International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 14 (2017) pp. 4329-4339

An Efficient Password-Only Authenticated Three-Party Key Exchange Protocol

Youngsook Lee[#1], Juryon Paik [*2] and Younsung Choi [#3]

[#] *Department of Cyber Security, Howon University, 64, 3-gil, Gunsan, Jeollabuk-do, 54058, Republic of Korea.*

[*] *Department of Digital Information & Statistics, Pyeongtaek University, 3825, Seodong-daero, Pyeongtaek-si, Gyeonggi-do, 17869, Republic of Korea.*

Abstract

Password-only authenticated key exchange (PAKE) protocols allow to generate cryptographically strong keys from human-memorable passwords. The design of an efficient PAKE protocol is difficult, especially in the three-party setting where dictionary attacks by malicious insiders are a major concern. The difficulty is well illustrated by the fact that after twenty years of research, only a handful of three-party PAKE protocols are known to be provably secure in a model that captures insider attacks. This paper proposes a new, efficient three-party PAKE protocol which incorporates the design principle of Bresson et al.'s two-party PAKE protocol called OMDHKE. A cost comparison in terms of communication and computation complexities shows that the overall performance of our protocol is superior to those of previously published three-party PAKE protocols. Moreover, our protocol has an advantage over its competitors in that it can be easily transformed into a simpler and more efficient protocol in an environment where undetectable online dictionary attacks do not pose a significant threat. We provide a proof of security for the protocol in the widely accepted model of Bellare et al. which captures insider attack.

Keyword: Cryptography, Authenticated key exchange, Dictionary attack, Three-party setting

INTRODUCTION

Human-memorable passwords have gained immense popularity as an almost universal means of authentication in today's distributed and heterogeneous computing environments. It is incredible to see how passwords, despite its many security shortcomings, have been persistently and widely used in getting access to numerous Internet services in our daily lives. The widespread and persistent popularity of passwords has attracted a great deal of attention on the design of password-only authenticated key exchange (PAKE) protocols in the areas of cryptography and network security [2], [4]-[7], [14], [17], [18], [41], [43]. As a class of key exchange protocols, PAKE protocols allow two or more parties to generate a shared, cryptographically strong key (commonly called a session key) from their weak, easy-to-remember passwords, and thereby to establish a secure communication channel over a public network which might be controlled by an adversary. Back in 1992, Bellovin and Merritt [5] proposed the first PAKE protocols, known as encrypted key exchange, with heuristic arguments for their security. Encrypted key exchange has been followed by many improvements and generalizations over the past two decades, including both practical and provably secure protocols; see, e.g., [4], [6], [7], [18], [27].

PAKE protocols should be designed to protect passwords of parties from a dictionary attack (also known as a password guessing attack), in which an attacker enumerates all possible passwords while testing each one against known password verifiers in order to determine the correct one. Dictionary attacks have been used by both criminals as well as law enforcement officers and digital forensics practitioners to gain access to password-protected data (e.g., on smartphones and portable devices based on RIM BlackBerry and Apple iOS platforms - see Elcomsoft Phone Password Breaker http://www.elcomsoft.com/eppb.html) Dictionary attacks can be classified into two types, online and offline. Unlike offline dictionary attacks where password guesses can be verified offline, online dictionary attacks are the ones where the attacker verifies each password guess via an online transaction with the password holder(s). In the two-party setting where the session key is usually established between a client and a server sharing the same password, online dictionary attacks do not pose a significant threat as they are likely to be detected and the server may lock out the problematic client after a certain number of invalid transactions. However, such attacks can go completely undetected in the three-party setting, where the session key is established between two clients who do not share the same password but hold their individual password shared only with a trusted server. Therefore, three-party

PAKE protocols should be able to resist undetectable online dictionary (UDOD) attacks as well as offline dictionary attacks [11], [28], [34], [37], [44]. Indeed, a three-party PAKE protocol is said to be secure if detectable online dictionary attacks are the best possible attacks that an adversary can mount against the protocol.

Efficiency, in addition to security, is an important consideration when designing three-party PAKE protocols. In general, the efficiency of a protocol is evaluated in terms of both the communication cost and the computation cost incurred by the protocol. Two common indicators for measuring the communication cost of a protocol are: (1) the round complexity, the number of communication rounds until the protocol terminates, and (2) the message complexity, the number of messages exchanged in the protocol. A communication round consists of all messages that can be sent by protocol participants independently in parallel. The computation cost of a protocol is typically measured by the number of cryptographic operations such as modular exponentiations (or scalar-point multiplications), symmetric /asymmetric encryptions/decryptions, hash function evaluations, and so on. Symmetric-key operations and hash function evaluations are often ignored in cost estimates since they are much faster than modular exponentiations and asymmetric-key operations. In the password-only three-party setting, preventing UDOD attacks usually entails a significant increase in both communication and computation costs. For this reason, security against UDOD attacks has occasionally been sacrificed in the interest of efficiency [2], [21], [38]. Despite all the work conducted over the past two decades, it still remains a challenging task to design a secure yet efficient three-party PAKE protocol. The key research challenge in this domain is to prevent insider dictionary attacks, as evidenced by the flaws discovered in published protocols [15], [19], [28]-[30], [37]. Note that in the three-party setting, clients do not share any password between them but hold their own private password and thus, a malicious client may attempt to discover the password of its partner client by mounting a dictionary attack. However, some protocols [1], [12], [37], [38] achieve provable security only in a restricted model where the adversary is not allowed to corrupt parties. A protocol proven secure in such a restricted model cannot guarantee its security against insider dictionary attacks. (Readers who are unfamiliar with formal security models are referred to Section 2.) Somewhat surprisingly, the majority of existing three-party PAKE protocols have never been proven secure in any model [10], [15], [16], [19], [25], [26], [33], [35] and/or have been found to be susceptible to either insider or outsider attacks [9], [10], [19], [21], [23], [24], [28]-[30], [32]-[34], [36], [37], [40], [44]. Indeed, there are only a very limited number of three-party PAKE protocols proven secure in a model that allows party corruption [24], [31], [40]. We remark that the recent protocol of Yang and Cao [42] achieves stronger notions of security but works only in a "hybrid" three-party setting where a server's public key is required in addition to passwords (see [20], [23], [36], [39], [45] for other recent protocols designed to work in a hybrid setting).

Table 1: Comparative Efficiency and Security for Three-Party PAKE Protocols

Protocol	Rounds	Messages	Exponentiations		Resistance to UDOD attacks	Security Proof
			Client	Server		
e3PAKE (our protocol)	3	6	3	4	Yes	Yes
LH [24]	6	6	3	4	Yes	Yes
Parallel version of LH [24]	4	14	3	4	Yes	Yes
WPWH [40]	6	6	3	$\simeq 4.5^{\dagger}$	Yes	Yes
Parallel version of WPWH [40]	4	10	3	$\simeq 4.5^{\dagger}$	Yes	Yes
2R3PAKE [31]	2	≥10	≥4	≥6	Yes	Yes
NWPAKE-1 [38]	≥3	≥8	≥4	≥6	Yes	Restricted model
3PAKE-RSA [12]	4	8	3	$2^{\ddagger}$	Yes	Restricted model
NGPAKE [37]	≥3	≥8	≥4	≥4	Yes	Restricted model
LHL [22]	6	6	3	4`	Yes	No

Parallel version of LHL [22]	4	10	3	4	Yes	No
LSSH-3PEKE [25]	7	7	3	4	Yes	No
GLMZ [15]	≥6	≥9	≥4.5	≥7	Yes	Broken[28]
CHY [8]	6	6	3	4	Yes	Broken[40]
Parallel version of CHY [8]	4	10	3	4	Yes	Broken[40]
S-EA-3PAKE [21]	4	14	≥3.5	≥7	Yes	Broken[29]
GPAKE [1]	≥3	≥8	≥4	≥4	No[37]	Restricted model
NWPAKE-2 [38]	≥2	≥8	≥4	≥6	No	Broken[31
S-IA-3PAKE [21]	4	8	≃3.5	≃7	No	Broken[29]
Huang [16]	5	5	2	2	No[44],[24]	Broken[44
S-3PAKE [26]	5	5	≃2.5	≃3	No[15],[34], [19]	Broken[10],[15], [34],33]
3PAKE [2]	2	4	2	2	No[37]	Broken[30

† the numbers of scalar-point multiplications.

‡ the numbers of exponentiations modulo an RSA modulus.

In this paper, we present an efficient three-party PAKE protocol that not only provides resistance to UDOD attacks but also achieves provable security against an active adversary with the corruption capability. Our protocol named e3PAKE ("e" for "efficient") is based on Bresson et al.'s two-party PAKE protocol [6], called OMDHKE, which in turn originates from the AuthA protocol of Bellare and Rogaway [4]. We prove the security of e3PAKE in the widely accepted indistinguishability-based model of Bellare et al. [3] — this model is, probably, one of the most popular models in the provable security paradigm for key exchange protocols. The overall performance of e3PAKE is superior to those of previously published three-party PAKE protocols. It runs in 3 communication rounds with 6 messages in total and requires each client and the sever to perform 3 and 4 modular exponentiations, respectively. Table 1 compares e3PAKE with other three-party PAKE protocols both in terms of efficiency and security. Although the 2R3PAKE protocol [31] runs only in 2 rounds, it incurs significantly higher message complexity and computation cost in comparison to e3PAKE. From the standpoint of the computational load of the server, the 3PAKE-RSA protocol [12] is the most efficient among all protocols that provide resistance to UDOD attacks. However, this protocol has a hidden computation cost not reflected in Table 1. In each session of 3PAKE-RSA, each client must generate an RSA modulus n and a prime encryption key e anew in order to achieve perfect forward secrecy. The generation of n and e , though its cost is hidden in the table, is the most expensive operation required for the 3PAKE-RSA protocol. It is also worthwhile to mention that in our protocol, the third round consists of 2 messages and is dedicated to preventing UDOD attacks. This means that our protocol can be easily simplified into a 2-round 4-message protocol if security against UDOD attacks is not desired (see Section 4). In contrast, the protocols of [24], [40], [31] cannot further reduce their round complexity even if resistance to UDOD attacks is not required.

The remainder of this paper is structured as follows. Section 2 presents a standard definition of security for three-party PAKE protocols. Section 3 reviews Bresson et al.'s OMDHKE protocol and prove the security of its slight variant. We then present our proposed three-party PAKE protocol, e3PAKE, and prove its security in Section 4. Section 5 concludes the paper.

THE SECURITY MODEL

Here we describe a security model adapted from Bellare et al.'s indistinguishability-based model [3] for analysis of PAKE protocols. This will be the model that is used to prove the security of our proposed three-party PAKE protocol.

Participants. Let $\mathcal{C}$ be the set of all clients registered with a trusted server S. Any two clients $C, C' \in \mathcal{C}$ may run a three-party PAKE protocol P with S at any point in time to establish a session key. Let $\mathcal{U} = \mathcal{C} \cup \{S\}$. A user $U' \in \mathcal{U}$ may execute

the protocol multiple times (including concurrent executions) with the same or different participants. Thus, at a given time, there could be many instances of a single user. We use $\prod_U^i$ to denote instance i of user U. We say that a client instance $\prod_U^i$ accepts when it successfully computes its session key sk_C^i in an execution of the protocol.

Long-term keys. Each client $C \in \mathcal{C}$ chooses a password pw_C from a fixed dictionary , and shares it with the server S via a secure channel. Accordingly, S holds all the passwords $\{ pw_C \mid C \in \mathcal{C} \}$. Each password pw_C is used as the long-term secret key of C and S.

Partnering. Informally, two instances are said to be *partners* if they participate in a protocol execution and establish a (shared) session key. Formally, the partnership relations between instances is defined using the notions of session identifiers and partner identifiers. Session identifier (*sid*) is a unique identifier of a protocol session and is usually defined as a function of the messages transmitted in the session. Let sid_U^i denotes the *sid* of instance $\prod_U^i$. A partner identifier (*pid*) is a sequence of identities of participants of a specific protocol session. Instances are given as input a *pid* before they can run the protocol. pid_U^i denotes the pid given to instance $\prod_U^i$. In a typical session, there will be three participants, namely two clients C and C' and the server S . We say that two instances $\prod_C^i$ and $\prod_{C'}^j$ are partners if all of the following conditions are satisfied: (1) both $\prod_C^i$ and $\prod_{C'}^j$ have accepted, (2) $sid_C^i = sid_{C'}^j$, and (3) $pid_C^i = pid_{C'}^j$.

Adversary capabilities. During the executions of the protocol, the probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ has the complete control of all message exchanges and may ask users to open up access to session keys and passwords. The capabilities of $\mathcal{A}$ are captured via a pre-defined set of oracle queries as described below.

• Excute ($\prod_C^i$, $\prod_C^j$, $\prod_S^k$) : This query models passive attacks against the protocol. It prompts an execution of the protocol between the instances $\prod_C^i$, $\prod_C^j$, and $\prod_S^k$, and returns the transcript of the protocol execution to $\mathcal{A}$.

• Send ($\prod_U^i$, m) : This query sends message m to instance $\prod_U^i$, modelling active attacks against the protocol. Upon receiving m , the instance $\prod_U^i$ proceeds according to the protocol specification. The message output by $\prod_U^i$, if any, is returned to $\mathcal{A}$. A query of the form Send ($\prod_C^i$, start: (C, C' , S)) prompts $\prod_C^i$ to initiate the protocol with $pid_C^i = (C, C', S)$.

• Reveal ($\prod_C^i$) : This query captures the notion of known key security, and if $\prod_C^i$ has accepted, returns the session key sk_C^i back to $\mathcal{A}$. However, this session (key) will be rendered unfresh (see Definition 1).

• Corrupt (U): This query returns U 's password pw U to $\mathcal{A}$. If $U = S$ (i.e., the server is corrupted), all clients' passwords stored by the server are returned. This query captures not only the notion of forward secrecy but also attacks by malicious clients.

• Test ($\prod_C^i$): This query can be asked only if $\prod_C^i$ has accepted, and is used to define the indistinguishability-based security of the protocol. Depending on the hidden bit b chosen by the Test oracle uniformly at random from {0,1} , the query outputs the real session key sk_C^i held by $\prod_C^i$ if $b = 1$, or outputs a random key drawn from the session-key space if b = 0 . $\mathcal{A}$ is allowed to ask as many Test queries as it wishes. All Test queries are answered using the same value of the hidden bit b . Namely, the keys output by the Test oracle are either all real or all random. But, we require that for each different set of partners, $\mathcal{A}$ should access the Test oracle only once.

We represent the amount of queries used by an adversary as an ordered sequence of five non-negative integers, $Q = (q_{\text{exec}} , q_{\text{send}} , q_{\text{reve}} , q_{\text{corr}} , q_{\text{test}})$, where the five elements refer to the numbers of queries that the adversary made respectively to its Execute, Send, Reveal, Corrupt , and Test oracles. We call this usage of queries by an adversary the *query complexity* of the adversary.

AKE Security. The typical indistinguishability-based security of a three-party PAKE protocol is defined via the notion of *freshness*. Intuitively, an instance is considered to be fresh if it holds a session key which should not be known to the adversary , and is considered to be unfresh if its session key (or some information about the key) can be known by trivial means. Formally, freshness is defined as follows:

Definition 1. *An instance* $\prod_C^i$ *is fresh if none of the following occurs*: (1) $\mathcal{A}$ *queries* Reveal($\prod_C^i$) *or* Reveal($\prod_{C'}^j$), *where* $\prod_C^j$ *is the partner of* $\prod_C^i$, *and* (2) $\mathcal{A}$ *queries* Corrupt(U) , *for some* $U \in pid_C^i$, *before* $\prod_C^i$ *or its partner* $\prod_{C'}^j$ *accepts*.

Given the definition of freshness, the security of a three-party PAKE protocol P against $\mathcal{A}$ is defined in terms of the probability that $\mathcal{A}$ can correctly guess the hidden bit b chosen by the Test oracle in the following two-stage experiment:

Experiment **Exp**$_0$:

Stage 1. A makes any oracle queries at will as many times as it wishes, except that:

• $\mathcal{A}$ is not allowed to ask the Test (Π_C^i) query if the instance Π_C^i is unfresh.

• $\mathcal{A}$ is not allowed to ask the Reveal (Π_C^i) query if it has already made a Test query to Π_C^i

or $\Pi_{C'}^i$, where $\Pi_{C'}^i$ is the partner of Π_C^i.

Stage 2. Once $\mathcal{A}$ decides that Stage 1 is over, it outputs a bit b' as a guess on the hidden bit b chosen by the

Test oracle. A is said to succeed if $b = b'$

Let Succ_0 be the event that $\mathcal{A}$ succeeds in the experiment **Exp**$_0$. The advantage of $\mathcal{A}$ in breaking the security of the authenticated key exchange protocol P is $\text{Adv}_P^{\text{AKE}}(\mathcal{A}) = 2 \cdot \Pr_{P,A}[\text{Succ}_o] - 1$.

Definition 2. *$\mathcal{A}$ three-party PAKE protocol P is* AKE*-secure if, for any* PPT *adversary $\mathcal{A}$ asking at most q_{send}* Send *queries,* $\text{Adv}_P^{\text{AKE}}(\mathcal{A})$ *is only negligibly larger than $c \cdot q_{send}/|\mathcal{D}|$, where c is a very small constant (usually around 2 or 4) when compared with $|\mathcal{D}|$.*

To quantify the security of protocol P in terms of the amount of resources expended by adversaries, we let $\text{Adv}_P^{\text{AKE}}(t, Q)$ denote the maximum value of $\text{Adv}_P^{\text{AKE}}(\mathcal{A})$ over all PPT adversaries $\mathcal{A}$ with time complexity at most t and query complexity at most Q.

THE OMDHKE PROTOCOL

The OMDHKE protocol, Bresson et al.'s two-party PAKE protocol [6], originates from the AuthA protocol [4] which was submitted for standardization by the IEEE P1363.2 Standard working group. Since AuthA is open-ended in how the Diffie-Hellman values can be encrypted, OMDHKE was proposed to prove that AuthA is secure in the random oracle model when the encryption primitive is instantiated via a mask generation function computed as the product of the Diffie-Hellman value and a hash of the password.

The arithmetic is in a finite cyclic group $\mathbb{G}$ of prime order q. Let g be a generator of $\mathbb{G}$. The protocol uses three hash functions:

- $G : \{0, 1\}^* \rightarrow \mathbb{G}$,
- $F : \{0, 1\}^* \rightarrow \{0, 1\}^{\mathcal{K}}$ where $\mathcal{K}$ is the bit-length of the authenticator $Auth_{\text{SA}}$ (see the protocol description below), and
- $H : \{0, 1\}^* \rightarrow \{0, 1\}^{\ell}$ where ℓ is the bit−length of session keys.

As illustrated in Fig. 1, OMDHKE is a two-round two-message protocol and runs between a client $\mathcal{A}$ and a server S. $\mathcal{A}$ and S are assumed to have pre-established a shared password pw_{A}. The protocol starts when $\mathcal{A}$ chooses a random $x \in \mathbb{Z}_q^*$, computes $\bar{X} = g^x$, $PW_A = G(pw_A)$ and $X = X \cdot PW_A$, and sends $\langle A, \bar{X} \rangle$ to S. Upon receiving the message $\langle A, \bar{X} \rangle$ from A, S computes $PW_{\text{A}} = G(pw_A)$ and $X = \bar{X} / PW_A$, chooses a random $z \in \mathbb{Z}_q^*$, and computes $Z = g^z$, $K = X^z$ and $Auth_{SA} = \text{F}(A \| S \| \bar{X} \| Z \| PW_A \| K)$. Then S sends $\langle S, Z, Auth_{SA} \rangle$ to $\mathcal{A}$ and computes the session key, $sk = H(A \| S \| \bar{X} \| Z \| PW_A \| K)$. $\mathcal{A}$ computes $K = Z^x$, verifies $Auth_{SA}$ in the straightforward way, and computes the session key $sk = H(A \| S \| \bar{X} \| Z \| PW_A \| K)$ if the verification succeeds.

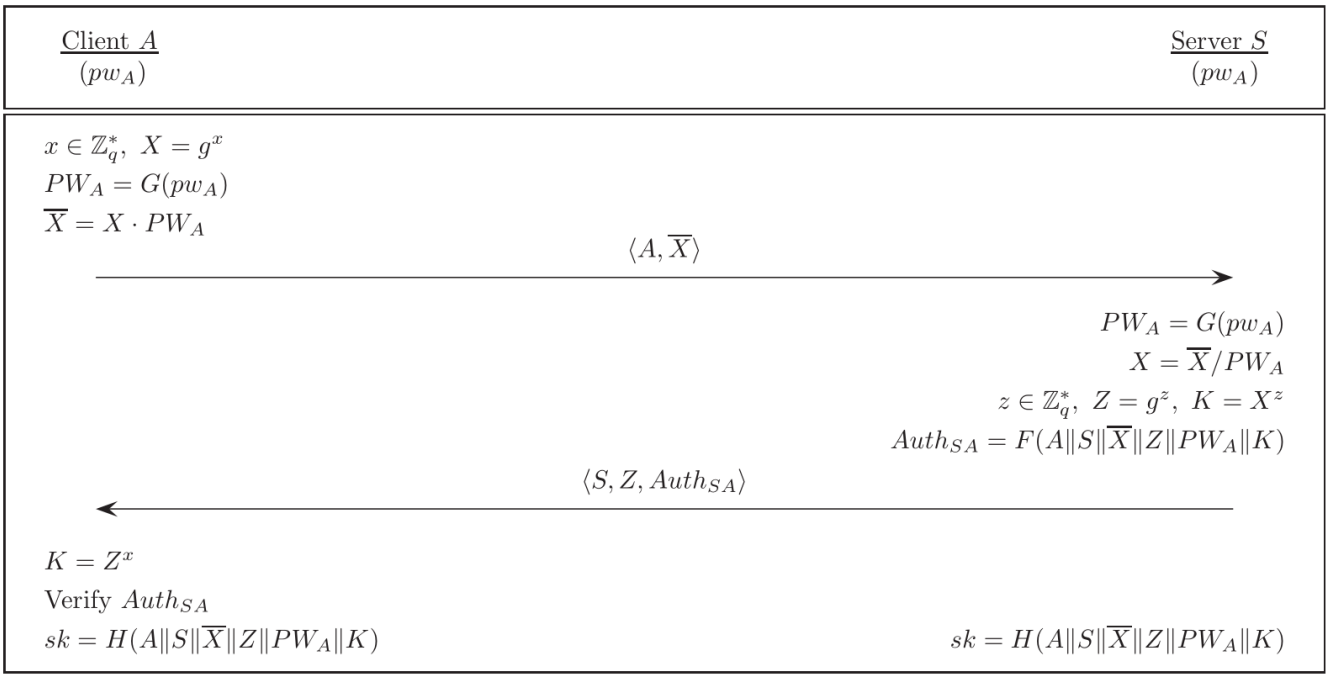

Figure 1. OMDHKE: Bresson et al.'s two-party PAKE protocol [6].

OMDHKE provides server-to-client authentication via the authenticator $Auth_{SA}$ and was proven secure in Bellare et al.'s model [3] under the assumption that the hash functions G, F and H are random oracles. Our proposed three-party PAKE protocol is built upon a slight variant of OMDHKE, which we denote by OMDHKE′. The only difference between

OMDHKE and OMDHKE′ is that the latter defines session key *sk* as

$$sk = H(A \parallel S \parallel \bar{X} \parallel Z \parallel K).$$

That is, the OMDHKE′ protocol excludes the password-derived group element PW_A from the input to the key derivation function H . We make this modification to prevent a malicious client from mounting an insider dictionary attack against its partner client in our three-party protocol. OMDHKE′ can be proven secure under the security of OMDHKE, as claimed by Theorem 1.

Theorem 1. The OMDHKE′ protocol is AKE-secure as long as the OMDHKE protocol is AKE-secure. *Proof.* Assume an adversary $\mathcal{A}'$ who breaks the AKE security of OMDHKE′ . Then we prove the theorem by showing that an adversary $\mathcal{A}$ who breaks the AKE security of OMDHKE can be constructed from the adversary $\mathcal{A}'$. The proof idea is to convert an ability of breaking the AKE security of OMDHKE′ into an ability of mounting an offline dictionary attack against OMDHKE.

$\mathcal{A}$ runs $\mathcal{A}'$ as a subroutine while simulating all the oracles on its own. $\mathcal{A}$ outputs a random ℓ -bit string in response to each distinct H query while storing the input-output pairs of H into a list, which we denote as HList. For Execute / Send / Corrupt / G / F queries of $\mathcal{A}'$, $\mathcal{A}$ answers them by asking the same queries to its own corresponding oracles. Let $U \in C \cup \{S\}$. When $\mathcal{A}'$ asks a Reveal query on an instance $\prod_U^i$ whose session key is expected to be set to $H(C \parallel S \parallel \bar{X} \parallel Z \parallel K)$, $\mathcal{A}$ checks if HList contains an entry of the form $(C \parallel S \parallel \bar{X} \parallel Z \parallel K^*, h)$ for some $K^* \in \mathbb{Z}$. If not, A answers the Reveal query with a random ℓ -bit string. Otherwise, $\mathcal{A}$ mounts the following dictionary attack in an attempt to find out the password pw_C :

An offline dictionary attack against OMDHKE:

STEP 1. $\mathcal{A}$ asks the Reveal($\prod_U^i$) query and receives in return the session key sk_U^i computed as $sk_U^i = H(C \parallel S \parallel \bar{X} \parallel Z \parallel PW_C \parallel K)$.

STEP 2. $\mathcal{A}$ makes a guess pw'_C on the password pw_C and computes $PW'_C = G(pw'_C)$ and $sk'^i_U = H(C \parallel S \parallel \bar{X} \parallel Z \parallel PW'_C \parallel K^*)$.

STEP 3. $\mathcal{A}$ verifies the correctness of pw'_C by comparing sk'^i_U against sk_U^i. With an overwhelming probability, sk'^i_U is equal to sk_U^i if and only if pw'_C and pw_C are equal.

STEP 4. $\mathcal{A}$ repeats Steps 2 & 3 for all guesses pw'_C and for all entries $(C \parallel S \parallel \bar{X} \parallel Z \parallel K^*, h)$ until a match is found.

If A fails to discover the password pw_C, it returns a random ℓ -bit string in response to the Reveal query. Otherwise, A can perfectly impersonate client C against server S , or vice versa, using the discovered password pw_C and therefore, can break the AKE security of OMDHKE with advantage 1.

When $\mathcal{A}'$ asks a Test query, $\mathcal{A}$ responds with a random ℓ -bit string. Let $\prod_U^i$ denote any instance whose session key is expected to be set to $H(C \parallel S \parallel \bar{X} \parallel Z \parallel K)$. Let Ask be the event that $\mathcal{A}'$ makes the $H(C \parallel S \parallel \bar{X} \parallel Z \parallel K)$ query when $\prod_U^i$ is one of the instances asked a Test query. Since H is a random oracle, $\mathcal{A}'$ cannot gain any advantage in attacking OMDHKE′ if Ask does not occur. When $\mathcal{A}'$ terminates and outputs its guess b' , $\mathcal{A}$ mounts the above-described offline dictionary attack assuming the occurrence of Ask and thereby, breaks the AKE security of OMDHKE with advantage 1. This concludes the proof of Theorem 1.

THE PROPOSED PROTOCOL

We now present an efficient three-party PAKE protocol (e3PAKE) which is built upon the two-party OMDHKE protocol. The e3PAKE protocol is not only AKE-secure but also provides resistance to UDOD attacks. Compared with OMDHKE, e3PAKE requires only 1 additional communication round where two client messages are sent in parallel solely to achieve security against UDOD attacks.

A. Preliminaries

We begin with the cryptographic primitives which underlie the security of e3PAKE.

Decisional Diffie-Hellman (DDH) assumption. Consider a cyclic group $\mathbb{G}$ having prime order q . Informally stated, the DDH problem for G is to distinguish between two distributions (g^x , g^y , g^{xy}) and (g^x , g^y , g^z), where g is a random generator of $\mathbb{G}$ and x, y, z are chosen at random from $\mathbb{Z}_q^*$. We say that the DDH assumption holds in $\mathbb{G}$ if it is computationally infeasible to solve the DDH problem for $\mathbb{G}$. More formally, we define the advantage of an algorithm $\mathcal{D}$ in solving the DDH problem for $\mathbb{G}$ as $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\mathcal{D}) = |\Pr[\mathcal{D}(\mathbb{G}, g, g^x, g^y, g^{xy}) = 1] - \Pr[\mathcal{D}(\mathbb{G}, g, g^x, g^y, g^z) = 1]|$. We say that the DDH assumption holds in $\mathbb{G}$ if $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\mathcal{D})$ is negligible for all PPT algorithms $\mathcal{D}$. $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(t)$ denotes the maximum value of $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{DDH}}(\mathcal{D})$ over all algorithms $\mathcal{D}$ running in time at most t . A standard way of generating $\mathbb{G}$ where the DDH assumption is assumed to hold is to choose two primes p, q such that $p = rq + 1$ for some small $r \in \mathbb{N}$ (e.g., $r = 2$) and let $\mathbb{G}$ be the subgroup of order q in $\mathbb{Z}_p^*$.

Message authentication codes. A message authentication code (MAC) scheme Σ is a triple of efficient algorithms (Gen, Mac, Ver) where: (1) the key generation algorithm Gen takes

as input a security parameter 1^{ℓ} and outputs a key k chosen uniformly at random from $\{0,1\}^{\ell}$; (2) the MAC generation algorithm Mac takes as input a key k and a message m , and outputs a MAC (also known as a tag) σ ; and (3) the MAC verification algorithm Ver takes as input a key k , a message m , and a MAC σ , and outputs 1if σ is valid for m under k or outputs 0 if σ is invalid. Let $\text{Adv}_{\Sigma}^{\text{SUF-CMA}}(\mathcal{A})$ be the advantage of an adversary $\mathcal{A}$ in violating the strong existential unforgeability of Σ under an adaptive chosen message attack. More precisely, $\text{Adv}_{\Sigma}^{\text{SUF-CMA}}(\mathcal{A})$ is the probability that an adversary $\mathcal{A}$, who mounts an adaptive chosen message attack against Σ with oracle access to $\text{Mac}_k(\cdot)$ and $\text{Ver}_k(\cdot)$, outputs a message/tag pair (m, σ) such that: (1) $\text{Ver}_k(m, \sigma) = 1$ and (2) σ was not previously output by the oracle $\text{Mac}_k(\cdot)$ as a MAC on the message m . We say that the MAC scheme Σ is secure if $\text{Adv}_{\Sigma}^{\text{SUF-CMA}}(\mathcal{A})$ is negligible for every PPT adversary . Let $\text{Adv}_{\Sigma}^{\text{SUF-CMA}}(t, q_{mac}, q_{ver})$ denotes the maximum value of $\text{Adv}_{\Sigma}^{\text{SUF-CMA}}(\mathcal{A})$ over all adversaries $\mathcal{A}$ running in time at most t and asking at most q_{mac} and q_{ver} queries to $\text{Mac}_k(\cdot)$ and $\text{Ver}_k(\cdot)$ respectively.

Symmetric encryption schemes. A symmetric encryption scheme Ω is a triple of efficient algorithms (Gen , Enc , Dec) where: (1) the key generation algorithm Gen takes as input a security parameter 1^{ℓ} and outputs a key k chosen uniformly at random from $\{0,1\}^{\ell}$; (2) the encryption algorithm Enc takes as input a key k and a plaintext message m , and outputs a ciphertext c ; and (3) the decryption algorithm Dec takes as input a key k and a ciphertext c , and outputs a message m . We require that $\text{Dec}_k(\text{Enc}_k(m)) = m$ holds for all $k \in \{0,1\}^{\ell}$ and all $m \in M$, where M is the plaintext space. For an eavesdropping adversary $\mathcal{A}$ against Ω and for a random bit $b \in_R \{0,1\}$, consider the following indistinguishability experiment:

$\text{Experiment}\text{Exp}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A}, b)$

$k \leftarrow \text{Gen}(1^{\ell})$

$(m_0, m_1) \leftarrow \mathcal{A}(\Omega)$, where $|m_0| = |m_1|$

$c \leftarrow \text{Enc}_k(m_b)$

$b' \leftarrow \mathcal{A}(c)$, where $b' \in \{0,1\}$

return b'

For simplicity, we assume, in this experiment, that the security parameter 1^{ℓ} is implicit in the description of Ω . Let $\text{Adv}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A})$ be the advantage of a single eavesdropper $\mathcal{A}$ in breaking the indistinguishability of Ω , and let it be defined as

$$\text{Adv}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A}) = |\Pr[\mathbf{Exp}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A}, 0) = 1] - \Pr[\mathbf{Exp}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A}, 1) = 1]|$$

We say that the encryption scheme Ω is secure (with respect to a single encryption) if $\text{Adv}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A})$ is negligible for every PPT adversary . We use $\text{Adv}_{\Omega}^{\text{IND-SEAV}}(t)$ to denote the maximum value of $\text{Adv}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A})$ over all adversaries $\mathcal{A}$ running in time at most t.

We now claim that if a symmetric encryption scheme is secure with respect to a single encryption, then it is also secure with respect to multiple encryptions under different keys. For an integer $n \geq 1$, consider the indistinguishability experiment below:

$\text{Experiment}\text{Exp}_{\Omega}^{\text{IND-SEAV}}(\mathcal{A}, b, n)$

For i=1 **to** n

$k \leftarrow \text{Gen}(1^{\ell})$

$(m_0, m_1) \leftarrow \mathcal{A}(\Omega)$, where $|m_{0,i}| = |m_{1,i}|$

$c_i \leftarrow \text{Enc}_{ki}(m_{b,i})$

$\mathcal{A}(c_i)$,

$b' \leftarrow \mathcal{A}$, where $b' \in \{0,1\}$

return b'

Then we define $\text{Adv}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A})$ and $\text{Adv}_{\Omega}^{\text{IND-MEAV}}(t)$ respectively as

$$\text{Adv}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A}) = |\Pr[\mathbf{Exp}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A}, 0, n) = 1] - \Pr[\mathbf{Exp}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A}, 1, n) = 1]|$$

And

$$\text{Adv}_{\Omega}^{\text{IND-MEAV}}(t) = \max_{\mathcal{A}} \{\text{Adv}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A})\}$$

where the maximum is over all $\mathcal{A}$ running in time at most t.

Lemma 1. *For any symmetric encryption scheme* Ω

$$\text{Adv}_{\Omega}^{\text{IND-MEAV}}(t) \leq n \cdot \text{Adv}_{\Omega}^{\text{IND-SEAV}}(t)$$

where n is as defined for experiment $\mathbf{Exp}_{\Omega}^{\text{IND-MEAV}}(\mathcal{A}, b, n)$.

Cryptographic hash functions. Additionally, e3PAKE uses three cryptographic hash functions $F: \{0,1\}^* \rightarrow \{0,1\}^{\ell}$, $G: \{0,1\}^* \rightarrow \mathbb{G}$, and $H: \{0,1\}^* \rightarrow \{0,1\}^{\ell}$ These hash functions are modelled as random oracles in our proof of security for e3PAKE.

B. Description of e3PAKE

Let A and B be two clients who wish to establish a session key, and S be the trusted server with which A and B have registered their passwords pw_A and pw_B respectively. We assume that the following information has been pre-established and is known to all parties in the network: (1) a cyclic group $\mathbb{G}$ of prime order q, and a generator g of $\mathbb{G}$, (2) a MAC scheme Σ = (Gen, Mac, Ver), (3) a symmetric encryption scheme Ω = (Gen, Enc, Dec), and (4) three hash functions F, G and H. These public parameters can be determined by the server S and broadcast to all registered clients. The e3PAKE protocol is depicted in Fig. 2 and its description is as follows:

ROUND 1. A sets $pid_A = (A, B, S)$, selects a random $x \in \mathbb{Z}_q^*$, computes $\bar{X} = X \cdot PW_A$, where $X = g^x$ and $PW_A = G(pw_A)$, and then sends $\langle A, \bar{X}, pid_A \rangle$ to S. Meanwhile, B sets $pid_B = (A, B, S)$, selects a random $y \in \mathbb{Z}_q^*$, computes $\bar{Y} = Y \cdot PW_B$, where $Y = g^y$ and $PW_B = G(PW_B)$, and sends $\langle B, \bar{Y}, pid_B \rangle$ to S.

ROUND 2. S verifies that pid_A is equal to pid_B. If the verification fails, S aborts the protocol. Otherwise, S sets $pid_S = pid_A$, selects two random $z \in \mathbb{Z}_q^*$, and $\bar{z} \in \{0,1\}^{|q|/2}$, computes

$$Z = g^z,$$

$$PW_A = G(pw_A), PW_B = \mathrm{G}(pw_B),$$

$$X = \bar{X}/PW_A, Y = \bar{Y}/PW_A,$$

$$K_A = X^Z, K_B = Y^Z$$

$$k_A^{mac} = H(A||S||\bar{X}||\mathrm{Z}||K_A), k_B^{mac} = H(B||S||\bar{Y}||\mathrm{Z}||K_B)$$

$$k_A^{enc} = F(k_A^{mac}), k_B^{enc} = F(k_B^{mac})$$

$$\bar{K}_A = K_A{}^{\bar{z}}, \bar{K}_B = K_B{}^{\bar{z}}$$

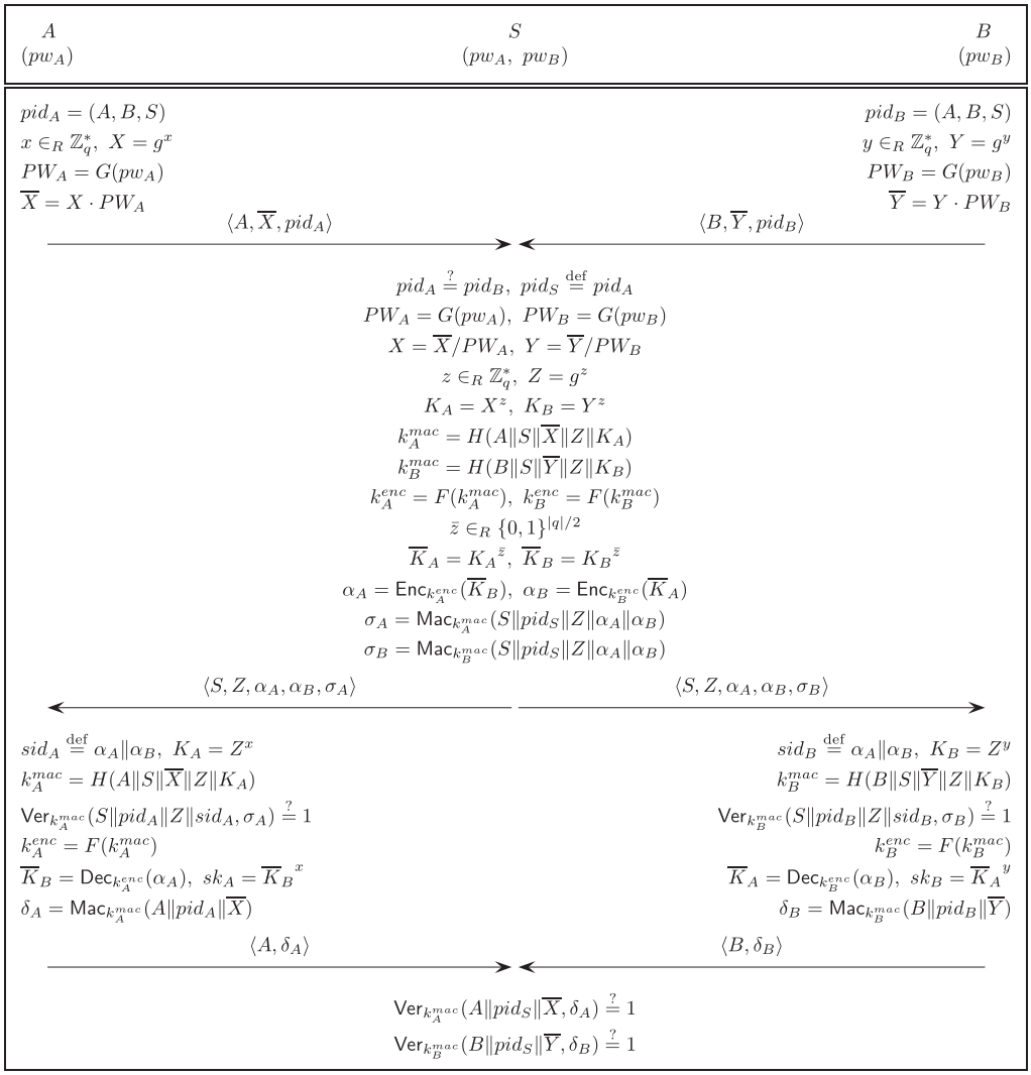

Figure 2. e3PAKE: Our proposed three-party PAKE protocol.

$$\alpha_A = \mathrm{Enc}_{k_A^{mac}}(\bar{K}_B)\ ,\ \alpha_B = \mathrm{Enc}_{k_B^{mac}}(\bar{K}_A)$$

$$\sigma_A = Mac_{k_A^{mac}}(S||pid_S||Z||\alpha_A||\alpha_B),\ \sigma_B = Mac_{k_B^{mac}}(S||pid_S||Z||\alpha_A||\alpha_B)$$

and sends $\langle S, Z, \alpha_A, \alpha_B, \sigma_A \rangle$ and $\langle S, Z, \alpha_A, \alpha_B, \sigma_B \rangle$ to A and B, respectively.

ROUND 3. A sets $sid_A = a_A \parallel a_B$, computes

$$\bar{K}_A = X^z,$$
$$k_A^{mac} = H(A||S||\bar{X}||Z||K_A),$$
$$k_A^{enc} = F(k_A^{mac}),$$

and verifies that $\mathrm{Ver}k_A^{mac}(S \parallel pid_A \parallel Z \parallel sid_A, \sigma_A) = 1$. If the verification fails, A aborts the protocol. Otherwise, A computes the session key $sk_A = \bar{K}_B^{\,x}$, where $\bar{K}_B = \mathrm{Dec}k_A^{enc}(\alpha_A)$, and sends the authenticator $\delta_A = \mathrm{Mac}k_A^{mac}(A \parallel pid_A \parallel \bar{X})$ to S. B proceeds correspondingly; it sets $sid_B = a_A \parallel a_B$, computes

$$\bar{K}_B = Z^y,$$
$$k_B^{mac} = H(B||S||\bar{X}||Z||K_B),$$
$$k_B^{enc} = F(k_B^{mac}),$$

and checks if $\mathrm{Ver}k_B^{mac}(S \parallel pid_B \parallel Z \parallel sid_B, \sigma_B) = 1$. B aborts if the check fails. Otherwise, B computes $sk_B = \bar{K}_A^{\,y}$, where $\bar{K}_A = \mathrm{Dec}k_B^{enc}(\alpha_B)$, and sends $\delta_B = \mathrm{Mac}k_A^{mac}(B \parallel pid_B \parallel \bar{Y})$ to S. Upon receiving δ_A = and δ_B, S aborts if either of these authenticators is invalid.

In the presence of a passive adversary, A and B will compute session keys of the same value $g^{xyz\bar{z}}$, as shown below:

$$\begin{aligned} sk_A &= \bar{K}_B{}^x \\ &= K_B{}^{x\bar{z}} \\ &= g^{xyz\bar{z}} \\ &= X^{yz\bar{z}} \\ &= K_A{}^{y\bar{z}} \\ &= \bar{K}_A{}^y \\ &= sk_B \end{aligned}$$

e3PAKE achieves resistance to UDOD attacks via the MAC values δ_A and δ_B sent in the third round. If key confirmation is required, e3PAKE can be extended to incorporate the well-known technique of Bellare et al. [3]. This extension would require two additional messages to be exchanged (between A and B) in the third round. Exponentiating $\bar{K}_A$ and $\bar{K}_B$ to the power $\bar{z}$ prevents a malicious insider from learning immediately the MAC key of its partner client and thereby from mounting an offline dictionary attack similar to the one presented by Nam et al. [30]. We note that the exponent $\bar{z}$ is of length |q|/2 and thus, both the computations $\bar{K}_A = K_A^{\bar{z}}$ and $\bar{K}_B = K_B^{\bar{z}}$ count as a half exponentiation (i.e., half the number of modular multiplications compared with a full exponentiation).

The hash function $G : \{0,1\}^* \rightarrow \mathbb{G}$ can be constructed from a typical hash function in several ways, as indicated in [27]. For the sake of efficiency, we suggest to construct G by: (1) defining $\mathbb{G}$ as a subgroup of order q in $\mathbb{Z}_p^*$ where p is a safe prime (i.e., $p = 2q + 1$), (2) choosing a hash function G' that outputs elements of $\mathbb{Z}_p^*$ (i.e., G': $\{0,1\}^* \rightarrow \mathbb{Z}_p^*$), and then (3) setting $G(\cdot) = G'(\cdot)^2$. With this method, computing $G(pw)$ would only require about one hash function evaluation and one modular multiplication.

CONCLUSION

This work has presented an efficient protocol for password-only authenticated key exchange (PAKE) in the three-party setting. It is fair to say that in light of the overall cost of communication and computation, our protocol (e3PAKE) performs best among all competing protocols. The third communication round of e3PAKE can be omitted if security against undetectable online dictionary attacks is not desired (see the protocol description in Section 4.2). This simplified protocol would achieve better efficiency and still be AKE-secure in the sense of Definition 2 (i.e., Theorem 2 also holds for the simplified protocol). Moreover, e3PAKE can be easily extended to incorporate the well-known key confirmation technique of Bellare et al. [3] without increasing the number of communication rounds. The proof model we used is the one of Bellare et al. [3] which allows the adversary to ask Corrupt queries. Therefore, our security proof implies that e3PAKE not only achieves forward secrecy but also is secure against dictionary attacks by malicious insiders. Future work includes designing a three-party PAKE protocol that achieves the same (or even better) level of efficiency as e3PAKE but does not rely its security proof on the existence of randomoracles

ACKNOWLEDGMENT

This work is supported by NRF-2017R1A2B1007015.

REFERENCES

[1] M. Abdalla, P. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," IEEProceedings-Information Security, vol.

153, no. 1, pp. 27–39, 2006. An earlier version was presented in PKC 2005.

[2] M. Abdalla and D. Pointcheval, "Interactive Diffie-Hellman assumptions with applications to password-based authentica-tion," Proc. FC 2005, LNCS vol. 3570, pp. 341–356, 2005.

[3] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," Proc.EUROCRYPT 2000, LNCS vol. 1807, pp. 139–155, 2000.

[4] M. Bellare and P. Rogaway, "The AuthA protocol for password-based authenticated key exchange," Contributions to IEEE P1363, 2000.

[5] S. Bellovin and M. Merritt, "Encrypted key exchange: password-based protocols secure against dictionary attacks," Proc.1992 IEEE Symposium on Research in Security and Privacy, pp. 72–84, 1992.

[6] E. Bresson, O. Chevassut, and D. Pointcheval, "New security results on encrypted key exchange," Proc. PKC 2004, LNCS vol. 2947, pp. 145–158, 2004. The full version is available at http://www.di.ens.fr/users/pointche

[7] R. Canetti, D. Dachman-Soled, V. Vaikuntanathan, and H. Wee, "Efficient password authenticated key exchange via oblivious transfer," Proc. PKC 2012, LNCS vol. 7293, pp. 449-466, 2012.

[8] T. Chang, M. Hwang, and W. Yang, "A communication-efficient three-party password authenticated key exchange protocol," Information Sciences, vol. 181, no. 1, pp. 217–226, 2011.

[9] KKR. Choo, C. Boyd, and Y. Hitchcock, "Examining indistinguishability-based proof models for key establishment protocols," Proc. ASIACRYPT 2005, LNCS vol. 3788, pp. 585–604, 2005.

[10] H. Chung and W. Ku, "Three weaknesses in a simple three-party key exchange protocol," Information Sciences, vol. 178, no. 1, pp. 220–229, 2008.

[11] Y. Ding and P. Horster, "Undetectable on-line password guessing attacks," ACM Operating Systems Review, vol. 29, no. 4, pp. 77–86, 1995.

[12] E. Dongna, Q. Cheng, and C. Ma, "Password authenticated key exchange based on RSA in the three-party settings," Proc. ProvSec 2009, LNCS vol. 5848, pp. 168–182, 2009.

[13] S. Goldwasser and S. Micali, "Probabilistic encryption," Journal of Computer and System Sciences, vol. 28, no. 2, pp. 270–299, 1984.

[14] V. Goyal, A. Jain, and R. Ostrovsky, "Password-authenticated session-key generation on the Internet in the plain model," Proc. CRYPTO 2010, LNCS vol. 6223, pp. 277-294, 2010.

[15] H. Guo, Z. Li, Y. Mu, and X. Zhang, "Cryptanalysis of simple three-party key exchange protocol," Computers & Security, vol. 27, no. 1, pp. 16–21, 2008.

[16] H. Huang, "A simple three-party password-based key exchange protocol," International Journal of Communication Systems, vol. 22, no. 7, pp. 857–862, 2009.

[17] J. Katz, R. Ostrovsky, and M. Yung, "Efficient and secure authenticated key exchange using weak passwords," Journal of the ACM, vol. 57, no. 1, article 3, 2009. An earlier version was presented in EUROCRYPT 2001.

[18] J. Katz and V. Vaikuntanathan, "Round-optimal password-based authenticated key exchange," Journal of Cryptology, vol. 26, no. 4, pp. 714–743, 2013. An earlier version was presented in TCC 2011.

[19] H. Kim and J. Choi, "Enhanced password-based simple three-party key exchange protocol," Computers and Electrical Engineering, vol. 35, no. 1, pp. 107–114, 2009.

[20] C. Lee, S. Chen, and C. Chen, "A computation-efficient three-party encrypted key exchange protocol," Applied Mathematics & Information Sciences, vol. 6, no. 3, pp. 573–579, 2012.

[21] T. Lee and T. Hwang, "Simple password-based three-party authenticated key exchange without server public keys," Information Sciences, vol. 180, no. 9, pp. 1702–1714, 2010.

[22] T. Lee, T. Hwang, and C. Lin, "Enhanced three-party encrypted key exchange without server public keys," Computers & Security, vol. 23, no. 7, pp. 571–577, 2004.

[23] H. Liang, J. Hu, and S. Wu, "Re-attack on a three-party password-based authenticated key exchange protocol," Mathematical and Computer Modelling, vol. 57, no. 5–6, pp. 1175–1183, 2013.

[24] C. Lin and T. Hwang, "On 'a simple three-party password-based key exchange protocol'," International Journal of Communication Systems, vol. 24, no. 11, pp. 1520–1532, 2011.

[25] C. Lin, H. Sun, M. Steiner, and T. Hwang, "Three-party encrypted key exchange without server public-keys," IEEE Communications Letters, vol. 5, no. 12, pp. 497–499, 2001.

[26] R. Lu and Z. Cao, "Simple three-party key exchange protocol," Computers & Security, vol. 26, no. 1. pp. 94–97, 2007.

[27] P. MacKenzie, "The PAK suite: Protocols for password-authenticated key exchange," Contributions to IEEE P1363.2, 2002.

[28] J. Nam, KKR. Choo, M. Kim, J. Paik, and D. Won, "Dictionary attacks against password-based authenticated three-party key exchange protocols," KSII Transactions on Internet and Information Systems, vol. 7, no. 12, pp. 3244–3260, 2013.

[29] J. Nam, KKR. Choo, J. Paik, and D. Won, "On the security of a password-only authenticated three-party key exchange protocol," Cryptology ePrint Archive, Report 2013/540, 2013.

[30] J. Nam, KKR. Choo, J. Paik, and D. Won, "An offline dictionary attack against a three-party key exchange protocol," Cryptology ePrint Archive, Report 2013/666, 2013.

[31] J. Nam, KKR. Choo, J. Paik, and D. Won, "Two-round password-only authenticated key exchange in the three-party setting," Cryptology ePrint Archive, Report 2014/017, 2014.

[32] J. Nam, Y. Lee, S. Kim, and D. Won, "Security weakness in a three-party pairing-based protocol for password authenticated key exchange," Information Sciences, vol. 177, no. 6, pp. 1364–1375, 2007.

[33] J. Nam, J. Paik, H. Kang, U. Kim, and D. Won, "An off-line dictionary attack on a simple three-party key exchange protocol," IEEE Communications Letters, vol. 13, no. 3, pp. 205–207, 2009.

[34] R. Phan, W. Yau, and B. Goi, "Cryptanalysis of simple three-party key exchange protocol (S-3PAKE)," Information Sciences, vol. 178, no. 13, pp. 2849–2856, 2008.

[35] M. Steiner, G. Tsudik, and M. Waidner, "Refinement and extension of encrypted key exchange," ACM SIGOPS Operating Systems Review, vol. 29, no. 3, pp. 22–30, 1995.

[36] H. Tsai and C. Chang, "Provably secure three party encrypted key exchange scheme with explicit authentication," Information Sciences, vol. 238, pp. 242–249, 2013.

[37] W. Wang and L. Hu, "Efficient and provably secure generic construction of three-party password-based authenticated key exchange protocols," Proc. INDOCRYPT 2006, LNCS vol. 4329, pp. 118–132, 2006.

[38] W. Wang, L. Hu, and Y. Li, "How to construct secure and efficient three-party password-based authenticated key exchange protocols," Proc. INSCRYPT 2010, LNCS vol. 6584, pp. 218–235, 2011.

[39] S. Wu, K. Chen, Q. Pu, and Y. Zhu, "Cryptanalysis and enhancements of efficient three-party password-based key exchange scheme," International Journal of Communication Systems, vol. 26, no. 5, pp. 674–686, 2013.

[40] S. Wu, Q. Pu, S. Wang, and D. He, "Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol," Information Sciences, vol. 215, pp. 83–96, 2012.

[41] H. Xiong, Y. Chen, Z. Guan, and Z. Chen, "Finding and fixing vulnerabilities in several three-party password authenticated key exchange protocols without server public keys," Information Sciences, vol. 235, pp. 329–340, 2013

[42] J. Yang and T. Cao, "Provably secure three-party password authenticated key exchange protocol in the standard model," Journal of Systems and Software, vol. 85, no. 2, pp. 340–350, 2012.

[43] X. Yi, S. Ling, and H. Wang, "Efficient two-server password-only authenticated key exchange," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 9, pp. 1773–1782, 2013.

[44] E. Yoon and K. Yoo, "Cryptanalysis of a simple three-party password-based key exchange protocol," International Journal of Communication Systems, vol. 24, no. 4, pp. 532–542, 2011.

[45] J. Zhao and D. Gu, "Provably secure three-party password-based authenticated key exchange protocol," Information sciences, vol. 184, no. 1, pp. 310–323, 2012.