Symmetry **2015**, *7*, 105-124; doi:10.3390/sym7010105

OPEN ACCESS
symmetry
ISSN 2073-8994
www.mdpi.com/journal/symmetry

Article

Two-Round Password-Only Authenticated Key Exchange in the Three-Party Setting

Junghyun Nam [1], Kim-Kwang Raymond Choo [2], Sangchul Han [1,*], Juryon Paik [3] and Dongho Won [3]

[1] Department of Computer Engineering, Konkuk University, 268 Chungwondaero, Chungju, Chungcheongbukdo 380-701, Korea; E-Mail: jhnam@kku.ac.kr

[2] Information Assurance Research Group, Advanced Computing Research Centre, University of South Australia, Mawson Lakes, SA-5095, Australia; E-Mail: raymond.choo@unisa.edu.au

[3] Department of Computer Engineering, Sungkyunkwan University, 2066 Seoburo, Suwon, Gyeonggido 440-746, Korea; E-Mails: wise96@skku.edu (J.P.); dhwon@security.re.kr (D.W.)

* Author to whom correspondence should be addressed; E-Mail: schan@kku.ac.kr; Tel.: +82-43-840-3605; Fax: +82-43-840-3600.

Academic Editor: Laurence Yang

Received: 1 November 2014 / Accepted: 16 January 2015 / Published: 27 January 2015

Abstract: We present the first provably-secure three-party password-only authenticated key exchange (PAKE) protocol that can run in only two communication rounds. Our protocol is generic in the sense that it can be constructed from any two-party PAKE protocol. The protocol is proven secure in a variant of the widely-accepted model of Bellare, Pointcheval and Rogaway (2000) without any idealized assumptions on the cryptographic primitives used. We also investigate the security of the two-round, three-party PAKE protocol of Wang, Hu and Li (2010) and demonstrate that this protocol cannot achieve implicit key authentication in the presence of an active adversary.

Keywords: password-only authenticated key exchange (PAKE); three-party key exchange; symmetric encryption; communication round; dictionary attack; implicit key authentication

---

1. Introduction

Protocols for password-only authenticated key exchange (PAKE) enable two or more parties to generate a shared, cryptographically-strong key (called a session key) from their easy-to-remember passwords. PAKE protocols are increasingly popular, and perhaps, due to the popularity of passwords, as explained by Herley and van Oorschot, "despite countless attempts to dislodge passwords (in the past 20 years), they are more widely used and firmly entrenched than ever" [1]. There has been an enormous amount of research effort expended on the design and analysis of PAKE protocols, and yet, there are still worthwhile contributions to be made even in the simple scenario of two protocol participants (also known as clients) with an online trusted server. In such a three-party model, the server provides its registered clients with a centralized authentication service, which allows each client to remember and manage only a single password. Password guessing attacks (also known as dictionary attacks) present a more subtle threat in the three-party model (compared to a two-party model), as a malicious client can attempt to mount such an attack against another client; see [2–6].

It is generally considered that the design of secure, yet efficient key exchange protocols (including password-based protocols) is notoriously difficult, and performing a security analysis for such protocols is complicated and error-prone; see, e.g., [7–9]. The many vulnerabilities identified in published protocols have highlighted the importance of rigorous security proofs in a well-defined formal model. In the provable-security paradigm for key exchange protocols, a reductionist proof approach is adopted to show that an efficient algorithm for breaking the protocol implies an efficient algorithm for solving another problem believed to be (computationally) hard. A complete mathematical proof under a well-established cryptographic assumption offers a strong assurance to protocol implementers that the protocol in hand achieves the desired security properties. The provable-security paradigm for key exchange protocols was made popular by Bellare and Rogaway [10], who introduced the first formal model of adversary capabilities with an associated definition of session-key security. Since then, it has been standard practice for protocol designers to provide proofs of security for their protocols in a widely-accepted security model.

A number of three-party PAKE protocols have been proposed over the last decade [2,3,5,6,11–25]. Many of these protocols have never been proven secure in any model [3,13,17–21] and/or have been found to be vulnerable to some attack(s) [2,3,5,6,8,18–20,23,26–32]. Some protocols [2,11,12,15,23,24] have been proven secure only in a restricted model, in which the adversary is not allowed to corrupt protocol participants, and thus, no attacks by malicious clients can be captured.

Reducing the number of communication rounds is an important practical consideration in designing key exchange protocols. Adopting the usual convention in the three-party (and multi-party) setting, we let a round consist of all protocol messages that can be sent in parallel; note that messages in the same round cannot be dependent on one another. So far, there have been several two-round key exchange protocols presented in the three-party setting.

- The protocols of [15,24] are the only two-round, three-party PAKE protocols published with a claimed security proof (although the two protocols presented by Lee and Hwang [23] can run in two rounds (without key confirmation), and they are insecure in the presence of a malicious client [31]; both protocols are susceptible to a man-in-the-middle attack, as well as an offline dictionary attack). However, it was later found that both protocols are not secure against an active adversary, and their associated claims of provable security are invalid (see [2,8,32,33] and Section 3 of this paper).

- The protocols of [34,35] were proven secure and require only two rounds, but these protocols assume a "hybrid" three-party setting where a server's public key is required in addition to passwords.
- The recent protocol of Tsai and Chang [30] can run in two rounds (without key confirmation), but this protocol only works in a hybrid setting that requires both a cryptographic key and a password pre-established between each client and the server (see [4,29,36–44] for other protocols designed to work in a hybrid setting).

Table 1 summarizes the security properties and known weaknesses of published two-round three-party PAKE protocols with (claimed) proofs of security. To the best of our knowledge, there exists no (provably) secure three-party PAKE protocol running in only two rounds.

Table 1. A summary of security results for existing two-round, three-party PAKE protocols.

Protocol	**Major Weaknesses**	**Communication Model**	**Security Proof**
3PAKE [15]	Vulnerable to an offline dictionary attack [32]	The adversary is restricted from corrupting protocol participants	Based on an invalid assumption [33]
NWPAKE-2 [24]	Fails to achieve implicit key authentication (see Section 3)		Invalidated by an active attack (see Section 3)
S-IA-3PAKE, S-EA-3PAKE [23]	Vulnerable to an offline dictionary attack and a man-in-the-middle attack [31]		Invalidated by a passive attack (see Section 3.3 of [31])

We regard our contributions of this paper to be two-fold:

1. We present the first two-round, three-party PAKE protocol that is provably secure in a well-defined communication model; see Section 4. The communication model in which we work allows the adversary to corrupt protocol participants and, therefore, captures not only the notion of forward secrecy, but also attacks by malicious clients. We make no idealizing assumptions in our security proof. Similar to the protocols of [2,11,12,19,24], our protocol is generic in the sense that it can be constructed from any two-party PAKE protocol. If the underlying two-party protocol is round-optimal [45–47], then our three-party protocol runs in only two communication rounds.
2. We also present a previously unpublished flaw in an existing two-round, three-party PAKE protocol proposed by Wang, Hu and Li [24]; see Section 3.2. The Wang–Hu–Li protocol (named NWPAKE-2) was claimed to be provably secure in a variant of the Real-Or-Random (ROR) model. We reveal that the NWPAKE-2 protocol fails to achieve implicit key authentication in the presence of an active adversary who is not even registered with the server, which invalidates the "claimed" security proof.

The remainder of this paper is structured as follows: Section 2 describes a communication model along with the associated security definition. In Section 3, we revisit the NWPAKE-2 protocol of Wang, Hu and Li [24] and reveal a previously unpublished flaw in the protocol. We then present our proposed two-round, three-party PAKE protocol and prove its security in Section 4. The last section concludes the paper.

2. The Communication Model

We now describe a communication model adapted from the widely-accepted indistinguishability-based model of Bellare, Pointcheval and Rogaway [45]. This will be the model that is used to prove the security of our proposed three-party PAKE protocol.

2.1. Participants and Long-Term Keys

Let S be a trusted authentication server and $\mathcal{C}$ the set of all clients registered with S. During registration, each client $C \in \mathcal{C}$ selects a password pw_C from dictionary $\mathcal{D}$ and shares pw_C with S via a secure/authenticated channel. The password pw_C is used as the long-term secret key between C and S. Any two clients $C, C' \in \mathcal{C}$ may run a three-party PAKE protocol π 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, a single user could have many instances of it at a point of time. We denote instance i of user U by Π^i_U. We say that a client instance Π^i_C accepts when it successfully computes its session key sk^i_C in an execution of the protocol.

2.2. Partnering

Intuitively, two instances are partners if they participate in a protocol execution and establish a (shared) session key. Formally, partnering between instances is defined in terms of the notions of session identifiers and partner identifiers (see [48] on the role and the possible construct of session and partner identifiers as a form of partnering mechanism that enables the right session key to be identified in concurrent protocol executions). A session identifier (sid) is a string that uniquely identifies a protocol session and is usually defined as a function of the messages transmitted in the session. Let sid^i_U denote the sid of instance Π^i_U. 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^i_U denotes the pid given to instance Π^i_U. In a typical session, there will be three participants, namely two clients C and C' and the server S. We say that two instances Π^i_C and $\Pi^j_{C'}$ are partners if all of the following conditions are satisfied: (1) both Π^i_C and $\Pi^j_{C'}$ have accepted; (2) $sid^i_C = sid^j_{C'}$; and (3) $pid^i_C = pid^j_{C'}$.

2.3. Adversary Capabilities

The probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ is in complete control of all communications between users, and its capabilities are modeled via a pre-defined set of oracle queries, as described below.

- Execute($\Pi^i_C, \Pi^j_{C'}, \Pi^k_S$): This query models passive attacks against the protocol. It prompts an execution of the protocol between the instances Π^i_C, $\Pi^j_{C'}$ and Π^k_S and returns the transcript of the protocol execution to $\mathcal{A}$.

- Send(Π^i_U, m): This query sends a message m to instance Π^i_U, modeling active attacks against the protocol. Upon receiving m, the instance Π^i_U proceeds according to the protocol specification. The message output by Π^i_U, if any, is returned to $\mathcal{A}$. A query of the form Send(Π^i_C, `start`:(C, C', S)) prompts Π^i_C to initiate the protocol with $pid^i_C = (C, C', S)$.
- Reveal(Π^i_C): This query captures the notion of known key security (it is often reasonable to suppose that the adversary can obtain session keys from any sessions other than the one under attack) and, if Π^i_C has accepted, returns the session key sk^i_C 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(Π^i_C): This query is used to define the indistinguishability-based security of the protocol. If Π^i_C has accepted, then depending on a randomly-chosen bit b, $\mathcal{A}$ is given either the real session key sk^i_C (when $b = 1$) or a random key drawn from the session-key space (when $b = 0$). $\mathcal{A}$ 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. However, we require that for each different set of partners, $\mathcal{A}$ should access the test oracle only once.

Although Execute and Reveal oracles can be simulated by accessing Send and Test oracles, respectively, multiple times, the former (*i.e.*, Execute and Reveal oracles) often makes it easier to prove the security of protocols and to understand the proofs, and for this reason, we allow both Execute and Reveal queries in our model. The number of queries asked by an adversary is referred to as the query complexity of the adversary (Q) and is represented as an ordered sequence of five non-negative integers, $Q = (q_{\text{exec}}, q_{\text{send}}, q_{\text{reve}}, q_{\text{corr}}, q_{\text{test}})$. These five non-negative integers are the numbers of queries that the adversary asked, respectively, of the Execute, Send, Reveal, Corrupt and Test oracles.

2.4. Security Definition

We define the security of a three-party PAKE protocol via the notion of freshness. Intuitively, a fresh instance is one that holds a session key that should not be known to the adversary $\mathcal{A}$, and an unfresh instance is one whose session key (or some information about the key) can be known by trivial means. The formal definition of freshness is explained in Definition 1.

Definition 1. *An instance Π^i_C is fresh if none of the following occurs: (1) $\mathcal{A}$ queries* Reveal(Π^i_C) *or* Reveal($\Pi^j_{C'}$), *where $\Pi^j_{C'}$ is the partner of Π^i_C; and (2) $\mathcal{A}$ queries* Corrupt(U), *for some $U \in pid^i_C$, before Π^i_C or its partner $\Pi^j_{C'}$ accepts.*

The security of a three-party PAKE protocol π is defined in the context of the following experiment:

Experiment $\mathbf{Exp}_0$:

Phase 1. $\mathcal{A}$ makes any oracle queries at will as many times as it wishes, except that:

- $\mathcal{A}$ is not allowed to ask the Test(Π^i_C) query if the instance Π^i_C is unfresh.

- $\mathcal{A}$ is not allowed to ask the $\mathsf{Reveal}(\Pi^i_C)$ query if it has already made a Test query to Π^i_C or $\Pi^j_{C'}$, where $\Pi^j_{C'}$ is the partner of Π^i_C.

Phase 2. Once $\mathcal{A}$ decides that Phase 1 is over, it outputs a bit b' as a guess on the hidden bit b chosen by the Test oracle. $\mathcal{A}$ is said to succeed if $b = b'$.

Let Succ_0 be the event that $\mathcal{A}$ succeeds in the experiment $\mathbf{Exp}_0$. The advantage of $\mathcal{A}$ in breaking the security of the authenticated key exchange protocol π is $\mathsf{Adv}^{\text{ake}}_\pi(\mathcal{A}) = 2 \cdot \mathsf{Pr}_{\pi,\mathcal{A}}[\mathsf{Succ}_0] - 1$.

Definition 2. *A three-party PAKE protocol π is* AKE*-secure if, for any* PPT *adversary $\mathcal{A}$ asking at most q_{send} Send queries, $\mathsf{Adv}^{\text{ake}}_\pi(\mathcal{A})$ is only negligibly larger than $c \cdot q_{\text{send}}/|\mathcal{D}|$, where c is a constant.*

To represent the security of protocol π in terms of the amount of resources used by adversaries, we let $\mathsf{Adv}^{\text{ake}}_\pi(t, Q)$ be defined as:

$$\mathsf{Adv}^{\text{ake}}_\pi(t, Q) = \max_{\mathcal{A}} \{\mathsf{Adv}^{\text{ake}}_\pi(\mathcal{A})\}$$

where the maximum is taken over all PPT adversaries $\mathcal{A}$ with time complexity at most t and query complexity at most Q.

3. Revisiting Wang, Hu and Li's (2010) NWPAKE-2 Protocol

Implicit key authentication is among the fundamental security properties that should be achieved by key exchange protocols. In this section, we show that the NWPAKE-2 protocol of Wang, Hu and Li [24] does not achieve implicit key authentication.

3.1. Protocol Description

Assume two clients A and B who want to establish a session key. Let S be the trusted server with which A and B shared their passwords pw_A and pw_B, respectively. The public parameters of the NWPAKE-2 protocol include: (1) a cyclic group $\mathbb{G}$ of prime order q and a generator g of $\mathbb{G}$; (2) a two-party PAKE protocol, 2PAKE; and (3) a pair of message authentication code (MAC) generation/verification algorithms $(\mathsf{Mac}, \mathsf{Ver})$, where Ver outputs a bit, with 1 meaning `accept` and 0 meaning `reject`. If the underlying two-party protocol, 2PAKE, is round-optimal, NWPAKE-2 completes in two communication rounds, as depicted in Figure 1. The protocol description is as follows:

Step 1. A and S establish a secret key k_A by running the two-party protocol, 2PAKE. Likewise, B and S establish a secret key k_B.

Step 2. A (resp. B) selects a random $x \in \mathbb{Z}^*_q$ (resp. $y \in \mathbb{Z}^*_q$) and sends $X = g^x$ (resp. $Y = g^y$) to S.

Step 3. S chooses a random $z \in \mathbb{Z}^*_q$, computes:

$$\overline{X} = X^z, \;\; \overline{Y} = Y^z$$

$$\rho_A = \mathsf{Mac}_{k_A}(X \| \overline{Y} \| B \| A), \;\; \rho_B = \mathsf{Mac}_{k_B}(Y \| \overline{X} \| A \| B)$$

and sends $\langle \overline{Y}, \rho_A \rangle$ and $\langle \overline{X}, \rho_B \rangle$ to A and B, respectively.

Step 4. A and B abort if their received MAC is invalid. Otherwise, they will compute their respective session keys, $sk_A = \overline{Y}^x$ and $sk_B = \overline{X}^y$.

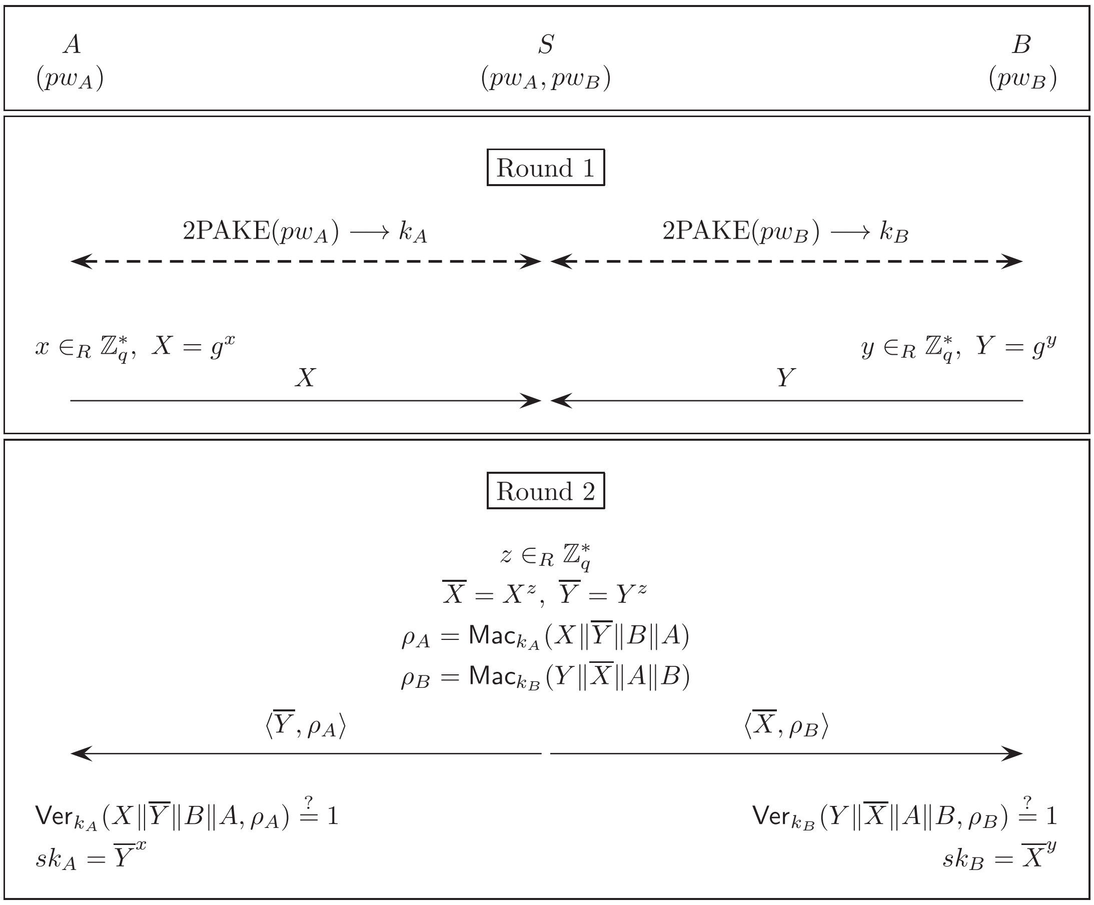
A
(pw_A)
S
(pw_A, pw_B)
B
(pw_B)
Round 1
$\text{2PAKE}(pw_A) \longrightarrow k_A$
$\text{2PAKE}(pw_B) \longrightarrow k_B$
$x \in_R \mathbb{Z}_q^*,\ X = g^x$
$y \in_R \mathbb{Z}_q^*,\ Y = g^y$
X
Y
Round 2
$z \in_R \mathbb{Z}_q^*$
$\overline{X} = X^z,\ \overline{Y} = Y^z$
$\rho_A = \mathsf{Mac}_{k_A}(X \| \overline{Y} \| B \| A)$
$\rho_B = \mathsf{Mac}_{k_B}(Y \| \overline{X} \| A \| B)$
$\langle \overline{Y}, \rho_A \rangle$
$\langle \overline{X}, \rho_B \rangle$
$\mathsf{Ver}_{k_A}(X \| \overline{Y} \| B \| A, \rho_A) \overset{?}{=} 1$
$sk_A = \overline{Y}^x$
$\mathsf{Ver}_{k_B}(Y \| \overline{X} \| A \| B, \rho_B) \overset{?}{=} 1$
$sk_B = \overline{X}^y$

Figure 1. Wang *et al.*'s two-round, three-party PAKE protocol (NWPAKE-2) [24].

At the end of the protocol execution, A and B will compute the same session key $sk_A = sk_B = g^{xyz}$.

3.2. Violating Implicit Key Authentication

We now assume that there exists an adversary C who is not registered with the server and demonstrate how C can easily violate the implicit key authentication property of NWPAKE-2.

1. C chooses a random $x' \in \mathbb{Z}_q^*$, computes $X' = g^{x'}$ and replaces X (sent by A to S) with X'.
2. Upon receipt of the "replaced" message, S will compute $\overline{X}$ as $\overline{X} = X'^z$, and therefore, B's session key sk_B will be set to $g^{x'yz}$.
3. C intercepts the message $\langle \overline{Y}, \rho_A \rangle$ sent by S to A and then computes $sk_C = \overline{Y}^{x'} = g^{x'yz} = sk_B$. In other words, C is able to compute B's session key even though C is not B's partner.

The design flaw exploited by the adversary is that the server S is provided with no means of authenticating the public values X and Y. Note that NWPAKE-2 exhibits a security weakness no matter which protocol is used for the instantiation of 2PAKE. Wang, Hu and Li [24] provide a proof sketch for the security of NWPAKE-2 in a model that allows Send queries. Any protocol proven secure in such a model should be secure against our above attack, and therefore, the security proof (sketch) for NWPAKE-2 is invalidated.

4. Our Proposed Protocol

This section presents our two-round, three-party PAKE protocol, which we denote as 2R3PAKE ("R" is for round) and proves its security in the communication model described in Section 2. The 2R3PAKE protocol can be viewed as a combined variant of the NWPAKE-2 protocol of Wang, Hu and Li [24] and the 3PKD protocol of Bellare and Rogaway [49]. 2R3PAKE is generic in the sense that it can be constructed from any secure two-party PAKE protocol. Our generic construction takes only one round of communication in addition to the number of rounds required to perform the underlying two-party protocol. Hence, applying our construction to a round-optimal two-party PAKE protocol immediately yields a three-party PAKE protocol running in two communication rounds.

4.1. Preliminaries

The security of 2R3PAKE is based on the decisional Diffie–Hellman assumption, the security of a message authentication code scheme, a two-party PAKE protocol, and a symmetric encryption scheme.

4.1.1. Decisional Diffie–Hellman Assumption

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

4.1.2. Message Authentication Codes

A message authentication code (MAC) scheme Σ is a triple of efficient algorithms $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Ver})$, 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 1 if σ is valid for m under k or outputs 0 if σ is invalid. Let $\mathsf{Adv}_{\Sigma}^{\mathrm{suf-cma}}(\mathcal{A})$ be the advantage of an adversary $\mathcal{A}$ in violating the strong existential unforgeability of Σ under an adaptive chosen message attack. More precisely, $\mathsf{Adv}_{\Sigma}^{\mathrm{suf-cma}}(\mathcal{A})$ is the probability that an adversary $\mathcal{A}$, who mounts an adaptive chosen message attack against Σ with oracle access to $\mathsf{Mac}_k(\cdot)$ and $\mathsf{Ver}_k(\cdot)$, outputs a message/tag pair (m, σ), such that: (1)

$\mathsf{Ver}_k(m, \sigma) = 1$; and (2) σ was not previously output by the oracle $\mathsf{Mac}_k(\cdot)$ as a MAC on the message m. We say that the MAC scheme Σ is secure if $\mathsf{Adv}_{\Sigma}^{\mathrm{suf-cma}}(\mathcal{A})$ is negligible for every PPT adversary $\mathcal{A}$. Let $\mathsf{Adv}_{\Sigma}^{\mathrm{suf-cma}}(t, q_{\mathrm{mac}}, q_{\mathrm{ver}})$ denote the maximum value of $\mathsf{Adv}_{\Sigma}^{\mathrm{suf-cma}}(\mathcal{A})$ over all adversaries $\mathcal{A}$ running in time at most t and asking at most q_{mac} and q_{ver} queries to $\mathsf{Mac}_k(\cdot)$ and $\mathsf{Ver}_k(\cdot)$, respectively.

4.1.3. Two-Party PAKE Protocols

2R3PAKE takes as input a two-party PAKE protocol, 2PAKE. We assume that the given two-party protocol, 2PAKE, outputs session keys distributed in $\{0, 1\}^n$, where $n = 2\ell$, and is AKE-secure against an adversary who is given access to all of the oracles: Execute, Send, Reveal, Corrupt and Test. Let $\mathsf{Adv}_{\mathrm{2PAKE}}^{\mathrm{ake}}(\mathcal{A})$ be the advantage of an adversary $\mathcal{A}$ at breaking the AKE security of 2PAKE. We require that, for any PPT adversary $\mathcal{A}$ asking at most q_{send} Send queries, $\mathsf{Adv}_{\mathrm{2PAKE}}^{\mathrm{ake}}(\mathcal{A})$ is only negligibly larger than $q_{\mathrm{send}}/|\mathcal{D}|$. $\mathsf{Adv}_{\mathrm{2PAKE}}^{\mathrm{ake}}(t, Q)$ denotes the maximum value of $\mathsf{Adv}_{\mathrm{2PAKE}}^{\mathrm{ake}}(\mathcal{A})$ over all adversaries $\mathcal{A}$ with time complexity at most t and query complexity at most Q.

4.1.4. 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 plain text 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 $\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m$ holds for all $k \in \{0, 1\}^\ell$ and all $m \in \mathcal{M}$, where $\mathcal{M}$ is the plain text space. For an eavesdropping adversary $\mathcal{A}$ against Ω and for a random bit $b \in_R \{0, 1\}$, consider the following experiment, $\mathbf{Exp}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}, b)$, where "ind-seav" denotes indistinguishability against single eavesdropping:

```
Experiment $\mathbf{Exp}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}, b)$
    $k \leftarrow \mathsf{Gen}(1^\ell)$
    $(m_0, m_1) \leftarrow \mathcal{A}(\Omega)$, where $|m_0| = |m_1|$
    $c \leftarrow \mathsf{Enc}_k(m_b)$
    $b' \leftarrow \mathcal{A}(c)$, 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. Note that in the experiment, the adversary $\mathcal{A}$ generates the plain text pair (m_0, m_1) and is given the ciphertext c, which is the encryption of either m_0 or m_1. Let $\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A})$ be the advantage of an eavesdropper $\mathcal{A}$ in breaking the indistinguishability of Ω, and let it be defined as:

$$\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}) = |\Pr[\mathbf{Exp}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}, 0) = 1] - \Pr[\mathbf{Exp}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}, 1) = 1]|$$

We say that the symmetric encryption scheme Ω is secure (with respect to a single encryption) if $\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A})$ is negligible for every PPT adversary $\mathcal{A}$. We use $\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(t)$ to denote the maximum value of $\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A})$ 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 following experiment, $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}(\mathcal{A}, b, n)$, where "ind-meav" denotes indistinguishability against multiple eavesdropping:

```
Experiment $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}(\mathcal{A}, b, n)$
    for $i = 1$ to $n$
        $k_i \leftarrow \mathsf{Gen}(1^{\ell})$
        $(m_{0,i}, m_{1,i}) \leftarrow \mathcal{A}(\Omega)$, where $|m_{0,i}| = |m_{1,i}|$
        $c_i \leftarrow \mathsf{Enc}_{k_i}(m_{b,i})$
        $\mathcal{A}(c_i)$
    $b' \leftarrow \mathcal{A}$, where $b' \in \{0, 1\}$
    return $b'$
```

Then, we define $\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(\mathcal{A})$ and $\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(t)$, respectively, as:

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

and:

$$\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(t) = \max_{\mathcal{A}} \{\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(\mathcal{A})\}$$

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

Lemma 1. *For any symmetric encryption scheme* Ω*,*

$$\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(t) \leq n \cdot \mathsf{Adv}_{\Omega}^{\text{ind-seav}}(t),$$

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

Proof. Let $\mathcal{A}$ be a multiple eavesdropper attacking the indistinguishability of Ω, with advantage $\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(\mathcal{A})$ and time complexity t. The proof proceeds with a standard hybrid argument [50]. Consider a sequence of $n + 1$ hybrid experiments $\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n)$, $0 \leq \xi \leq n$, where each $\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n)$ is different from $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}(\mathcal{A}, b, n)$ only in that each c_i is set as follows:

$$c_i \leftarrow \begin{cases} \mathsf{Enc}_{k_i}(m_{1,i}) & \text{if } i \leq \xi \\ \mathsf{Enc}_{k_i}(m_{0,i}) & \text{otherwise} \end{cases}$$

The experiments $\mathbf{Exp}_{\Omega,0}^{\text{ind-meav}}(\mathcal{A}, b, n)$ and $\mathbf{Exp}_{\Omega,n}^{\text{ind-meav}}(\mathcal{A}, b, n)$ at the extremes of the sequence are identical to $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}(\mathcal{A}, 0, n)$ and $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}(\mathcal{A}, 1, n)$, respectively. As we move from $\mathbf{Exp}_{\Omega,\xi-1}^{\text{ind-meav}}(\mathcal{A}, b, n)$ to $\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n)$ in the sequence, we change the ξ-th ciphertext c_ξ from the encryption of $m_{0,\xi}$ to the encryption of $m_{1,\xi}$. Since there are n such moves from $\mathbf{Exp}_{\Omega,0}^{\text{ind-meav}}(\mathcal{A}, b, n)$ to $\mathbf{Exp}_{\Omega,n}^{\text{ind-meav}}(\mathcal{A}, b, n)$, the inequality of the lemma follows immediately if we prove that the difference between the probabilities that $\mathcal{A}$ outputs 1 in any two neighboring experiments $\mathbf{Exp}_{\Omega,\xi-1}^{\text{ind-meav}}(\mathcal{A}, b, n)$ and $\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n)$ is at most $\mathsf{Adv}_{\Omega}^{\text{ind-seav}}(t)$. To complete the proof, it suffices to show that for any $1 \leq \xi \leq n$,

$$|\Pr[\mathbf{Exp}_{\Omega,\xi-1}^{\text{ind-meav}}(\mathcal{A}, b, n) = 1] - \Pr[\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n) = 1]| \leq \mathsf{Adv}_{\Omega}^{\text{ind-seav}}(t) \quad (1)$$

Let $\varepsilon = |\Pr[\mathbf{Exp}_{\Omega,\xi-1}^{\text{ind-meav}}(\mathcal{A}, b, n) = 1] - \Pr[\mathbf{Exp}_{\Omega,\xi}^{\text{ind-meav}}(\mathcal{A}, b, n) = 1]|$. We will prove Equation (1) by constructing, from $\mathcal{A}$, a single eavesdropper $\mathcal{A}_\xi$ who breaks the indistinguishability of Ω with advantage ε and time complexity t.

$\mathcal{A}_\xi$ begins by invoking adversary $\mathcal{A}$, then proceeds to simulate the indistinguishability experiment for $\mathcal{A}$ and, finally, ends by outputting whatever bit $\mathcal{A}$ eventually outputs. In the simulated experiment, $\mathcal{A}_\xi$ generates the ciphertexts exactly as in the hybrid experiment $\mathbf{Exp}_{\Omega,\xi}^{\mathrm{ind-meav}}(\mathcal{A}, b, n)$, except that it generates the ξ-th ciphertext c_ξ as follows:

> When $\mathcal{A}$ outputs the ξ-th plain text pair $(m_{0,\xi}, m_{1,\xi})$, $\mathcal{A}_\xi$ outputs this as its own plain text pair in experiment $\mathbf{Exp}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}_\xi, b)$, receives in return a ciphertext c and sets $c_\xi = c$.

It follows that:

- The probability that $\mathcal{A}_\xi$ outputs 1 when the given ciphertext c is the encryption of $m_{0,\xi}$ is equal to the probability that $\mathcal{A}$ outputs 1 in the experiment $\mathbf{Exp}_{\Omega,\xi-1}^{\mathrm{ind-meav}}(\mathcal{A}, b, n)$.
- The probability that $\mathcal{A}_\xi$ outputs 1 when the given ciphertext c is the encryption of $m_{1,\xi}$ is equal to the probability that $\mathcal{A}$ outputs 1 in the experiment $\mathbf{Exp}_{\Omega,\xi}^{\mathrm{ind-meav}}(\mathcal{A}, b, n)$.

This means that:

$$\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}_\xi) = |\Pr[\mathbf{Exp}_{\Omega,\xi-1}^{\mathrm{ind-meav}}(\mathcal{A}, b, n) = 1] - \Pr[\mathbf{Exp}_{\Omega,\xi}^{\mathrm{ind-meav}}(\mathcal{A}, b, n) = 1]|$$

Since $\mathcal{A}_\xi$ has time complexity t, it follows that $\mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(\mathcal{A}_\xi) \leq \mathsf{Adv}_{\Omega}^{\mathrm{ind-seav}}(t)$ by definition. This completes the proof of Equation (1) and, hence, the proof of Lemma 1. □

4.2. The 2R3PAKE Protocol

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 $\Sigma = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Ver})$; (3) a two-party PAKE protocol 2PAKE; and (4) a symmetric encryption scheme $\Omega = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. These public parameters can be determined by the server and broadcast to all registered clients. 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. The partner identifier assigned to (an instance of) A (resp. B) is pid_A (resp. pid_B). Recall that pid is a sequence of identities of protocol participants; for simplicity, we assume that $pid_A = pid_B = (A, B, S)$. Our 2R3PAKE protocol is depicted in Figure 2 and its description is as follows:

Step 1. A (resp. B) selects a random $x \in \mathbb{Z}_q^*$ (resp. $y \in \mathbb{Z}_q^*$), computes $X = g^x$ (resp. $Y = g^y$) and sends $\langle A, pid_A, X\rangle$ (resp. $\langle B, pid_B, Y\rangle$) to S.

Step 2. A and S establish a 2ℓ-bit key k_A by running the two-party protocol, 2PAKE. Likewise, B and S establish a 2ℓ-bit key k_B. Let $k_A = k_A^{enc} \| k_A^{mac}$ and $k_B = k_B^{enc} \| k_B^{mac}$.

Step 3. A computes $\sigma_A = \mathsf{Mac}_{k_A^{mac}}(A \| pid_A \| X)$ and sends $\langle A, \sigma_A\rangle$ to S. Similarly, B computes $\sigma_B = \mathsf{Mac}_{k_B^{mac}}(B \| pid_B \| Y)$ and sends $\langle B, \sigma_B\rangle$ to S.

Step 4. S sets $pid_S = pid_A$, chooses a random $z \in \mathbb{Z}_q^*$ and computes:

$$\overline{X} = X^z, \quad \overline{Y} = Y^z,$$

$$\alpha_A = \mathsf{Enc}_{k_A^{enc}}(\overline{Y}), \quad \alpha_B = \mathsf{Enc}_{k_B^{enc}}(\overline{X}),$$

$$\rho_A = \mathsf{Mac}_{k_A^{mac}}(S \| pid_S \| \alpha_A \| \alpha_B), \quad \rho_B = \mathsf{Mac}_{k_B^{mac}}(S \| pid_S \| \alpha_A \| \alpha_B).$$

S then sends $\langle S, \alpha_A, \alpha_B, \rho_A\rangle$ and $\langle S, \alpha_A, \alpha_B, \rho_B\rangle$ to A and B, respectively.

Step 5. A sets the session identifier, $sid_A = \alpha_A \| \alpha_B$, and verifies that $\mathsf{Ver}_{k_A^{mac}}(S\|\ pid_A \| sid_A, \rho_A) = 1$. If the verification fails, A aborts the protocol. Otherwise, A recovers $\overline{Y}$ as $\overline{Y} = \mathsf{Dec}_{k_A^{enc}}(\alpha_A)$ and computes the session key, $sk_A = \overline{Y}^x$. B proceeds correspondingly; it aborts if $\mathsf{Ver}_{k_B^{mac}}(S\|pid_B\|sid_B, \rho_B) = 0$, where $sid_B = \alpha_A\|\alpha_B$, and, otherwise, computes $\overline{X} = \mathsf{Dec}_{k_B^{enc}}(\alpha_B)$ and $sk_B = \overline{X}^y$.

Step 6. S checks that $\mathsf{Ver}_{k_A^{mac}}(A\|pid_S\|X, \sigma_A) = 1$. If the check fails, S marks this transaction as invalid. For MAC σ_B, S checks its validity in the same way. (To prevent online dictionary attacks (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 server), S may lock out a problematic client after a certain number of invalid transactions.)

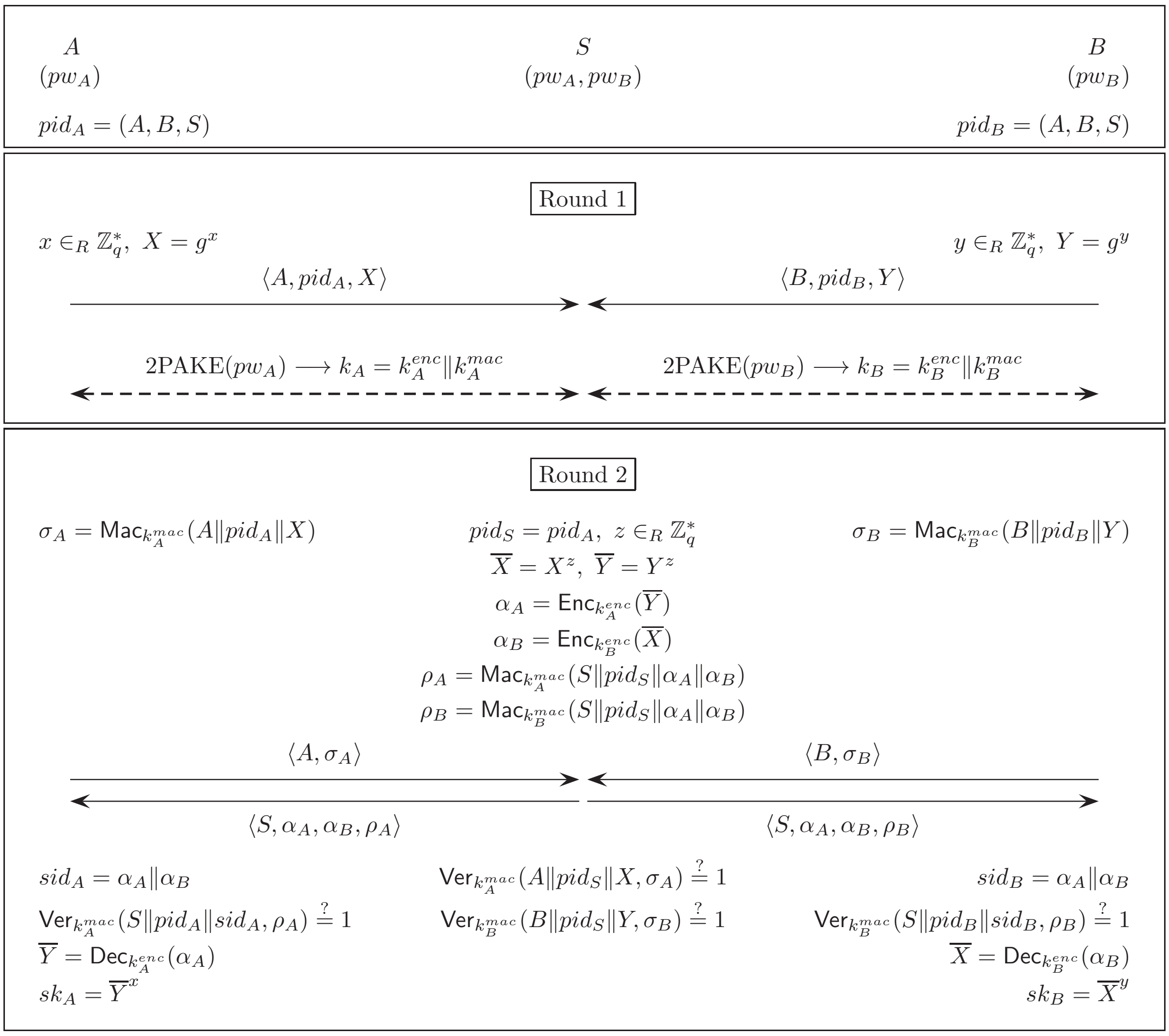
A
(pw_A)
S
(pw_A, pw_B)
B
(pw_B)
$pid_A = (A, B, S)$
$pid_B = (A, B, S)$
Round 1
$x \in_R \mathbb{Z}_q^*,\ X = g^x$
$y \in_R \mathbb{Z}_q^*,\ Y = g^y$
$\langle A, pid_A, X\rangle$
$\langle B, pid_B, Y\rangle$
2PAKE$(pw_A) \longrightarrow k_A = k_A^{enc}\|k_A^{mac}$
2PAKE$(pw_B) \longrightarrow k_B = k_B^{enc}\|k_B^{mac}$
Round 2
$\sigma_A = \mathsf{Mac}_{k_A^{mac}}(A\|pid_A\|X)$
$pid_S = pid_A,\ z \in_R \mathbb{Z}_q^*$
$\overline{X} = X^z,\ \overline{Y} = Y^z$
$\alpha_A = \mathsf{Enc}_{k_A^{enc}}(\overline{Y})$
$\alpha_B = \mathsf{Enc}_{k_B^{enc}}(\overline{X})$
$\rho_A = \mathsf{Mac}_{k_A^{mac}}(S\|pid_S\|\alpha_A\|\alpha_B)$
$\rho_B = \mathsf{Mac}_{k_B^{mac}}(S\|pid_S\|\alpha_A\|\alpha_B)$
$\sigma_B = \mathsf{Mac}_{k_B^{mac}}(B\|pid_B\|Y)$
$\langle A, \sigma_A\rangle$
$\langle B, \sigma_B\rangle$
$\langle S, \alpha_A, \alpha_B, \rho_A\rangle$
$\langle S, \alpha_A, \alpha_B, \rho_B\rangle$
$sid_A = \alpha_A\|\alpha_B$
$\mathsf{Ver}_{k_A^{mac}}(S\|pid_A\|sid_A, \rho_A) \overset{?}{=} 1$
$\overline{Y} = \mathsf{Dec}_{k_A^{enc}}(\alpha_A)$
$sk_A = \overline{Y}^x$
$\mathsf{Ver}_{k_A^{mac}}(A\|pid_S\|X, \sigma_A) \overset{?}{=} 1$
$\mathsf{Ver}_{k_B^{mac}}(B\|pid_S\|Y, \sigma_B) \overset{?}{=} 1$
$sid_B = \alpha_A\|\alpha_B$
$\mathsf{Ver}_{k_B^{mac}}(S\|pid_B\|sid_B, \rho_B) \overset{?}{=} 1$
$\overline{X} = \mathsf{Dec}_{k_B^{enc}}(\alpha_B)$
$sk_B = \overline{X}^y$

Figure 2. Our proposed two-round, three-party PAKE (2R3PAKE) protocol.

Steps 1 and 2 constitute the first round of communication, while Steps 3 and 4 constitute the second communication round. It is trivial to note that in the presence of a passive adversary, A and B will compute session keys of the same value g^{xyz}. We do not require 2PAKE to be instantiated with a protocol that provides either unilateral or mutual authentication, as 2R3PAKE already provides mutual authentication between the server and the clients (via the MAC values exchanged in the second round). Hence, any two-party protocol that provides implicit key authentication, including one-round protocols, will be a suitable candidate to instantiate 2PAKE. We emphasize that sending $\overline{X}$ and $\overline{Y}$ in an encrypted form is necessary for the security of the protocol, as evidenced by the flaws discovered in previous two-round PAKE protocols (see Section 3 and [32]).

4.3. Security Proof

Theorem 1. *For any adversary who has time complexity at most t and query complexity at most $Q = (q_{\text{exec}}, q_{\text{send}}, q_{\text{reve}}, q_{\text{corr}}, q_{\text{test}})$, its advantage in breaking the* AKE *security of 2R3PAKE is bounded by:*

$$\begin{aligned}\mathsf{Adv}^{\text{ake}}_{\text{2R3PAKE}}(t, Q) \leq \;& 2 \cdot \mathsf{Adv}^{\text{ake}}_{\text{2PAKE}}(t', Q') \\ &+ 2 \cdot q_{\text{send}} \cdot \mathsf{Adv}^{\text{suf}-\text{cma}}_{\Sigma}(t', 2, 2) \\ &+ 2 \cdot q_{\text{send}} \cdot \mathsf{Adv}^{\text{ind}-\text{seav}}_{\Omega}(t') \\ &+ 2 \cdot \mathsf{Adv}^{\text{ddh}}_{\mathbb{G}}(t')\end{aligned}$$

where $Q' = (2q_{\text{exec}}, q_{\text{send}}, q_{\text{send}}, q_{\text{corr}}, 2q_{\text{exec}} + q_{\text{send}})$ and t' is the maximum time required to perform the experiment $\mathbf{Exp}_0$ *involving an adversary who attacks 2R3PAKE with time complexity t.*

Proof. Let $\mathcal{A}$ be a PPT adversary who attacks the AKE security of 2R3PAKE with time complexity t and query complexity $Q = (q_{\text{exec}}, q_{\text{send}}, q_{\text{reve}}, q_{\text{corr}}, q_{\text{test}})$. We prove the theorem by making a series of modifications to the experiment $\mathbf{Exp}_0$, bounding the difference in $\mathcal{A}$'s success probability between two consecutive (modified) experiments and ending up with an experiment in which $\mathcal{A}$ has a success probability of 1/2 (*i.e.*, $\mathcal{A}$ has no advantage). By Succ_i, we denote the event that $\mathcal{A}$ correctly guesses the hidden bit b in experiment $\mathbf{Exp}_i$.

Before presenting the first modified experiment, we define the notion of a clean instance.

Definition 3. *We say an instance Π^i_U is unclean if $\mathcal{A}$ has queried* $\mathsf{Corrupt}(U')$ *for some $U' \in pid^i_U$. Otherwise, we say it is clean.*

Experiment $\mathbf{Exp}_1$. We modify the experiment by replacing each different 2ℓ-bit key (established by an execution of 2PAKE) with a random key drawn uniformly from $\{0,1\}^{2\ell}$ for all clean instances. The difference in $\mathcal{A}$'s success probability between $\mathbf{Exp}_0$ and $\mathbf{Exp}_1$ is bounded by:

Claim 1. $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_1] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_0]\right| \leq \mathsf{Adv}^{\text{ake}}_{\text{2PAKE}}(t', Q')$.

Proof. We prove the claim by constructing an adversary $\mathcal{A}'$ who attacks the AKE security of 2PAKE with advantage equal to $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_1] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_0]\right|$. Let k^i_U denote the 2ℓ-bit key held by instance Π^i_U.

$\mathcal{A}'$ chooses a random bit $b \in \{0,1\}$ and invokes the adversary $\mathcal{A}$. $\mathcal{A}'$ then simulates the oracles for $\mathcal{A}$ as follows:

$\mathsf{Execute}$ queries. When an $\mathsf{Execute}(\Pi^i_A, \Pi^j_B, \Pi^k_S)$ query is asked, $\mathcal{A}'$ first checks if A, B or S was previously corrupted.

- If so, $\mathcal{A}'$ answers the Execute query as in experiment $\mathbf{Exp}_0$.
- Otherwise, $\mathcal{A}'$ answers the query using its own oracles. $\mathcal{A}'$ first asks two queries $\mathsf{Execute}(\Pi_A^i, \Pi_S^k)$ and $\mathsf{Execute}(\Pi_B^j, \Pi_S^{k'})$. Let $\mathsf{T}_{\text{2PAKE}}$ and $\mathsf{T}'_{\text{2PAKE}}$ be two transcripts returned in response to the Execute queries. Next, $\mathcal{A}'$ makes the queries $\mathsf{Test}(\Pi_A^i)$ and $\mathsf{Test}(\Pi_B^j)$ and receives in return two keys $\overline{k}_A^i$ and $\overline{k}_B^j$ (either real or random). $\mathcal{A}'$ then generates the rest of the protocol messages, using $\overline{k}_A^i$ and $\overline{k}_B^j$ as k_A^i and k_B^j, respectively. Finally, $\mathcal{A}'$ returns these messages together with $\mathsf{T}_{\text{2PAKE}}$ and $\mathsf{T}'_{\text{2PAKE}}$ after ordering them properly.

Send queries. For each $\mathsf{Send}(\Pi_U^i, m)$ query, $\mathcal{A}'$ checks if m is a message for initiating a new session (of 2R3PAKE) or the Send query belongs to an execution of 2PAKE.

1. If both are untrue, $\mathcal{A}'$ responds to the query as in experiment $\mathbf{Exp}_0$.
2. Otherwise, $\mathcal{A}'$ answers it by making the same query to its own Send oracle. If the query prompts Π_U^i to accept, then $\mathcal{A}'$ checks if Π_U^i is clean.
   (a) If so, $\mathcal{A}'$ makes a $\mathsf{Test}(\Pi_U^i)$ query (unless the partner of Π_U^i has already been tested) and sets k_U^i to be the output of this Test query.
   (b) Otherwise, $\mathcal{A}'$ makes a $\mathsf{Reveal}(\Pi_U^i)$ query and sets k_U^i to be the output of this Reveal query.

Reveal queries. $\mathcal{A}'$ responds to the queries as per the protocol specification.
Corrupt queries. $\mathcal{A}'$ answers these queries using its own Corrupt oracle.
Test queries. $\mathcal{A}'$ responds to these queries based on the randomly chosen bit b at the beginning of the simulation. $\mathcal{A}'$ will return the real session key if $b = 1$ and a random key chosen uniformly at random from $\mathbb{G}$ if $b = 0$.

At some point in time, $\mathcal{A}$ will terminate and output its guess b'. When this happens, $\mathcal{A}'$ outputs 1 if $b = b'$ and 0 otherwise.

From the simulation, it is clear that:

- The probability that $\mathcal{A}'$ outputs 1 when its Test oracle returns real session keys is equal to the probability that $\mathcal{A}$ correctly guesses the bit b in experiment $\mathbf{Exp}_0$.
- The probability that $\mathcal{A}'$ outputs 1 when its Test oracle returns random keys is equal to the probability that $\mathcal{A}$ correctly guesses the bit b in experiment $\mathbf{Exp}_1$.

That is, $\mathsf{Adv}_{\text{2PAKE}}^{\text{ake}}(\mathcal{A}') = \left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_1] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_0]\right|$. Since $\mathcal{A}'$ has at most time complexity t' and query complexity $Q' = (2q_{\text{exec}}, q_{\text{send}}, q_{\text{send}}, q_{\text{corr}}, 2q_{\text{exec}} + q_{\text{send}})$, it follows, by definition, that $\mathsf{Adv}_{\text{2PAKE}}^{\text{ake}}(\mathcal{A}') \leq \mathsf{Adv}_{\text{2PAKE}}^{\text{ake}}(t', Q')$. This completes the proof of Claim 1. $\square$

Experiment $\mathbf{Exp}_2$. This experiment is different from $\mathbf{Exp}_1$, only in that it is aborted and the adversary does not succeed if the following event Forge occurs.

Forge**:** The event that the adversary $\mathcal{A}$ makes a Send query of the form $\mathsf{Send}(\Pi_U^i,\ V\|msg)$ for uncorrupted U and V, such that msg contains a MAC forgery.

Then, we have:

Claim 2. $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_2] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_1]\right| \leq q_{\text{send}} \cdot \mathsf{Adv}_{\Sigma}^{\text{suf-cma}}(t', 2, 2)$.

Proof. Assuming that the event Forge occurs, we construct a forger $\mathcal{F}$ who outputs, with a non-negligible probability, a forgery against the MAC scheme Σ. The forger $\mathcal{F}$ is given oracle access to $\mathsf{Mac}_k(\cdot)$ and $\mathsf{Ver}_k(\cdot)$. The goal of $\mathcal{F}$ is to produce a message/tag pair (m, σ), such that: (1) $\mathsf{Ver}_k(m, \sigma) = 1$; and (2) σ was not previously output by the $\mathsf{Mac}_k(\cdot)$ oracle on input m.

Let n be the number of all different MAC keys established via a Send query made by $\mathcal{A}$. Clearly, $n \leq q_{\text{send}}$. $\mathcal{F}$ begins by choosing a random $\alpha \in \{1, \ldots, n\}$. Let k_{α}^{mac} denote the $\alpha-$th key among all of the n MAC keys and Send_{α} be a Send query that should be answered and/or verified using k_{α}^{mac}. $\mathcal{F}$ invokes $\mathcal{A}$ as a subroutine and handles the oracle calls of $\mathcal{A}$ as in experiment $\mathbf{Exp}_1$, except that: it answers all Send_{α} queries by accessing its MAC generation and verification oracles. As a result, the $\alpha-$th MAC key k_{α}^{mac} is never used during the simulation. If Forge occurs against an instance that holds k_{α}^{mac}, $\mathcal{F}$ halts and outputs the message/tag pair generated by $\mathcal{A}$ as its forgery. Otherwise, $\mathcal{F}$ halts and outputs a failure indication.

If the guess α is correct, then the simulation is perfect, and $\mathcal{F}$ achieves its goal. Namely, $\mathsf{Adv}_{\Sigma}^{\text{suf-cma}}(\mathcal{F}) = \Pr[\mathsf{Forge}]/n$. Since $n \leq q_{\text{send}}$, we get $\Pr[\mathsf{Forge}] \leq q_{\text{send}} \cdot \mathsf{Adv}_{\Sigma}^{\text{suf-cma}}(\mathcal{F})$. As $\mathcal{F}$ has at most time complexity t' and makes at most two queries to $\mathsf{Mac}_k(\cdot)$ and $\mathsf{Ver}_k(\cdot)$, it follows, by definition, that $\mathsf{Adv}_{\Sigma}^{\text{suf-cma}}(\mathcal{F}) \leq \mathsf{Adv}_{\Sigma}^{\text{suf-cma}}(t', 2, 2)$. This completes the proof of Claim 2. □

Experiment $\mathbf{Exp}_3$. We further modify the experiment so that Execute and Send oracles are simulated as in "the $\mathbf{Exp}_3$ modification" described below.

The $\mathbf{Exp}_3$ modification

When $\mathcal{A}$ asks an Execute or Send query, the simulator answers it exactly as in experiment $\mathbf{Exp}_2$, except that it modifies the way of generating the ephemeral public values (denoted as X and Y in the protocol) as follows:

- The simulator chooses two random $v_1, v_2 \in \mathbb{Z}_q^*$ and computes $V_1 = g^{v_1}$ and $V_2 = g^{v_2}$.
- For each instance Π_C^i, the simulator chooses a random $r \in \mathbb{Z}_q^*$, computes:

$$R = \begin{cases} {V_1}^r & \text{if } C \text{ appears first in } pid_C^i \\ {V_2}^r & \text{if } C \text{ appears second in } pid_C^i, \end{cases}$$

and uses R as the ephemeral public value (*i.e.*, as X or Y) of Π_C^i.

Since the view of $\mathcal{A}$ is identical between $\mathbf{Exp}_2$ and $\mathbf{Exp}_3$, it is straightforward to see that:

Claim 3. $\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_3] = \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_2]$.

Experiment $\mathbf{Exp}_4$. In this experiment, each $\overline{X}$ and $\overline{Y}$ is computed as $\overline{X} = X$ and $\overline{Y} = Y$ (instead of as $\overline{X} = X^z$ and $\overline{Y} = Y^z$) if they are expected to be encrypted with a key held by a clean (server) instance. This is the only difference from $\mathbf{Exp}_3$.

Claim 4. $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_4] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_3]\right| \leq q_{\text{send}} \cdot \mathsf{Adv}_{\Omega}^{\text{ind-seav}}(t')$.

Proof. We prove the claim by constructing a multiple eavesdropper $\mathcal{A}_{\text{meav}}$ who attacks the indistinguishability of Ω with advantage equal to $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_4] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_3]\right|$.

$\mathcal{A}_{\text{meav}}$ chooses a random bit $b \in \{0, 1\}$ and invokes the adversary $\mathcal{A}$. $\mathcal{A}_{\text{meav}}$ then handles all of the oracle queries of $\mathcal{A}$ as in experiment $\mathbf{Exp}_3$, except that it generates α_A and α_B for each clean server instance as follows:

> $\mathcal{A}_{\text{meav}}$ outputs $(X, \overline{X} = X^z)$ and $(Y, \overline{Y} = Y^z)$ as its own (two) plain text pairs (in the indistinguishability experiment $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}$), receives in return two ciphertexts c_1 and c_2 and sets $\alpha_A = c_2$ and $\alpha_B = c_1$. (Note, here, that c_1 and c_2 are encryptions of either X and Y or $\overline{X}$ and $\overline{Y}$.)

When $\mathcal{A}$ outputs its guess b', $\mathcal{A}_{\text{meav}}$ outputs 1 if $b = b'$ and 0 otherwise. It easily follows that:

- The probability that $\mathcal{A}_{\text{meav}}$ outputs 1 when the first plain texts are encrypted in experiment $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}$ is equal to the probability that $\mathcal{A}$ succeeds in experiment $\mathbf{Exp}_4$.
- The probability that $\mathcal{A}_{\text{meav}}$ outputs 1 when the second plain texts are encrypted in experiment $\mathbf{Exp}_{\Omega}^{\text{ind-meav}}$ is equal to the probability that $\mathcal{A}$ succeeds in experiment $\mathbf{Exp}_3$.

Therefore, $\mathsf{Adv}_{\Omega}^{\text{ind-meav}}(\mathcal{A}_{\text{meav}}) = \left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_4] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_3]\right|$. Since $\mathcal{A}_{\text{meav}}$ eavesdrops at most q_{send} encryptions and has time complexity at most t', Claim 4 follows immediately from Lemma 1 of Section 4.1. □

Experiment $\mathbf{Exp}_5$. We now modify the way session keys are computed. For each clean instance and its partner instance, the shared session key is chosen uniformly at random from $\mathbb{G}$.

Claim 5. $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_5] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_4]\right| \leq \mathsf{Adv}_{\mathbb{G}}^{\text{ddh}}(t')$.

Proof. We prove the claim by constructing an algorithm $\mathcal{A}_{\text{ddh}}$ that solves the DDH problem in $\mathbb{G}$ with advantage equal to $\left|\Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_5] - \Pr_{\text{2R3PAKE},\mathcal{A}}[\mathsf{Succ}_4]\right|$.

On input, a DDH-problem instance $(W_1 = g^{w_1}, W_2 = g^{w_2}, W_3) \in \mathbb{G}^3$, $\mathcal{A}_{\text{ddh}}$ chooses a random bit $b \in \{0, 1\}$, invokes the adversary $\mathcal{A}$ and simulates the oracles on its own. $\mathcal{A}_{\text{ddh}}$ handles all of the queries of $\mathcal{A}$ as in experiment $\mathbf{Exp}_4$, except for the following:

- $\mathcal{A}_{\text{ddh}}$ uses W_1 and W_2 in place of V_1 and V_2 (see "the $\mathbf{Exp}_3$ modification").
- For each clean instance Π_C^i who sends $X = {W_1}^r$ and receives $Y = {W_2}^{r'}$, or *vice versa*, $\mathcal{A}_{\text{ddh}}$ sets the session key sk_C^i to be $W_3^{rr'}$.

Later, when $\mathcal{A}$ outputs its guess b', $\mathcal{A}_{\text{ddh}}$ outputs 1 if $b = b'$ and 0 otherwise.

The simulation above clearly shows that:

- The probability that $\mathcal{A}_{\text{ddh}}$ outputs 1 on a true Diffie–Hellman triple is equal to the probability that $\mathcal{A}$ correctly guesses the bit b in experiment $\mathbf{Exp}_4$.
- The probability that $\mathcal{A}_{\text{ddh}}$ outputs 1 on a random triple is equal to the probability that $\mathcal{A}$ correctly guesses the bit b in experiment $\mathbf{Exp}_5$.

Hence, $\mathsf{Adv}_{\mathbb{G}}^{\text{ddh}}(\mathcal{A}_{\text{ddh}}) = \left|\Pr_{2\text{R3PAKE},\mathcal{A}}[\mathsf{Succ}_5] - \Pr_{2\text{R3PAKE},\mathcal{A}}[\mathsf{Succ}_4]\right|$. From this and since $\mathsf{Adv}_{\mathbb{G}}^{\text{ddh}}(\mathcal{A}_{\text{ddh}}) \leq \mathsf{Adv}_{\mathbb{G}}^{\text{ddh}}(t')$, we obtain the inequality of Claim 5. □

In experiment $\mathbf{Exp}_5$, the session keys of all fresh instances are chosen uniformly at random from $\mathbb{G}$, and thus, the adversary $\mathcal{A}$ obtains no information on the bit b chosen by the Test oracle. Therefore, it follows that $\Pr[\mathsf{Succ}_5] = 1/2$. This result combined with Claims 1–5 yields the statement of Theorem 1. □

5. Concluding Remarks

In this paper, we have proposed an efficient and secure three-party password-only authenticated key exchange protocol that requires only two communication rounds. We have rigorously proven the security of the protocol in a widely-accepted adversary model. Since our proof of security requires no idealizing assumptions, our proposed protocol would be considered equivalent to being provably secure in the standard model, as long as the building blocks are also instantiated with schemes proven secure in the standard model. For a more efficient implementation of our proposed protocol, Steps 3 and 6 (see the protocol description in Section 4.2) can be omitted if security against undetectable online dictionary attacks is not required. This simplified protocol would still be AKE-secure in the sense of Definition 2 (*i.e.*, Theorem 1 also holds for the simplified protocol). We finally note that it seems impossible to design an AKE-secure, one-round key exchange protocol in the password-only, three-party setting, although we are unable to prove this statement formally.

Acknowledgments

This work was supported by Konkuk University.

Author Contributions

S.H. and D.W. conceived and designed the experiments; S.H. and J.P. performed the experiments; J.P. and D.W. analyzed the data; J.N. and K.C. proved the security of the protocol; J.N. and K.C. wrote the paper.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Hervey, C.; van Oorschot, P. A research agenda acknowledging the persistence of passwords. *IEEE Secur. Priv.* **2012**, *10*, 28–36.
2. Wang, W.; Hu, L. Efficient and provably secure generic construction of three-party password-based authenticated key exchange protocols. In Proceedings of the INDOCRYPT 2006—7th International Conference on Cryptology, Kolkata, India, 11–13 December 2006; Volume 4329, pp. 118–132.

3. Nam, J.; Paik, J.; Kang, H.; Kim, U.; Won, D. An off-line dictionary attack on a simple three-party key exchange protocol. *IEEE Commun. Lett.* **2009**, *13*, 205–207.
4. Lo, N.; Yeh, K. Cryptanalysis of two three-party encrypted key exchange protocols. *Comput. Stand. Interfaces* **2009**, *31*, 1167–1174.
5. Lin, C.; Hwang, T. On a simple three-party password-based key exchange protocol. *Int. J. Commun. Syst.* **2011**, *24*, 1520–1532.
6. Wu, S.; Pu, Q.; Wang, S.; He, D. Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol. *Inform. Sci.* **2012**, *215*, 83–96.
7. Choo, K.K.R.; Boyd, C.; Hitchcock, Y. Errors in computational complexity proofs for protocols. In Proceedings of the ASIACRYPT 2005—11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, 4–8 December 2005; Volume 3788, pp. 624–643.
8. Choo, K.K.R.; Boyd, C.; Hitchcock, Y. Examining indistinguishability-based proof models for key establishment protocols. In Proceedings of the ASIACRYPT 2005—11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, 4–8 December 2005; Volume 3788, pp. 585–604.
9. Choo, K.K.R.; Boyd, C.; Hitchcock, Y. The importance of proofs of security for key establishment protocols: Formal analysis of Jan–Chen, Yang–Shen–Shieh, Kim–Huh–Hwang–Lee, Lin–Sun–Hwang, and Yeh–Sun protocols. *Comput. Commun.* **2006**, *29*, 2788–2797.
10. Bellare, M.; Rogaway, P. Entity authentication and key distribution. In Proceedings of the 13th Annual International Cryptology Conference, Santa Barbara, CA, USA, 22–26 August 1993; Volume 773, pp. 232–249.
11. Abdalla, M.; Fouque, P.; Pointcheval, D. Password-based authenticated key exchange in the three-party setting. In Proceedings of the Public Key Cryptography 2005, Les Diablerets, Switzerland, 23–26 January 2005; Volume 3386, pp. 65–84.
12. Abdalla, M.; Fouque, P.; Pointcheval, D. Password-based authenticated key exchange in the three-party setting. *IEE Proc. Inform. Secur.* **2006**, *153*, 27–39.
13. Lin, C.; Sun, H.; Steiner, M.; Hwang, T. Three-party encrypted key exchange without server public-keys. *IEEE Commun. Lett.* **2001**, *5*, 497–499.
14. Lee, T.; Hwang, T.; Lin, C. Enhanced three-party encrypted key exchange without server public keys. *Comput. Secur.* **2004**, *23*, 571–577.
15. Abdalla, M.; Pointcheval, D. Interactive Diffie-Hellman assumptions with applications to password-based authentication. In *Financial Cryptography and Data Security*; Patrick, A.S., Yung, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 341–356.
16. Wen, H.; Lee, T.; Hwang, T. Provably secure three-party password-based authenticated key exchange protocol using Weil pairing. *IEE Proc. Commun.* **2005**, *152*, 138–143.
17. Lu, R.; Cao, Z. Simple three-party key exchange protocol. *Comput. Secur.* **2007**, *26*, 94–97.
18. Chung, H.; Ku, W. Three weaknesses in a simple three-party key exchange protocol. *Inform. Sci.* **2008**, *178*, 220–229.
19. Guo, H.; Li, Z.; Mu, Y.; Zhang, X. Cryptanalysis of simple three-party key exchange protocol. *Comput. Secur.* **2008**, *27*, 16–21.

20. Kim, H.; Choi, J. Enhanced password-based simple three-party key exchange protocol. *Comput. Electr. Eng.* **2009**, *35*, 107–114.
21. Huang, H. A simple three-party password-based key exchange protocol. *Int. J. Commun. Syst.* **2009**, *22*, 857–862.
22. Dongna, E.; Cheng, Q.; Ma, C. Password authenticated key exchange based on RSA in the three-party settings. In Proceedings of the Provable Security Conference 2009, Guangzhou, China, 11–13 November 2009; Volume 5848, pp. 168–182.
23. Lee, T.; Hwang, T. Simple password-based three-party authenticated key exchange without server public keys. *Inform. Sci.* **2010**, *180*, 1702–1714.
24. Wang, W.; Hu, L.; Li, Y. How to construct secure and efficient three-party password-based authenticated key exchange protocols. In Proceedings of The 6th China International Conference on Information Security and Cryptology, Shanghai, China, 20–24 October 2010; Volume 6584, pp. 218–235.
25. Chang, T.; Hwang, M.; Yang, W. A communication-efficient three-party password authenticated key exchange protocol. *Inform. Sci.* **2011**, *181*, 217–226.
26. Nam, J.; Lee, Y.; Kim, S.; Won, D. Security weakness in a three-party pairing-based protocol for password authenticated key exchange. *Inform. Sci.* **2007**, *177*, 1364–1375.
27. Phan, R.; Yau, W.; Goi, B. Cryptanalysis of simple three-party key exchange protocol (S-3PAKE). *Inform. Sci.* **2008**, *178*, 2849–2856.
28. Yoon, E.; Yoo, K. Cryptanalysis of a simple three-party password-based key exchange protocol. *Int. J. Commun. Syst.* **2011**, *24*, 532–542.
29. Liang, H.; Hu, J.; Wu, S. Re-attack on a three-party password-based authenticated key exchange protocol. *Math. Comput. Model.* **2013**, *57*, 1175–1183.
30. Tsai, H.; Chang, C. Provably secure three party encrypted key exchange scheme with explicit authentication. *Inform. Sci.* **2013**, *238*, 242–249.
31. Nam, J.; Choo, K.K.R.; Park, M.; Paik, J.; Won, D. On the security of a simple three-party key exchange protocol without server's public keys. *Sci. World J.* **2014**, 479534:1–479534:7 .
32. Nam, J.; Choo, K.K.R.; Paik, J.; Won, D. An offline dictionary attack against Abdalla and Pointcheval's key exchange in the password-only three-party setting. *IEICE Trans. Fundam. Electr. Commun. Comput. Sci.* **2015**, in press.
33. Szydlo, M. A note on Chosen-Basis Decisional Diffie-Hellman assumptions. In Proceedings of the Financial Cryptography 2006, Anguilla, British West Indies, 27 February–2 March 2006; Volume 4107, pp. 166–170.
34. Yoneyama, K. Efficient and strongly secure password-based server aided key exchange. In Proceedings of the INDOCRYPT 2008: 9th International Conference on Cryptology, Kharagpur, India, 14–17 December 2008; Volume 5365, pp. 172–184.
35. Zhao, J.; Gu, D. Provably secure three-party password-based authenticated key exchange protocol. *Inform. Sci.* **2012**, *184*, 310–323.
36. Lin, C.; Sun, H.; Hwang, T. Three-party encrypted key exchange: Attacks and a solution. *ACM SIGOPS Oper. Syst. Rev.* **2000**, *34*, 12–20.

37. Chang, C.; Chang, Y. A novel three-party encrypted key exchange protocol. *Comput. Stand. Interfaces* **2004**, *26*, 471–476.
38. Chen, H.; Chen, T.; Lee, W.; Chang, C. Security enhancement for a three-party encrypted key exchange protocol against undetectable on-line password guessing attacks. *Comput. Stand. Interfaces* **2008**, *30*, 95–99.
39. Yoon, E.; Yoo, K. Improving the novel three-party encrypted key exchange protocol. *Comput. Stand. Interfaces* **2008**, *30*, 309–314.
40. Chien, H.; Wu, T. Provably secure password-based three-party key exchange with optimal message steps. *Comput. J.* **2009**, *52*, 646–655.
41. Lou, D.; Huang, H. Efficient three-party password-based key exchange scheme. *Int. J. Commun. Syst.* **2011**, *24*, 504–512.
42. Yang, J.; Cao, T. Provably secure three-party password authenticated key exchange protocol in the standard model. *J. Syst. Softw.* **2012**, *85*, 340–350.
43. Lee, C.; Chen, S.; Chen, C. A computation-efficient three-party encrypted key exchange protocol. *Appl. Math. Inform. Sci.* **2012**, *6*, 573–579.
44. Wu, S.; Chen, K.; Pu, Q.; Zhu, Y. Cryptanalysis and enhancements of efficient three-party password-based key exchange scheme. *Int. J. Commun. Syst.* **2013**, *26*, 674–686.
45. Bellare, M.; Pointcheval, D.; Rogaway, P. Authenticated key exchange secure against dictionary attacks. In Proceedings of the Eurocrypt 2000, Bruges, Belgium, 14–18 May 2000; Volume 1807, pp. 139–155.
46. Abdalla, M.; Pointcheval, D. Simple password-based encrypted key exchange protocols. In Proceedings of the CT-RSA 2005, San Francisco, CA, USA, 14–18 February 2005; Volume 3376, pp. 191–208.
47. Katz, J.; Vaikuntanathan, V. Round-optimal password-based authenticated key exchange. In Proceedings of the Theory of Cryptography Conference 2011, Providence, RI, USA, 28–30 March 2011; Volume 6597, pp. 293–310.
48. Choo, K.K.R. A proof of revised Yahalom protocol in the Bellare and Rogaway (1993) model. *Comput. J.* **2007**, *50*, 591–601.
49. Bellare, M.; Rogaway, P. Provably secure session key distribution—The three party case. In Proceedings of the 27th ACM Symposium on Theory of Computing, Las Vegas, NV, USA, May 1995; pp. 57–66.
50. Goldwasser, S.; Micali, S. Probabilistic encryption. *J. Comput. Syst. Sci.* **1984**, *28*, 270–299.