure key exchange protocol against dictionary attacks and known-key attacks and provides forward secrecy.

Theorem 1. Assuming $\mathbb{G}$ satisfies the CDH assumption, $\mathcal{VB}$-$\mathcal{PAKE}$ is a secure PAKE-KK&FS scheme when $\mathcal{H}_1$ is modeled as a random oracle. Concretely,

$$\mathsf{Adv}^{\mathsf{pake\text{-}kk\&fs}}_{\mathcal{VB}-\mathcal{PAKE}}(k, T, q_{ex}, q_{se}, q_h) \leq 4q_h N_s \mathsf{Adv}^{\mathsf{cdh}}_{\mathbb{G},\mathcal{A}}(T) + \frac{q_{se}}{\mathcal{PW}} + \frac{(q_{ex} + q_{se})^2}{q},$$

where T is the maximum total game time including an adversary's running time, and an adversary makes q_{ex} Execute queries, q_{se} Send queries, and q_h Hash queries to $\mathcal{H}_1$. N_s is the upper bound of the number of sessions that an adversary makes, and $\mathcal{PW}$ is the size of the password space.

Proof of Theorem 1. The detailed proof of this theorem appears in the full version of the paper [18].

4 Concluding Remarks

All previous provably-secure verifier-based PAKE protocols have been constructed in the ideal hash model [11,13]. In this paper, we have also proposed a provably-secure protocol in the ideal hash model. However, no provably-secure verifier-based PAKE scheme in the standard model has been proposed yet. The difficulty is dealing with a pre-shared password for secure key agreement. Designing an efficient verifier-based PAKE protocol which is probably-secure in the standard model is the subject of ongoing work.

References

1. M. Abdalla, E. Bresson, O. Chevassut, A. Essiari, B. M öller, and D. Pointcheval, *Provably Secure Password-Based Authentication in TLS*, In Proc. of ASIACCS'06, ACM Press, pages 35-45, ACM Press, 2006.
2. M. Abdalla, O. Chevassut, and D. Pointcheval. *One-time Verifier-based Encrypted Key Exchange*, In PKC '05, LNCS 3386, pages 47-64, Springer-Verlag, 2005.
3. M. Abdalla and D. Pointcheval. *Simple password-based encrypted key exchange protocols*, In Proc. of CT-RSA 2005, LNCS 3376, pages 191-208. Springer-Verlag, 2005.
4. S. Bellovin and M.merritt. *Encrypted Key Exchange: Password-Based Protocols Secure against Dictionary Attacks*, In Proc. of the Symposium on Security and Privacy, pages 72-84. IEEE Computer Society, 1992.
5. M. Bellare, D. Pointcheval and P. Rogaway. *Authenticated key exchange secure against dictionary attack* In Eurocrypt '00, LNCS 1807, pages 139-155, Springer-Verlag, 2000.
6. E. Bresson, O. Chevassut, and D. Pointcheval. *Security Proofs for an Efficient Password-Based Key Exchange*, In Proc. of the 10th ACM Conference on Computer and Communications Security, pages 241-250, ACM Press, 2003.

7. E. Bresson, O. Chevassut, and D. Pointcheval. *New Security Results on Encrypted Key Exchange*, In Proc. of PKC 04, LNCS 2947, pages 145-158, Springer-Verlag, 2004.
8. S. Bellovin and M. Merritt. *Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password-file compromise*, ACM Conference on Computer and Communications Security, pages 244-250, 1993.
9. V. Boyko, P. MacKenzie, and S. Patel. *Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman*, In Proc. of EUROCRYPT '01, LNCS 1807, pages 156-171, Springer-Verlag, 2001.
10. D. Denning and G. M. Sacco. *Timestamps in Key Distribution Protocols*, Communications of the ACM 24(8), pages 533-536, 1981.
11. C. Gentry, P. MacKenzie, and Z. Ramzan. *PAK-Z+*, Contributions to IEEE P1363, August 2005. Available from `http://grouper.ieee.org/groups/1363/`.
12. O. Goldreich and Y. Lindell. *Session-Key Generation using Human Passwords Only*, In Proc. of CRYPTO '01, LNCS 2139, pages 408-432. Springer-Verlag, 2001.
13. Y. H. Hwang, D. H. Yum, and P. J. Lee. *EPA: An Efficient Password-Based Protocol for Authenticated Key Exchange*, In ACISP '03, LNCS 2727, pages 452-463, Springer-Verlag, 2003.
14. D. Jablon. *Extended password key exchange protocols immune to dictionary attack, In Proc. of WETICE97 Workshop on Enterprise Security*, 1997.
15. J. Katz, R. Ostrovsky, and M. Yung. *Efficient Password-Authenticated Key Exchange using Human-Memorable Passwords*, In Proc. of EUROCRYPT '01, LNCS 2045, pages 475-494, Springer-Verlag, 2001.
16. J. Katz, R. Ostrovsky, and M. Yung. *Forward secrecy in Password-only Key Exchange Protocols*, In Proc. of SCN '02, LNCS 2576, pages 29-44, Springer-Verlag, 2002.
17. T. Kwon. Ultimate Solution to Authentication via Memorable Password, Contributions to IEEE P1363, May 2000. Available from `http://grouper.ieee.org/groups/1363/`.
18. J. O. Kwon, K. Sakurai and D. H. Lee. Full version of this paper, Available at `http://cist.korea.ac.kr/~pitapat/VBTS200610.ps`.
19. T. Kwon and J. Song. *Secure agreement scheme for gxy via password authentication*, Electronics Letters, 35(11):892-893, 1999.
20. P. MacKenzie. *More Efficient Password Authenticated Key Exchange*, In Proc. of the RSA Data Security Conference, Cryptographer's Track (RSA CT '01), LNCS 2020, pages 361-377, Springer-Verlag, 2001.
21. P.MacKenzie and R.Swaminathan. *Secure network authentication with password identification*, Presented to IEEE P1363a, August 1999.
22. M. Steiner, P. Buhler, T. Eirich, and M. Waidner. *Secure Password-Based Cipher Suite for TLS*, ACM Transactions on Information and System Security (TISSEC) 4(2):134-157, 2001.
23. T. Wu. *Secure remote password protocol*, In Proc. of the ISOC NDSS Symposium, pages 99-111, 1998.