情報保護學會論文誌
第15卷 第5號, 2005. 10

패스워드 인증된 Joux의 키 교환 프로토콜*

이 상 곤,[1†‡] Yvonne Hitcock,[2] 박 영 호,[3] 문 상 재[4]
[1]동서대학교 인터넷공학부, [2]ISI, QUT, 호주, [3]상주대학교 전자전기공학부,
[4]경북대학교 전자전기컴퓨터학부

Password Authenticated Joux's Key Exchange Protocol*

Sang-gon Lee,[1†‡] Yvonne Hitcock,[2] Young-ho Park[3] Sang-jae Moon[4]
[1]Dongseo Univ., [2]ISI, QUT, Australia, [3]Sangju National Univ.,
[4]Kyungpook National Univ.

요 약

Joux의 3자 키 교환 프로토콜은 키 합의 분야에서 가장 뛰어난 업적 가운데 하나이다. 하지만 Joux 프로토콜은 인증 기능을 제공하지 않아 man-in-the-middle 공격에 취약하다. 비록 Joux 프로토콜에 대하여 인증서 기반 그리고 ID 기반 인증기법이 제안되었지만, 아직 1 라운드에 실행되는 안전성이 증명 가능한 패스워드 기반 3자 키 교환 프로토콜은 제안된 바 없다. 본 논문에서는 Joux 프로토콜에 EC-PAK의 패스워드 인증 기법을 적용하여 안전성이 증명 가능한 1 라운드 3자 키 교환 프로토콜을 제안하였다. 그리고 렌덤 오라클 모델을 사용하여 프로토콜의 안전성도 증명하였다.

ABSTRACT

Joux's tripartite key agreement protocol is one of the most prominent developments in the area of key agreement. Although certificate-based and ID-based authentication schemes have been proposed to provide authentication for Joux's protocol, no provably secure password-based one round tripartite key agreement protocol has been proposed yet. We propose a secure one round password-based tripartite key agreement protocol that builds on Joux's protocol and adapts PAK-EC scheme for password-based authentication, and present a proof of its security.

Keywords : *Tripartite Key Agreement Protocol; Password-based Authentication; Probable Security; Bilinear Diffie-Hellman Problem; Joux's Protocol.*

Ⅰ. Introduction

A *Key Agreement Protocol* is the mechanism by which two or more parties can establish a common secret key over a network controlled by an adversary. This secret key is commonly called a session key and can then be used to create a secure communications channel among the parties.

The situation where three or more parties share a key is often called conference keying. The three-party (or tripartite) case is of the most practical importance, not only because it is the most common size for electronic conferences, but also because it can be used to provide a range

접수일 : 2005년 5월 20일 ; 채택일 : 2005년 8월 16일
* 본 연구는 정보통신부 대학 IT 연구센터 육성지원사업의 연구결과로 수행되었습니다.
† 주저자, ‡ 교신저자, nok60@dongseo.ac.kr

of services for two communicating parties. For example, a third party can be added to chair, or referee a conversation for ad hoc auditing. Also, a three-party key agreement protocol can be used for tree based group key agreement protocols.[3,6,19]

Joux's tripartite key agreement protocol[14] is one of the most prominent development in the area of key agreement. This protocol makes use of pairings on elliptic curves and requires each entity to transmit only a single broadcast message. This should be contrasted with the obvious extension of the Diffie-Hellman protocol to three parties, which requires two broadcasts per entity. However, like the basic Diffie-Hellman protocol, Joux's protocol also suffers from man-in-the-middle attacks because it does not provide key *authentication*.

To transform Joux's protocol into a secure tripartite protocol that still requires only a single broadcast per entity, many protocols have been proposed. Basically these protocols can be divided into two broad categories, i.e., certificate-based protocols[1,24] and ID-based protocols.[22,23,25,26] Another authentication scheme for key agreement is making use of a password.[7,10,20,21] Certificate-based authentication requires a certificate authority and ID-based authentication requires a trusted dealer who has a universal secret key. However, password-based authentication doesn't require any trusted third party. No provably secure password-based one round tripartite key agreement protocol has been proposed.

PAK-EC[20] is a concrete two party password authenticated key agreement protocol built on elliptic curves. In this paper we present a provably secure one round password-based tripartite key agreement protocol that builds on Joux's protocol *and adapts the PAK-EC scheme for password-based authentication*. Our contribution is to present a provably secure password-based Joux's protocol version.

In section 2, we explain our security model for the proof of security of the proposed protocol. In section 3, the proposed protocol is described. In section 4, we describe our proof of the proposed protocol. In section 5, we compare our protocol with other one in computation and communication complexity.

II. Security Model

For our proof of security we use the model of Bellare, Pointcheval and Rogaway[4](which builds on earlier models,[5,6] and is also used by Kate et al.[15] and *MacKenzie*[21]), *and adopt MacKenzie's approach.*[21] Our model is for implicitly authenticated key exchange between parties A, B and C who share a secret. The goal is for them to engage in a protocol such that after the protocol is completed, they each hold a session key that is known to nobody but the three of them. In the following, we will describe our model.

Protocol Participants. Let I be a non-empty set of participants. We assume each participant $U \in I$ is labeled by a string, and we simply use U to denote this string. We will also use $A, B, C, \cdots$ to refer to protocol participants. Each group of three participants, $A, B, C, \in I$, who will set up a secret key shared amongst themselves are assumed to share a secret password with each other, π_{ABC}, before the protocol begins.

Execution of protocol. For a protocol P, each participant is able to execute P multiple times with different partners,

and we model this by allowing unlimited number of *instances* of each participant. Instance i (session number i) of participant $U \in I$ is denoted Π_i^U. To describe the security of the protocol, we assume there is an adversary $\mathcal{A}$ that has complete control over the environment (mainly, the network), and thus provides the input to instances of participants. Formally, the adversary is a probabilistic algorithm with a distinguished query tape. Participants respond to queries written to this tape according to P; the allowed queries are based on and extend the model of Bellare et. al.[4]. Oracles exist in one of several possible states ***Accept***, ***Reject***, or *. The state * means no decision has yet been reached. In our protocol, an oracle accepts only after receipt of two correctly formatted messages from the two other participants with whom the oracle wishes to establish, and the transmission of one message. When an oracle accepts, we assume it accepts holding key K that is bits in length.

Send(U,i,M) : causes message M to be sent to instant Π_i^U. The instance computes what the protocol says to, the oracle's state is updated, any outgoing messages are given to $\mathcal{A}$. If this query causes Π_i^U to accept or terminate, this will also be shown to $\mathcal{A}$. To initiate a session between three participants, the adversary should send a message containing the name of two participants to unused instances of the other two participants.

Execute(A,i,B,j,C,l) : causes P to be executed to completion between Π_i^A, Π_j^B and Π_l^C (where $A,B,C \in I$), and outputs the transcript of the execution. This query captures the intuition of a passive adversary who simply eavesdrops on the execution of P.

Reveal(U,i) : causes the output of the session key held by Π_i^U.

Test(U,i) : causes Π_i^U to flip a bit b. if b = 1 the session key sk_U^i is output; otherwise, *a string is drawn uniformly from* the space of session keys and output. A **Test** query may be asked at any time during the execution of P, but may only be asked once.

Corrupt(U) : if $U \in I$, this query returns any passwords that U holds.

Partnering. A participant instant that accepts holds a partner-id *pid*, session-id *sid*, and a session key *sk*. Then instance Π_i^A, Π_j^B and Π_l^C (where $A,B,C \in I$) are said to be partnered if all of them accept, they hold (pid_A, sid_A, sk_A), (pid_B, sid_B, sk_B) and (pid_C, sid_C, sk_C) respectively, with $pid_A = <B,C>$, $pid_B = <A,C>$, $pid_C = <A,B>$, $sid_A = sid_B = sid_C$, and $sk_A = sk_B = sk_C$, and no other instance accepts with session-id equal to sid_A, sid_B or sid_C.

Freshness. We define two notions of freshness, as in [4]. Specifically, an instance Π_i^U is nfs-fresh(fresh with no requirement for forward secrecy) unless either (1) a **Reveal**(U,i) query occurs, (2) a **Reveal**(U',j) query occurs where $\Pi_j^{U'}$ is a partner of Π_i^U, or (3) a **Corrupt** (U') query occurs for any party U'. (For convenience, when we do not make a requirement for forward secrecy, we simply disallow **Corrupt** queries) An instance Π_i^U is fs-fresh(fresh with forward secrecy) unless either (1) a **Reveal**(U,i) query occurs, (2) a **Reveal**(U',j) query occurs where $\Pi_j^{U'}$ is the partner of Π_i^U, or (3) a **Corrupt**(U') query occurs for any party U' before the

Test query and a **Send**(U,i,M) query occurs for some string M.

Advantage of the adversary. We now formally define the authenticated key exchange (ake) advantage of the adversary against protocol P. Let $Succ_P^{ake}(\mathcal{A})$ be the event that $\mathcal{A}$ makes a single **Test** query directed to some fresh instance Π_i^U that has terminated, and eventually outputs a bit b', where $b'=b$ for the bit b that was selected in the **Test** query. The *ake* advantage of A attacking P is defined to be

$$Adv_P^{ake}(\mathcal{A}) \stackrel{def}{=} 2\Pr[Succ_P^{ake}(\mathcal{A})]-1.$$

The following fact is easily verified.

Fact 2.1

$$Pr(Succ_P^{ake}(\mathcal{A})) = Pr(Succ_P^{ake}(\mathcal{A})) + \epsilon$$
$$\Leftrightarrow Adv_P^{ake}(\mathcal{A}) = Adv_P^{ake}(\mathcal{A})) + 2\epsilon$$

Ⅲ. Tripartite Password Protected Key Exchange Protocol via Pairings.

In this section, we briefly describe some background on pairings on elliptic curves and the BDH assumption, and then present our new tripartite PPK protocols based on Joux's protocol.[14]

3.1 Bilinear Pairings and the BDH assumption

We use the same notation as in [8]. Let G_1 be a cyclic additive group generated by Q, whose order is a prime q, and G_2 be a cyclic multiplicative group with the same order q. We assume that the discrete logarithm problem(DLP) in both G_1 and G_2 is hard. Let $e: G_1 \times G_2 \rightarrow G_2$ be a pairing which satisfies the following conditions:

1. Bilinear:
   $e(W,X+Z)=e(W,X)\cdot e(W,Z)$ and
   $e(W+X,Z)=e(W,Z)\cdot e(X,Z)$ for $W,X,Z \in G_1$
2. Non-degenerate: an element $Q \in G_1$ satisfying $e(Q,Q) \neq 1$ is known
3. Computability: there is an efficient algorithm to compute $e(P,Q)$ for all $P,Q \in G_1$.

The map e will be derived from either the Weil or Tate pairing on the elliptic curve. We refer to [9,12,17,18] for more details.

Definition 1. Bilinear Diffie-Hellman (BDH) Problem

Here we formally state the BDH assumption. Let G_1 *and* G_2 *be as defined above with generator* Q *and* $e(Q,Q)$ *respectively. Let* ACCEPTABLE(v) *be a function that returns true if and only if* $v \in G_1$. *For three values* X, Y, *and* Z, *if* ACCEPTABLE(Y), *and* ACCEPTABLE(Z), *and* $X=aQ$, *let* $BDH(X,Y,Z)=e(Y,Z)^a$, *else if* ACCEPTABLE(X), *and* ACCEPTABLE(Z), *and* $Y=bQ$, *let* $BDH(X,Y,Z)=(X,Z)^b$, *else if* ACCEPTABLE(X), *and* ACCEPTABLE(Y), *and* $Z=cQ$, *let* $BDH(X,Y,Z)=(X,Y)^c$. (*Note that if* $X=aQ$, $Y=bQ$ *and* $Z=cQ$, *then by definition* $BDH(X,Y,Z)=e(Q,Q)^{abc}$.) *Let* $\mathcal{D}$ *be an algorithm with input* (X,Y,Z). *Let*

$$Adv_{G_1G_2}^{BDH}(\mathcal{D}) \stackrel{def}{=} \Pr[(a,b,c) \stackrel{R}{\leftarrow} Z_q^*; X \leftarrow aQ; Y \leftarrow bQ; Z \leftarrow cQ: BDH(X,Y,Z) \in \mathcal{D}(X,Y,Z)].$$

Let $Adv_{G_1G_2}^{BDH}(t,n)=max_D\{Adv_{G_1G_2}^{BDH}(\mathcal{D})\}$, *where the maximum is taken over all adversaries of time complexity at most* t *that output a list containing at most* n *elements of* G_2. The BDH assumption states

that for t and n polynomial in the security parameter κ, $Adv^{BDH}_{G_1G_2}(t,n)$ is negligible.

3.2 One Round Tripartite PPK based on Joux's protocol.

Fig. 1 presents a new one round tripartite PPK protocol based on Joux's protocol. In the protocol, $f_i(A,B,C,\pi)$ is defined to be a function generating a random point on elliptic curve E from A,B,C and π. The details of the function are given in Appendix A.

We use the terminology "in a **Participant** U **Action** k query to Π_i^U" to mean "in a **Send** query to Π_i^U that results in the a **Participant** U **Action** k procedure being executed." The possible actions with their associated inputs and outputs are shown in Table 1, where A is the initiator, B is the second participant and C is the third participant.

Table 1. Inputs and outputsfor the Participant queries.

Query \ Participant		Action 0	Action 1
A(initiator)	Input	$<B, C>$	$<B, \mu, C, \upsilon>$
	output	$<A, m>$	-
B (second participant)	Input	$<A, m>$	$<C, \upsilon>$
	Output	$<B, \mu>$	-
C (third participant)	Input	$<A, m, B, \mu>$	-
	Output	$<C, \upsilon>$	-

Let be the cryptographic security parameter $q=\kappa$. We use an elliptic curve E over the integers modulo p with coefficients a,b in Standard Weiestrass form and $\#E=rq$, with $gcd(r,q)=1$. (Currently, $p=162$ and $q=160$ would be considered reasonably secure [13]). The complete specification is below:

$sid \leftarrow pid \leftarrow sk \leftarrow \epsilon$, $acc \leftarrow term \leftarrow FALSE$

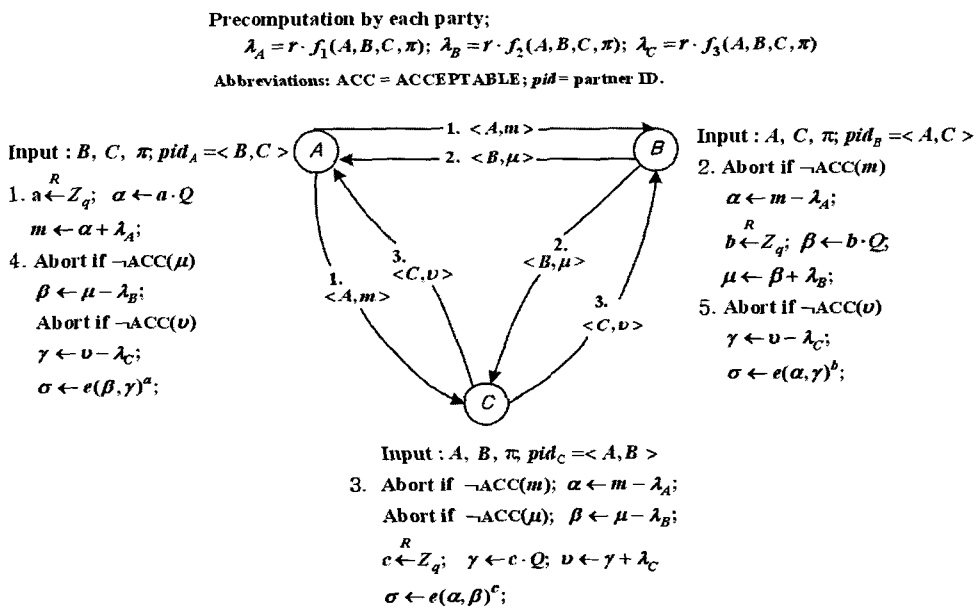


Fig. 1. Tripartite PPK protocol. Session ID is $sid \leftarrow A\|B\|C\|m\|\mu\|\upsilon$. Partner ID for A is $pid_A = \langle B, C\rangle$, partner ID for B is $pid_B = \langle A, C\rangle$, and partner ID for C is $pid_C = \langle A, B\rangle$. Shared session key is $sk = H(\langle A, B, C, m, \mu, \upsilon, \sigma, \lambda_A, \lambda_B, \lambda_C\rangle)$.

if $state = READY$ **and** ($U \in I$ AND U is the initiator) **then**

{Participant A Action 0}

$< A > \leftarrow U$; $< B, C > \leftarrow msg - in$ where $B, C, \in I$;

$a \overset{R}{\leftarrow} Z_q$; $\alpha = aQ$; $\lambda_A = r \cdot f_1(A, B, C, \pi)$; $m \leftarrow \alpha + \lambda_A$; $state \leftarrow < A, a, m, \lambda_A >$; $msg - out \leftarrow < A, m >$

return $(msg - out, acc, term, sid, pid, sk, state)$

elseif $state = READY$ **and** ($U \in I$ AND U is the second participant) **then**

{Participant B Action 0}

$< B > \leftarrow U$; $< A, m > \leftarrow msg - in$, where $A \in I$ and ACCEPTABLE(m);

$\lambda_A = r \cdot f_1(A, B, C, \pi)$; $\alpha \leftarrow m - \lambda_A$; $b \overset{R}{\leftarrow} Z_q$; $\beta = bQ$; $\lambda_B = r \cdot f_2(A, B, C, \pi)$; $\mu \leftarrow \beta + \lambda_B$; $state \leftarrow < A, m, B, b, \mu, \lambda_A, \lambda_B >$; $msg - out \leftarrow < B, \mu >$

return$(msg - out, acc, term, sid, pid, sk, state)$

elseif $state = READY$ **and** ($U \in I$ AND U is the third participant) **then**

{Participant C Action 0}

$< C > \leftarrow U$; $< A, m, B, \mu > \leftarrow msg - in$, where $A, B \in I$ and ACCEPTABLE(m) and ACCEPTABLE(μ); $\lambda_A = r \cdot f_1(A, B, C, \pi)$; $\alpha \leftarrow m - \lambda_A$; $\lambda_B = r \cdot f_2(A, B, C, \pi)$; $\beta \leftarrow \mu - \lambda_B$;

$c \overset{R}{\leftarrow} Z_q$; $\gamma = cQ$; $\lambda_C = r \cdot f_3(A, B, C, \pi)$; $v \leftarrow \gamma + \lambda_C$; $\sigma \leftarrow e(\alpha, \beta)^C$; $state = DONE$; $msg - out \leftarrow < C, v >$; $sid \leftarrow A \| B \| C \| m \| \mu \| v$; $pid \leftarrow < A, B >$; $sk \leftarrow H(< A, B, C, m, \mu, v, \sigma, \lambda_A, \lambda_B, \lambda_C >)$; $acc \leftarrow term \leftarrow TRUE$;

return $(msg - out, acc, term, sid, pid, sk, state)$

elseif $state = < A, a, m, \lambda_A >$ ($U \in I$ AND U is the initiator) **then**

{Participant A Action 1}

$< B, \mu, C, v > \leftarrow msg - in$, where $B, C \in I$ and ACCEPTABLE(μ) and ACCEPTABLE (v); $\lambda_B = r \cdot f_2(A, B, C, \pi)$; $\lambda_C = r \cdot f_3(A, B, C, \pi)$ $\beta \leftarrow \mu - \lambda_B$; $\gamma \leftarrow v - \lambda_C$; $\sigma \leftarrow e(\beta, \gamma)^a$;

The original protocol P.

P_1 If honest parties randomly choose m, μ or v values seen previously in the execution of the protocol, the protocol halts and the adversary fails.

P_2 The protocol answers **Send** and **Execute** queries without making any random oracle queries. Subsequent random oracle queries by the adversary are backpatched, as much as possible, to be consistent with the response to **Send** and **Execute** queries.

P_3 If an $H(\cdot)$ query is made, it is not checked for consistency against Execute queries. That is, instead of backpatching to maintain consistency with an Execute query, the protocol responds with a random output.

P_4 If correct password guess is made against any one of the participant instance (determined by an $H(\cdot)$ query using the correct inputs to compute a session key), the protocol halts and the adversary automatically succeeds.

P_5 If the adversary makes three password guesses against Participant A instance, the protocol halts and the adversary fails.

P_6 If the adversary makes three password guesses against Participant B instance, the protocol halts and the adversary fails.

P_7 If the adversary makes three password guesses against Participant C instance, the protocol halts and the adversary fails.

P_8 The protocol uses an internal password oracle that holds all passwords and only accepts simple queries that test whether a given password is correct password for a given tripartite pair. The test for correct password guesses (from P_4) is changed so that whenever the adversary makes a password guess, a query is submitted to the oracle to determine if it is correct.

Fig. 2. Informal description of protocols P_0 to P_8

$state \leftarrow DONE$; $msg-out \leftarrow \epsilon$; $sid \leftarrow A \| B \| C \| m \| \mu \| v$; $pid \leftarrow < B, C >$; $sk \leftarrow H(< A, sk \leftarrow H(< A, B, C, m, \mu, v, \sigma, \lambda_A, \lambda_B, \lambda_C >)$;
$acc \leftarrow term \leftarrow TRUE$;
return $(msg-out, acc, term, sid, pid, sk, state)$

elseif $state = < A, a, m, B, b, \mu, \lambda_A, \lambda_B >$ **and** ($U \in I$ AND U is the second participant) **then**

{Participant B Action 1}
$< C, v > \leftarrow msg-in$, where $C \in I$ and ACCEPTABLE(v) : $\lambda_C = r \cdot f_3(A, B, C, \pi)$; $\gamma \leftarrow v - \lambda_C$; $\sigma \leftarrow e(\alpha, \gamma)^b$; $state \leftarrow DONE$; $msg-out \leftarrow \epsilon$; $sid \leftarrow A \| B \| C \| m \| \mu \| v$; $pid \leftarrow < A, C >$: $sk \leftarrow H(< A, B, C, m, \mu, v, \sigma, \lambda_A, \lambda_B, \lambda_C >)$; $acc \leftarrow term \leftarrow TRUE$;
return $(msg-out, acc, term, sid, pid, sk, state)$

Ⅳ. Security of the protocol

Here we prove that the tripartite PPK protocol is secure, in the sense that an adversary attacking the system cannotdetermine session keys of fresh instances with greater advantage than that of an online dictionary attack.

Theorem 4.1 *Let P be the protocol described in Fig. 1 (and formally described in Appendix B), using groups G_1 and G_2 of order q, with a password dictionary of size N. Fix an adversary $\mathcal{A}$ that runs in time t, and makes n_{se}, n_{ex}, and n_{ro} queries of type* **Send**, **Execute**, *and* **Reveal**, *respectively, and n_{ro} queries to the random oracles. Let t_{op} be the time required to perform a scalar multiplication and a pairing of elliptic curve point in G_1 and an exponentiation in G_2. Then for $t' = O(t + (n_{ro}^3 + n_{se} + n_{ex}) t_{op})$:*

$$Adv_P^{ake}(\mathcal{A}) = \frac{2n_{se}}{N} + O\left(Adv_{G_1 G_2}^{BDH}(t', n_{ro}^3) + \frac{(n_{se}+n_{ex})(n_{ro}+n_{se}+n_{ex})}{q} \right)$$

Proof : The proof proceeds by introducing a series of protocols $P_0, P_1, \cdots, P_8$ related to P, with $P_0 = P$. In P_8, $\mathcal{A}$ is reduced to a simple online guessing attack that admits straight-forward analysis. We describe these protocols informally in Fig. 2. For each i from 1 to 8, we will prove that the advantage of $\mathcal{A}$ attacking protocol P_{i-1} is at most negligibly more than the advantage of $\mathcal{A}$ attacking protocol P_i. $H(\cdot)$ is a hash function computing a session key.

We assume without loss of generality that n_{ro} and $n_{se}+n_{ex}$ are both at least 1. We make the standard assumption that random oracles are built "on the fly," that is, each new query to a random oracle is answered with a fresh random output, and each query that is not new is answered consistently with the previous queries. We also assume that the $f_j(\cdot)$ query is answered in the following way:

In an $f_j(A, B, C, \pi)$ query for $j \in \{1,2,3\}$, output $\phi_j[A, B, C, \pi]Q$, where $\phi_j[A, B, C, \pi]Q \overset{R}{\leftarrow} Z_q$. Also, put $\psi_j[A, B, C, \pi] = r\phi_j[A, B, C, \pi]$. $\lambda_A \leftarrow r\phi_1[A, B, C, \pi]Q$, $\lambda_B \leftarrow r\phi_2$ $\lambda_B \leftarrow r\phi_2[A, B, C, \pi]$ and $\lambda_C \leftarrow r\phi_3[A, B, C, \pi]Q$. Denote $\psi_j[A, B, C, \pi]$ and $\phi_j[A, B, C, \pi]$ as $\psi_j[\pi]$ and $\phi_j[\pi]$ respectively. Thus $\psi_j[\pi] = r\phi_j[\pi]$.

We now define some events, corresponding to the adversary making a password guess against a participant instance, and against three participant instances that are partnered in an **Execute** query. In each case, we also define an associated value for the event, and we note that the

associated value is actually fixed by the protocol before the event occurs.

- **testpw**(U,i,V,W,π) : this is the event that the adversary makes a password guess against Π_i^U with $pid_U = < V, W >$. Let $\{U,V,W\} = \{A,B,C\}$ where A is the initiator, B is the second participant and C is the third participant. For some $m,\mu,\upsilon,\lambda_A,\lambda_B$ and λ_C, $\mathcal{A}$ makes an $H(< A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C >)$ query, and if $U = A$, $\mathcal{A}$ makes a **Participant** U **Action 0** query with input $< B,C >$ and output $< A,m >$, and a **Participant** U **Action 1** query with input $< B,\mu,C,\upsilon >$ to Π_i^U. Otherwise, if $U = B$, $\mathcal{A}$ makes a **Participant** U **Action 0** query with input $< A,m >$ and output $< B,\mu >$, and a **Participant** U **Action 1** query with input $< C,\upsilon >$ to Π_i^U. Otherwise, since $U = C$, $\mathcal{A}$ makes a **Participant** U **Action 0** query with input $< A,m.B,\mu >$ and output $< C,\upsilon >$ to Π_i^U. $\mathcal{A}$ also makes an $f_1(A,B,C,\pi)$ query returning $\phi_1[\pi]Q$, an $f_2(A,B,C,\pi)$ query returning $\phi_2[\pi]Q$, an $f_3(A,B,C,\pi)$ query returning $\phi_3[\pi]Q$, where $\sigma = BDH(\alpha,\beta,\gamma)$, $m \leftarrow \alpha + \lambda_A$, $\mu \leftarrow \beta + \lambda_B$, $\upsilon \leftarrow \gamma + \lambda_C$, $\lambda_A \leftarrow \phi_1[\pi]Q$, $\lambda_B \leftarrow \phi_2[\pi]Q$, $\lambda_C \leftarrow \phi_3[\pi]Q$, ACCEPTABLE$(m)$, ACCEPTABLE$(\mu)$, and ACCEPTABLE$(\upsilon)$. The event's associated value is $sk_U^i = H(< A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C >)$.
- **testexecpw**(A,i,B,j,C,l,π) : this is the event that the adversary makes a password guess against three instances that are partnered in an **Execute** query. For some $m,\mu,\upsilon,\lambda_A,\lambda_B$ and λ_C, $\mathcal{A}$ makes an $H(< A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C >)$ query, and previously $\mathcal{A}$ made an **Execute** (A,i,B,j,C,l,π) query that generated m,μ,υ, and $f_1(A,B,C,\pi)$, $f_2(A,B,C,\pi)$ and $f_3(A,B,C,\pi)$ queries returning $\phi_1[\pi]Q$, $\phi_2[\pi]Q$, and $\phi_3[\pi]Q$, where $\lambda_A \leftarrow \phi_1[\pi]Q$, $\lambda_B \leftarrow \phi_2[\pi]Q$, $\lambda_C \leftarrow \phi_3[\pi]Q$, $\sigma = BDH(\alpha,\beta,\gamma)$, $m \leftarrow \alpha + \lambda_A$, $\mu \leftarrow \beta + \lambda_B$, and $\upsilon \leftarrow \gamma + \lambda_C$. The associated value of this event is $sk_A^i = sk_B^j = sk_C^l = H(< A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C >)$
- **correctpw** : a **testpw**(U,i,V,W,π_{UVW}) event occurred, for some U,i,V,W where π_{UVW} is the password shared between U,V and W.
- **correctpwexec** : a **testexecpw**(A,i,B,j,C,l,π_{ABC}) event occurred for some A,i,B,j,C and l, where π_{ABC} is the password shared between A, B and C.
- **triplepw**(U) : a **testpw**(U,i,V,W,π) event, a **testpw**(U,i,V,W,π') event and a **testpw**$(U,i,V,W,\tilde{\pi})$ occurred, for some $U,i,V,W,\pi,\hat{\pi}$ and $\tilde{\pi}$, with $\pi \neq \hat{\pi} \neq \tilde{\pi} \neq \pi$.

Protocol P_1. Let E_1 be the event that an m value generated in a **Participant** A **Action 0** or **Execute** query is equal to an m value generated in a previous **Participant** A **Action 0** or **Execute** query, an m value sent as input in a previous **Participant** B **Action 0** or **Participant** C **Action 0** query, or m in a previous $f_i(\cdot)$ query (made by the adversary). Let E_2 be the event that a μ value generated in a **Participant** B **Action 0** or **Execute** query

is equal to a μ value generated in a previous **Participant** B **Action 0** or **Execute** query, a μ sent as input in a previous **Participant** A **Action 1** or **Participant** C **Action 0**, or μ value in a previous $f_i(\cdot)$ query (made by the adversary). Let E_3 be the event that a v value generated in a **Participant** C **Action 0** or **Execute** query is equal to a v value generated in a previous **Participant** C **Action 0** or **Execute** query, a v sent as input in a previous **Participant** A **Action 1** or **Participant** B **Action 1** query, or v value in a previous $f_i(\cdot)$ query (made by the adversary). Let $E = E_1 \vee E_2 \vee E_3$. Let P_1 be a protocol that is identical to P_0 except that if E occurs, the protocol aborts (and thus the adversary fails).

Claim 4.2 For an adversary $\mathcal{A}$

$$Adv^{ake}_{P_0}(\mathcal{A}) \le Adv^{ake}_{P_1}(\mathcal{A}) + \frac{O((n_{se}+n_{ex})(n_{ro}+n_{se}+n_{ex}))}{q}.$$

Proof : Consider the last m, μ or v value generated. There is a probability of no more than $(n_{ro}+n_{se}+n_{ex})/q$ that this value has previously been generated in a **Send**, **Execute** or Random oracle query. There are $n_{se}+n_{ex}$ values that are required to be unique if event E is not to occur. Hence the probability of any of the m, μ or v values not being unique is $O((n_{se}+n_{ex})(n_{ro}+n_{se}+n_{ex}))/q$, and the theorem follows. □

Protocol P_2. Let P_2 be a protocol that is identical to P_1 except that **Send** and **Execute** queries are answered without making any random oracle queries, and subsequent random oracle queries by the adversary are back patched, as much as possible, to be consistent with the responses to the **Send** and **Execute** queries. Specifically, the queries in P_2 are changed as follows:

- In a **Execute**(A,i,B,j,C,l) query, $m \leftarrow \tau[i,A]Q$, where $\tau[i,A]Q \xleftarrow{R} Z_q$, $\mu \leftarrow \tau[j,B]Q$, where $\tau[j,B]Q \xleftarrow{R} Z_q$, $v \leftarrow \tau[l,C]Q$, where $\tau[l,C]Q \xleftarrow{R} Z_q$, and $sk^i_A \leftarrow sk^j_B \leftarrow sk^l_C \xleftarrow{R} \{0,1\}^\kappa$.
- In a **Participant** A **Action 0** query to instance Π^A_i, $m \leftarrow \tau[i,A]Q$, where $\tau[i,A]Q \xleftarrow{R} Z_q$.
- In a **Participant** B **Action 0** query to instance Π^B_i, $\mu \leftarrow \tau[j,B]Q$, where $\tau[j,B]Q \xleftarrow{R} Z_q$.
- In a **Participant** C **Action 0** query to instance Π^C_l, $v \leftarrow \tau[l,C]Q$, where $\tau[l,C]Q \xleftarrow{R} Z_q$, and $sk^l_C \xleftarrow{R} \{0,1\}^\kappa$.
- In **a Participant** A **Action 1** query to instance Π^A_i, if Π^C_l is paired with instance Π^A_i and Π^B_j, $sk^i_A \leftarrow sk^j_B \leftarrow sk^l_C$, else if Π^C_l is paired with instance Π^A_i, $sk^i_A \leftarrow sk^l_C$, else if Π^B_j is paired with instance Π^A_i and has a session key sk^j_B, $sk^i_A \leftarrow sk^j_B$, else if this query causes a **testpw**(A,i,B,C,π_{ABC}) event to occur, set sk^i_A to the value associated with that event, else set $sk^l_C \xleftarrow{R} \{0,1\}^\kappa$.
- In a **Participant** B **Action 1** query to instance Π^B_j, if Π^C_l is paired with instance Π^A_i and Π^B_j, $sk^j_A \leftarrow sk^j_B \leftarrow sk^l_C$, else if Π^C_l is paired with instance Π^B_j, $sk^j_B \leftarrow sk^l_C$, else if Π^A_i is paired with in-

stance Π_j^B and has a session key sk_A^i, $sk_B^j \leftarrow sk_A^i$, else if this query causes a **testpw**(B,j,A,C,π_{ABC}) event to occur, set sk_B^j to the value associated with that event, else set $sk_B^j \xleftarrow{R} \{0,1\}^\kappa$.

- In a $H(< A,B< C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C >)$ query, if this $H(\cdot)$ causes a **testpw** (A,i,B,C,π_{ABC}), **testpw** (B,j,A,C,π_{ABC}), **testpw**(C,l,A,B,π_{ABC}), or **testexecpw** (A,i,B,j,C,l,π_{ABC}) event to occur, output the associated value of that event, else output a random value from $\{0,1\}^\kappa$.

Note that we can determine whether the appropriate event occurred using the $\phi_1[\pi]$, $\phi_2[\pi]$, $\phi_3[\pi]$, and τ values. Also note that by P_1 and the fact that a participant instance that is paired with any participant C instance copies the session key of the participant C instance (or, if there is no paired participant C instance, then it copies the key of its partner, if such a partner exists), there will never be more than one associated value that needs to be considered in the $H(\cdot)$ query.

Claim 4.3 For any adversary $\mathcal{A}$,

$$Adv_{P_1}^{ake}(\mathcal{A}) = Adv_{P_2}^{ake}(\mathcal{A}) + \frac{O(n_{ro})}{q}.$$

Proof : In P_1, participant instance Π_l^C creates a session key sk_C^l that is uniformly chosen from $\{0,1\}^\kappa$, independent of anything that previously occurred, since the $H(\cdot)$ query that determines sk_C^l is new. Also in P_1, for any participant A and B instances Π_i^A and Π_j^B that have had an Action 1 query, either:

1. exactly one instance Π_l^C is paired with Π_i^A and Π_j^B, in which case $sk_C^l = sk_A^i = sk_B^j$, or
2. only one instance Π_l^C is paired with Π_i^A or Π_j^B, in which case $sk_A^i = sk_C^l$ or $sk_B^j = sk_C^l$, or
3. no instance Π_l^C is paired with Π_i^A and/or Π_j^B, and Π_i^A and Π_j^B may or may not be paired with each other.

In both of these cases, either a **testpw**(A,i,B,C,π_{ABC}) or **testpw**(B,j,A,C,π_{ABC}) event occurs, and sk_A^i or sk_B^j is the value associated with that event (i.e., the output of the previous $H(\cdot)$ query associated with that event) or sk_A^i and sk_B^j are uniformly chosen from $\{0,1\}^\kappa$, independent of anything that previously occurred, since the $H(\cdot)$ query that determines sk_A^i and sk_B^j is new.

Finally, for any $H(< A,B,C,\cdot,\cdot,\cdot,\cdot,\lambda_A,\lambda_B,\lambda_C >)$ query, either (1) it causes a **testpw**(A,i,B,C,π_{ABC}), **testpw**(B,j,A,C,π_{ABC}), **testpw**(C,l,A,B,π_{ABC}) or **testexecpw**(A,i,B,j,C,l,π_{ABC}) event to occur, in which case the output is the associated value of that event, (2) $\lambda_A \leftarrow r \cdot f_1(A,B,C,\pi_{ABC})$, $\lambda_B \leftarrow r \cdot f_2(A,B,C,\pi_{ABC})$, and $\lambda_C \leftarrow r \cdot f_3(A,B,C,\pi_{ABC})$, but the adversary has not made $f_1(A,B,C,\pi_{ABC})$, $f_2(A,B,C,\pi_{ABC})$, and $f_3(A,B,C,\pi_{ABC})$ queries, or (3) the output of $H(\cdot)$ query is uniformly chosen from $\{0,1\}^\kappa$, independent of anything that previously occurred, since this is a new $H(\cdot)$ query.

If the second case for the $H(\cdot)$ query described above occurs, P_1 may be inconsistent with P_2, since the key associated with the relevant session may need to have been returned by P_2, instead of a random value. However, the probability of the adversary correctly guessing the values of λ_A, λ_B and λ_C in an $H(\cdot)$ query is less than $1/q$. Thus the total probability of an $H(\cdot)$ query causing the second case above is bounded by n_{ro}/q. If this case never occurs, then P_2 is consistent with P_1. Then the claim follows. □

Protocol P_3. Let P_3 be a protocol that is identical to P_3 except that in an $H(<A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C>)$ query, there is no check for a **testexecpw**(A,i,B,j,C,l,π_{ABC}) event.

Claim. 4.4 For any adversary $\mathcal{A}$ running in time t, there is a $t'=O(t+(n_{ro}+n_{ex})t_{op})$ such that

$$Adv^{ake}_{P_2}(\mathcal{A}) \le Adv^{ake}_{P_3}(\mathcal{A}) + 2Adv^{BDH}_{G_1G_2}(t', n_{ro}).$$

Proof : Let E be the event that a **correctpwexec** event occurs. If E does not occur, then P_2 and P_3 are indistinguishable. Let ϵ be the probability that E occurs when $\mathcal{A}$ is running against protocol P_2. Then $Pr(Succ^{ake}_{P_2}(\mathcal{A})) \le Pr(Succ^{are}_{P_3}(\mathcal{A})) + \epsilon$, and thus by Fact 2.1, $Adv^{ake}_{P_2}(\mathcal{A}) \le Adv^{ake}_{P_3}(\mathcal{A}) + 2\epsilon$

Now we construct an algorithm $\mathcal{D}$ that attempts to solve BDH by running $\mathcal{A}$ on a simulation of the protocol. Given (X, Y, Z), $\mathcal{D}$ simulates P_3 for $\mathcal{A}$ with these changes.

1. In an **Execute**(A,i,B,j,C,l) query, set $m \leftarrow X+\rho_{i,A}Q$, $\mu \leftarrow X+\rho_{j,B}Q$, $\upsilon \leftarrow Z+\rho_{l,C}Q$, where $\rho_{i,A}, \rho_{j,B}, \rho_{l,C} \overset{R}{\leftarrow} Z_1$.
2. When $\mathcal{A}$ finishes, for every $H(<A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C>)$ query, where m,μ and υ were generated in an **Execute** (A,i,B,j,C,l) query and a $f_1(A,B,C,\pi_{ABC})$ query returned $\phi_1[\pi]Q$ and an $f_2(A,B,C,\pi_{ABC})$ query returned $\phi_2[\pi]Q$, an $f_3(A,B,C,\pi_{ABC})$ query returned $\phi_3[\pi]Q$, and $\lambda_A \leftarrow r\phi_1[\pi]Q$, $\lambda_B \leftarrow r\phi_2[\pi]Q$, $\lambda_C \leftarrow r\phi_3[\pi]Q$, add

$$\sigma e(X,Z)^{\psi_3[\pi]-\rho_{l,C}} e(\upsilon,\ (\rho_{j,B}-\psi_2[\pi])X + (\rho_{i,A}-\psi_1[\pi])Y)^{-1} e(\upsilon,\ (\rho_{j,B}-\psi_2[\pi])Q)^{\psi_1[\pi]-\rho_{i,A}}$$
$$e\big((\rho_{j,B}-\psi_2[\pi])X + (\rho_{i,A}-\psi_1[\pi])Y,\ \psi_3[\pi]Q\big)$$
$$e\big((\rho_{j,B}-\psi_2[\pi])Q,\ (\rho_{i,A}-\psi_1[\pi])Q\big)^{\psi_3[\pi]}$$

to the list of possible values for $BDH(X, Y, Z)$, where $\psi_i[\pi]=r\phi_i[\pi]$ for $i=1,2,3$.

This simulation is perfectly indistinguishable from P_3 until E occurs, and in this case, $\mathcal{D}$ adds the correct $BDH(X,Y,Z)$ to the list. After E occurs the simulation may be distinguishable from P_3, but this does not change the fact that E occurs with probability ϵ. However, we do make the assumption that $\mathcal{A}$ still follows the appropriate time and query bounds (or at least that the simulator can stop $\mathcal{A}$ from exceeding these bounds), even if $\mathcal{A}$ distinguishes the simulation from P_3.

$\mathcal{D}$ creates a list of size n_{ro}, and its advantage is ϵ. Let t' be the running time of $\mathcal{D}$, and note that $t'=O(t+(n_{ro}+n_{ex})t_{op})$ The claim follows from the fact that

$$Adv(\mathcal{D}) \le Adv^{BDH}_{G_1G_2}(t', n_{ro}).$$ □

Protocol P_4. Let P_4 be a protocol that is identical to P_3 except that if **correctpw** occurs then the protocol halts and the adversary automatically succeeds. Note that the protocol already checks for a **correctpw** event, in the **Participant** A **Action 1** or **Participant** B **Action 1** query to determine if the session key has already been determined, and in the $H(\cdot)$query, to see if the output has already been determined.

Claim 4.5 For any adversary $\mathcal{A}$,

$$Adv_{P_3}^{ake}(\mathcal{A}) \le Adv_{P_4}^{ake}(\mathcal{A})$$

Proof : Obvious. □

Note that in P_4, until **correctpw** occurs, an $H(<A,B,C,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot>)$ query will output a value uniformly chosen from $\{0,1\}^\kappa$, and the session key for an unpaired client instance will be uniformly chosen from $\{0,1\}^\kappa$.

Protocol P_5. Let P_5 be a protocol that is identical to P_4 except that if **triplepw**($\mathcal{A}$) occurs, the protocol halts and the adversary fails. We assume that when a query is made, the test for **triplepw**($\mathcal{A}$) occurs before the test for **correctpw**.

Claim 4.6 For any adversary $\mathcal{A}$ running in time t, there is a $t' = O(t+(n_{ro}^3+n_{se}+n_{ex})t_{op})$ such that

$$Adv_{P_4}^{ake}(\mathcal{A}) \le Adv_{P_5}^{ake}(\mathcal{A}) + 9Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$$

Proof: Let ϵ be the probability that the **triplepw** (A) event occurs when $\mathcal{A}$ is running against protocol P_4. Then $Pr(Succ_{P_4}^{ake}(\mathcal{A})) \le Pr(Succ_{P_5}^{are}(\mathcal{A})) + \epsilon$, and thus by Fact 2.1, $Adv_{P_4}^{ake}(\mathcal{A}) \le Adv_{P_5}^{ake}(\mathcal{A}) + 2\epsilon$.

Now we construct an algorithm $\mathcal{D}$ that attempts to solve BDH by running $\mathcal{A}$ on a simulation of the protocol. Given (X,Y,Z), $\mathcal{D}$ simulates P_4 for $\mathcal{A}$ with these changes:

1. In an $f_2(A,B,C,\pi)$ query and $f_3(A,B,C,\pi)$ query, set $f_2(A,B,C,\pi) = \psi_2[\pi]Y + \psi_2'[\pi]Q$ where $\psi_2'[\pi] \xleftarrow{R} Z_q$, $f_3(A,B,C,\pi)$ $\psi_3[\psi]Z + \psi_3'[\pi]Q$, where $\psi_3'[\pi] \xleftarrow{R} Z_q$, and $(\psi_2[\pi],\psi_3[\pi]) \in_R \{(0,1),(2,0),(0,2)\}$
2. In a **Participant** $\mathcal{A}$ **Action 0** query to a Participant instance Π_i^A with input $<B,C>$, set
$$m \leftarrow X + \rho_{i,A}Q$$
3. Tests for **correctpw** (from P_4) are not made.
4. When $\mathcal{A}$ finishes, for every triple of queries
$H(<A,B,C,m,\mu,\upsilon,\sigma,\lambda_A,\lambda_B,\lambda_C>)$,
$H(<A,B,C,m,\mu,\upsilon,\hat{\sigma},\hat{\lambda}_A,\hat{\lambda}_B,\hat{\lambda}_C>)$,
$H(<A,B,C,m,\mu,\upsilon,\tilde{\sigma},\tilde{\lambda}_A,\tilde{\lambda}_B,\tilde{\lambda}_C>)$,

where ACCEPTABLE(σ), ACCEPTABLE($\hat{\sigma}$) and ACCEPTABLE($\tilde{\sigma}$) are true, and there was a **Participant** $\mathcal{A}$ **Action 0** query to a participant instance Π_i^A with input $<B,C>$ and output $<A,m>$, a **Participant** $\mathcal{A}$ **Action 1** query to Π_i^A with input $<B,\mu,C,\upsilon>$, an $f_k(A,B,C,\pi)$ query that returned λ_k, an $f_k(A,B,C,\hat{\pi})$ query that returned $\hat{\lambda}_k$, and an $f_k(A,B,C,\tilde{\pi})$ query that returned $\tilde{\lambda}_k$, for $k \in \{1,2,3\}$, add

$$\left(\sigma^2\hat{\sigma}^{-1}\tilde{\sigma}^{-1}e(\mu,\upsilon)^{r(2\phi_1[\pi]-\phi_1[\hat{\pi}]-\phi_1[\tilde{\pi}]}e(X,\mu)^{r(2\phi_3[\pi]-\phi_3[\hat{\pi}]-\phi_3[\tilde{\pi}])}\right.$$
$$e(X,\upsilon)^{r(2\phi_2[\pi]-\phi_2[\hat{\pi}]-\phi_2[\tilde{\pi}])}e(Y,\upsilon)^{-2r^2(\phi_1[\pi]-\phi_1[\tilde{\pi}])}$$
$$e(Z,\mu)^{-2r^2(\phi_1[\pi]-\phi_1[\tilde{\pi}])}e(X,Y)^{-2r^2(\phi_3[\pi]-\phi_3[\tilde{\pi}])}$$

$$e(X,Z)^{-2r^2(\phi_2[\pi]-\phi_2[\pi])}e(Y,Z)^{2r^2(r\phi_1[\pi]-\rho_{i,A})}$$
$$e(Q,\mu)^{r\rho_{i,A}(2\phi_3[\pi]-\phi_3[\hat{\pi}]-\phi_3[\pi])-r^2(2\phi_1[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_3[\pi])}$$
$$e(Q,\upsilon)^{r\rho_{i,A}(2\phi_2[\pi]-\phi_2[\hat{\pi}]-\phi_2[\pi])-r^2(2\phi_1[\pi]\phi_2[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi])}$$
$$e(Q,X)^{-r^2(2\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_2[\pi]\phi_3[\pi])}$$
$$e(Q,Y)^{-2r^2\rho_{i,A}(\phi_3[\pi]-\phi_3[\hat{\pi}])+2r^3(\phi_1[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_3[\hat{\pi}])}$$
$$e(Q,Z)^{-2r^2\rho_{i,A}(\phi_2[\pi]-\phi_2[\pi])+2r^3(\phi_1[\pi]\phi_2[\pi]-\phi_1[\pi]\phi_2[\pi])}$$
$$e(Q,Q)^{-r^2\rho_{i,A}(2\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_2[\pi]\phi_3[\pi])}$$
$$e(Q,Q)^{r^3(2\phi_1[\pi]\phi_2[\pi]\phi_3[\pi])-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi]\phi_3[\pi])}\Big)^{\frac{1}{2r^2}}$$

to the list of possible values of $BDH(X, Y, Z)$.

This simulation is perfectly indistinguishable from P_4 until a **triplepw**($\mathcal{A}$) event or a **correctpw** event occurs. If a **triplepw**($\mathcal{A}$) event occurs, then with probability 2/9 it occurs for three passwords π, $\hat{\pi}$, and $\tilde{\pi}$ with $\{(\psi_2[\pi],\psi_3[\pi]),(\psi_2[\hat{\pi}],\psi_3[\hat{\pi}]),(\psi_2[\tilde{\pi}],\psi_3[\tilde{\pi}])\} = \{(1,1),(0,2),(2,0)\}$, in this case $\mathcal{D}$ adds the correct $BDH(X, Y, Z)$ to the list. If a **correctpw** event occurs before a **triplepw**(A) event occurs, then the **triplepw**(A) event would never have occurred in P_4, since P_4 would halt. Note that in this case, the simulation may be distinguishable from P_4, but this does not change the fact that a **triplepw**(A) event will occur with probability at least ϵ in the simulation. However, we do make the assumption that $\mathcal{A}$ still follows the appropriate time and query bounds (or at least that the simulation can stop $\mathcal{A}$ from exceeding these bounds), even if $\mathcal{A}$ distinguished the simulation from P_4.

$\mathcal{D}$ creates a list of size less than n_{ro}^3, and its advantage is $(2/9)\epsilon$. Let t' be the running time of $\mathcal{D}$, and note that $t' = O(t+(n_{ro}^3+n_{se}+n_{ex})t_{op})$. Then the claim follows from the fact that $Adv_{G_1G2}^{BDH}(\mathcal{D}) \le Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$. □

Protocol P_6. Let P_6 be a protocol that is identical to P_5 except that if **triplepw**(B) occurs, the protocol halts and the adversary fails. We assume that when a query is made, the test for **triplepw**(B) occurs before the test for **correctpw**.

Claim 4.7 For any adversary $\mathcal{A}$ running in time t, there is a $t' = O(t+(n_{ro}^3+n_{se}+n_{ex})t_{op})$ such that

$$Adv_{P_5}^{ake}(\mathcal{A}) \le Adv_{P_6}^{ake}(\mathcal{A}) + 9Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$$

Proof: Let ϵ be the probability that the **triplepw** (B) occurs when $\mathcal{A}$ is running against protocol P_5. Then $Pr(Succ_{P_5}^{ake}(\mathcal{A})) \le Pr(Succ_{P_6}^{are}(\mathcal{A})) + \epsilon$ and thus by Fact 2.1, $Adv_{P_5}^{ake}(\mathcal{A}) \le Adv_{P_6}^{ake}(\mathcal{A}) + 2\epsilon$.

Now we construct an algorithm $\mathcal{D}$ that attempts to solve BDH by running $\mathcal{A}$ on a simulation of the protocol. Given (X, Y, Z), $\mathcal{D}$ simulates P_5 for $\mathcal{A}$ with these changes:

1. In an $f_1(A, B, C, \pi)$ query to participant A, and $f_3(A, B, C, \pi)$ query to participant C, set
   $f_1(A, B, C, \pi) = \psi_1[\pi]X + \psi_1'[\pi]Q$, where $\psi_1'[\pi] \underset{\leftarrow}{R} Z_q$,
   $f_3(A, B, C, \pi) = \psi_3[\pi]X + \psi_3'[\pi]Q$, where $\psi_3'[\pi] \underset{\leftarrow}{R} Z_q$,
   and $(\psi_1[\pi],\psi_3[\pi]) \in_R \{(0,1),(2,0),(0,2)\}$.
2. In a **Participant** B **Action 0** query to a Participant instance Π_j^B with input $< A, m >$, where ACCEPTABLE (m) is ture, set
   $$\mu \leftarrow Y + \rho_{j,B}Q$$
3. Tests for **correctpw**(from P_4) and **triplepw** ($\mathcal{A}$) (from P_5) are not made.
4. When $\mathcal{A}$ finishes, for every triple of queries

$H(< A, B, C, m, \mu, \upsilon, \sigma, \lambda_A, \lambda_B, \lambda_C >)$,
$H(< A, B, C, m, \mu, \upsilon, \hat{\sigma}, \hat{\lambda}_A, \hat{\lambda}_B, \hat{\lambda}_C >)$,
$H(< A, B, C, m, \mu, \upsilon, \tilde{\sigma}, \tilde{\lambda}_A, \tilde{\lambda}_B, \tilde{\lambda}_C >)$,

where ACCEPTABLE(σ), ACCEPTABLE($\hat{\sigma}$) and ACCEPTABLE($\tilde{\sigma}$) are true, and there was a **Participant** B **Action 0** query to a participant instance Π_j^B with input $< A, m >$ and output μ, a **Participant** B **Action 1** query to Π_j^B with input $< C, \mu >$, an $f_k(A, B, C, \pi)$ query that returned λ_K, an $f_k(A, B, C, \hat{\pi})$ query that returned $\hat{\lambda}_k$, and an $f_k(A, B, C, \tilde{\pi})$ query that returned $\tilde{\lambda}_k$, for $k \in \{1,2,3\}$, add

$$\Big(\sigma^2\hat{\sigma}^{-1}\tilde{\sigma}^{-1}e(m,\upsilon)^{r(2\phi_2[\pi]-\phi_2[\hat{\pi}]-\phi_2[\pi]}e(m,Y)^{r(2\phi_3[\pi]-\phi_3[\hat{\pi}]-\phi_3[\pi])}$$
$$e(Y,\upsilon)^{r(2\phi_1[\pi]-\phi_1[\hat{\pi}]-\phi_1[\pi])}e(X,\upsilon)^{-2r^2(\phi_2[\pi]-\phi_2[\pi])}$$
$$e(Z,m)^{-2r^2(\phi_2[\pi]-\phi_2[\pi])}e(X,Y)^{-2r^2(\phi_3[\pi]-\phi_3[\hat{\pi}])}$$
$$e(Y,Z)^{-2r^2(\phi_1[\pi]-\phi_1[\pi])}e(X,Z)^{2r^2(r\phi_2[\pi]-\rho_{i,B})}$$
$$e(Q,m)^{r\rho_{j,B}(2\phi_3[\pi]-\phi_3[\hat{\pi}]-\phi_3[\pi]-r^2(2\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_2[\pi]\phi_3[\pi])}$$
$$e(Q,\upsilon)^{r\rho_{j,B}(2\phi_1[\pi]-\phi_1[\hat{\pi}]-\phi_1[\pi])-r^2(2\phi_1[\pi]\phi_2[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi])}$$
$$e(Q,Y)^{-r^2(2\phi_1[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_3[\pi])}$$
$$e(Q,X)^{-2r^2\rho_{j,B}(\phi_3[\pi]-\phi_3[\hat{\pi}])+2r^3(\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}])}$$
$$e(Q,Z)^{-2r^2\rho_{j,B}(\phi_1[\pi]-\phi_1[\pi])+2r^3(\phi_1[\pi]\phi_2[\pi]-\phi_1[\pi]\phi_2[\pi])}$$
$$e(Q,Q)^{-r^2\rho_{j,B}(2\phi_1[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_3[\pi])}$$
$$e(Q,Q)^{r^3(2\phi_1[\pi]\phi_2[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi]\phi_3[\pi])}\Big)^{\frac{1}{2r^2}}$$

to the list of possible values of $BDH(X, Y, Z)$.

This simulation is perfectly indistinguishable from P_5 until a **triplepw**(A) event, **triplepw**(B) or a **correctpw** event occurs. If a **triplepw**(B) event occurs, then with probability 2/9 it occurs for three passwords $\pi, \hat{\pi}$, and $\tilde{\pi}$ with $\{(\psi_2[\pi], \psi_3[\pi]), (\psi_2[\hat{\pi}], \psi_3[\hat{\pi}]), (\psi_2[\tilde{\pi}], \psi_3[\tilde{\pi}])\} = \{(1,1), (0,2), (2,0)\}$. in this case adds the correct $BDH(X, Y, Z)$ to the list. If a **triplepw**(A) or a **correctpw** event occurs before a **triplepw**(B) event occurs, then the **triplepw**(B) event would never have occurred in P_5, since P_5 would halt. Note that in this case, the simulation may be distinguishable from P_5, but this does not change the fact that a **triplepw**(B) event will occur with probability at least ϵ in the simulation. However, we do make the assumption that $\mathcal{A}$ still follows the appropriate time and query bounds (or at least that the simulation can stop $\mathcal{A}$ from exceeding these bounds), even if $\mathcal{A}$ distinguished the simulation from P_5.

$\mathcal{D}$ creates a list of size less than n_{ro}^3, and its advantage is $(2/9)\epsilon$. Let t' be the running time of $\mathcal{D}$, and note that $t' = O(t + (n_{ro}^3 + n_{se} + n_{ex})t_{op})$. Then the claim follows from the fact that $Adv_{G_1G2}^{BDH}(\mathcal{D}) \le Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$. □

Protocol P_7. Let P_7 be a protocol that is identical to P_6 except that if **triplepw**(C) occurs, the protocol halts and the adversary fails. We assume that when a query is made, the test for **triplepw**(C) occurs before the test for **correctpw**.

Claim 4.8 For any adversary $\mathcal{A}$ running in time t, there is a $t' = O(t + (n_{ro}^3 + n_{se} + n_{ex})t_{op})$ such that

$$Adv_{P_6}^{ake}(\mathcal{A}) \le Adv_{P_7}^{ake}(\mathcal{A}) + 9Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$$

Proof: Let ϵ be the probability that the **triplepw** (C) occurs when $\mathcal{A}$ is running against protocol P_7. Then $Pr(Succ_{P_6}^{ake}(\mathcal{A})) \le Pr(Succ_{P_7}^{are}(\mathcal{A})) + \epsilon$ and thus by Fact 2.1,

$$Adv_{P_6}^{ake}(\mathcal{A}) \leq Adv_{P_7}^{ake}(\mathcal{A}) + 2\epsilon.$$

Now we construct an algorithm $\mathcal{D}$ that attempts to solve BDH by running $\mathcal{A}$ on a simulation of the protocol. Given (X, Y, Z), $\mathcal{D}$ simulates P_6 for $\mathcal{A}$ with these changes:

1. In an $f_1(A,B,C,\pi)$ query to participant A, and $f_2(A,B,C,\pi)$ query to participant B, set
   $f_1(A,B,C,\pi) = \psi_1[\pi]X + \psi_1'[\pi]Q$, where $\psi_1'[\pi] \xleftarrow{R} Z_q$,
   $f_2(A,B,C,\pi) = \psi_2[\pi]X + \psi_2'[\pi]Q$, where $\psi_2'[\pi] \xleftarrow{R} Z_q$,
   and $(\psi_1[\pi], \psi_2[\pi]) \in_R \{(0,1),(2,0),(0,2)\}$.
2. In a **Participant** C **Action 0** query to a Participant instance Π_l^C with input $< A,m,B,\mu >$, where ACCEPTABLE (m) and ACCEPTABLE(μ) is true, set $v \leftarrow Z + \rho_{l,c}Q$
3. Tests for **correctpw**(from P_4), **triplepw**(A) (from P_5), and **triplepw**(B) (from P_6) are not made.
4. When $\mathcal{A}$ finishes, for every triple of queries
   $H(< A,B,C,m,\mu,v,\sigma,\lambda_A,\lambda_B,\lambda_C >)$,
   $H(< A,B,C,m,\mu,v,\hat{\sigma},\hat{\lambda}_A,\hat{\lambda}_B,\hat{\lambda}_C >)$,
   $H(< A,B,C,m,\mu,v,\tilde{\sigma},\tilde{\lambda}_A,\tilde{\lambda}_B,\tilde{\lambda}_C >)$,

where ACCEPTABLE(σ), ACCEPTABLE$(\hat{\sigma})$ and ACCEPTABLE$(\tilde{\sigma})$ are true, and there was a **Participant C Action 0** query to a participant instance Π_l^C with input $< A, m, B, \mu >$ and output v, an $f_k(A,B,C,\pi)$ query that returned λ_k, an $f_k(A,B,C,\hat{\pi})$ query that returned $\hat{\lambda}_k$, and an $f_k(A,B,C,\tilde{\pi})$ query that returned $\tilde{\lambda}_k$, for $k \in \{1,2,3\}$, add

$$\Big(\sigma^2\hat{\sigma}^{-1}\tilde{\sigma}^{-1}e(m,\mu)^{r(2\phi_3[\pi]-\phi_3[\hat{\pi}]-\phi_3[\pi]}e(m,z)^{r(2\phi_2[\pi]-\phi_2[\hat{\pi}]-\phi_2[\pi])}$$
$$e(Z,\mu)^{r(2\phi_1[\pi]-\phi_1[\hat{\pi}]-\phi_1[\pi])}e(X,\mu)^{-2r^2(\phi_3[\pi]-\phi_3[\pi])}$$
$$e(Y,m)^{-2r^2(\phi_3[\pi]-\phi_3[\pi])}e(X,Z)^{-2r^2(\phi_2[\pi]-\phi_2[\hat{\pi}])}$$
$$e(Y,Z)^{-2r^2(\phi_1[\pi]-\phi_1[\pi])}e(X,Y)^{2r^2(r\phi_3[\pi]-\rho_{l,C})}$$
$$e(Q,m)^{r\rho_{l,C}(2\phi_2[\pi]-\phi_2[\hat{\pi}]-\phi_2[\pi]-r^2(2\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_2[\pi]\phi_3[\pi])}$$
$$e(Q,\mu)^{r\rho_{l,C}(2\phi_1[\pi]-\phi_1[\hat{\pi}]-\phi_1[\pi])-r^2(2\phi_1[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_3[\pi])}$$
$$e(Q,Z)^{-r^2(2\phi_1[\pi]\phi_2[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]-\phi_1[\pi]\phi_1[\pi])}$$
$$e(Q,X)^{-2r^2\rho_{l,C}(\phi_2[\pi]-\phi_2[\hat{\pi}])+2r^3(\phi_2[\pi]\phi_3[\pi]-\phi_2[\hat{\pi}]\phi_3[\hat{\pi}])}$$
$$e(Q,Y)^{-2r^2\rho_{l,C}(\phi_1[\pi]-\phi_1[\pi])+2r^3(\phi_1[\pi]\phi_3[\pi]-\phi_1[\pi]\phi_3[\pi])}$$
$$e(Q,Q)^{-r^2\rho_{l,C}(2\phi_1[\pi]\phi_2[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi])}$$
$$e(Q,Q)^{r^3(2\phi_1[\pi]\phi_2[\pi]\phi_3[\pi]-\phi_1[\hat{\pi}]\phi_2[\hat{\pi}]\phi_3[\hat{\pi}]-\phi_1[\pi]\phi_2[\pi]\phi_3[\pi])}\Big)^{\frac{1}{2r^2}}$$

to the list of possible values of $BDH(X,Y,Z)$.

This simulation is perfectly indistinguishable from P_6 until a **triplepw**(A) event, a **triplepw**(B) event, a **triplepw**(C) event or a **correctpw** event occurs. If a **triplepw**(C) event occurs, then with probability 2/9 it occurs for three passwords $\pi, \hat{\pi}$, and $\tilde{\pi}$ with $\{(\psi_2[\pi],\psi_3[\pi]),(\psi_2[\hat{\pi}],\psi_3[\hat{\pi}]),(\psi_2[\tilde{\pi}],\psi_3[\tilde{\pi}])\} = \{(1,1),(0,2),(2,0)\}$, in this case $\mathcal{D}$ adds the correct $BDH(X,Y,Z)$ to the list. If a **triplepw**(A) event, a **triplepw**(B) event, or a **correctpw** event occurs before a **triplepw**(C) event occurs, then the **triplepw** (C) event would never have occurred in P_6, since P_6 would halt. Note that in this case, the simulation may be distinguishable from P_6, but this does not change the fact that a **triplepw**(C) event will occur with probability at least ϵ in the simulation. However, we do make the assumption that $\mathcal{A}$ still follows the appropriate time and query bounds (or at least that the simulation can stop $\mathcal{A}$ from exceeding these bounds), even if $\mathcal{A}$ distinguished the simulation from P_6.

$\mathcal{D}$ creates a list of size less than n_{ro}^3, and

its advantage is $(2/9)\epsilon$. Let t' be the running time of $\mathcal{D}$, and note that $t' = O(t + (n_{ro}^3 + n_{se} + n_{ex})t_{op})$. Then the claim follows from the fact that $Adv_{G_1G2}^{BDH}(\mathcal{D}) \le Adv_{G_1G_2}^{BDH}(t', n_{ro}^3)$. □

Protocol P_8. Let P_8 be a protocol that is identical to P_7 except that there is a new internal oracle (i.e., not available to the adversary) that handles passwords, called a *password oracle*. This oracle generates all passwords during initialization. Then it accepts queries of the form **testpw**(π) and returns TRUE if π is correct, and FALSE otherwise. The protocol is changed only in the method for determining **correctpw**. Specifically, to test if **correctpw** occurs, whenever a **testpw**(A,i,B,C,π), a **testpw**(B,j,A,C,π) or a **testpw**(C,l,A,B,π) event occurs, a **testpw**(π) query is made to password oracle to see if π is correct.

Claim 4.9 For any adversary $\mathcal{A}$,

$$Adv_{P_7}^{ake}(\mathcal{A}) = Adv_{P_8}^{ake}(\mathcal{A})$$

Proof : By inspection, P_7 and P_8 are perfectly indistinguishable. The probability of the adversary $\mathcal{A}$ succeeding in P_8 is bounded by

$$\Pr\left(Succ_{P_8}^{ake}(\mathcal{A})\right) \le \Pr(correctpw) +$$
$$\Pr(Succ_{P_8}^{ake}(\mathcal{A}) \mid \neg correctpw)\Pr(\neg correctpw)$$

First, since there are at most $2n_{se}$ queries to the password oracle, and passwords are chosen uniformly from dictionary of size N, $Pr(correctpw) \le (2n_{se})/N$.

Now we compute $Pr(Succ_{P_8}^{ake}(\mathcal{A}) \mid \neg correctpw)$. If **correctpw** does not occur, then $\mathcal{A}$ succeeds by making a **Test** query to a fresh instance Π_i^U and guessing the bit used in that **Test** query. We will show that the view of the adversary is independent of sk_U^i, and thus the probability of success is exactly 1/2.

First we examine **Reveal** queries. Recall that since Π_i^U is fresh, there could be no **Reveal** (U,i) query, and if Π_j^U is partnered with Π_i^U, no **Reveal** (U,j) query. Second note that since *sid* includes m and μ and υ values, if more than three participant instances accept with the same *sid*, $\mathcal{A}$ fails (see P_1). Thus the output of **Reveal** queries is independent of sk_U^i.

Second we examine $H(\cdot)$ queries. As noted in the discussion following the description of P_4, an $H(\cdot)$ query returns random values independent of anything *that previously occurred*. Thus any $H(\cdot)$ queries that occurs after sk_U^i is set are independent of sk_U^i. But consider the following cases. (1) if U is the third participant, sk_U^i is chosen independently of anything that previously occurred (see P_2). (2) if U is the first or second participant, and is unpaired, sk_U^i is chosen independently of anything that previously occurred (see the discussion after P_4). (3) if U is the first or second participant and is paired, then $sk_U^i \leftarrow sk_U^j \leftarrow sk_{U'}^l$, where $\Pi_j^{U'}$ and $\Pi_l^{U''}$ are the partners of Π_i^U and U'' is the third participant and $sk_{U''}^i$ is chosen independently of anything that previously occurred (see P_2). This implies that the view of the adversary is independent of sk_U^i, and thus the probability of success is exactly 1/2.

Since $Pr(\neg correctpw) = 1 - Pr(correctpw)$, we have that

$$\Pr(Succ^{ake}_{P_i}(A)) \le \Pr(correctpw)$$

$$+ \Pr\left(Succ^{ake}_{P_i}(A) \mid \neg correctpw\right)(1 - \Pr(correctpw))$$

$$\le \Pr(correctpw) + \frac{1}{2}(1 - \Pr(correctpw))$$

$$\le \frac{1}{2} + \frac{\Pr(correctpw)}{2} \le \frac{1}{2} + \frac{n_{se}}{N}.$$

Therefore $Adv^{ake}_{P_i} \le (2n_{se})/N$. The theorem follows from this and Claims 4.2 to 4.9. □

V. Complexity comparison

In this section we compare our proposed protocol with Bresson et al.'s protocol[11] (we call BCP), N. Asokan et al.'s protocol [27](we call AG) and J. Y. Hwang et al.'s protocol[28](we call HCLB). Because these protocols are password-based group key agreement protocol, we fixed the group size of the protocol to 3 for the purpose of protocol comparison. The numbers in Table 2 shows the total cost of computation and communication required to execute the protocols once. However, the three (symmetric) encryptions/decryptions using the password as the key in BCP, the three (symmetric) encryptions and six decryptions in HCLB, and two (symmetric) encryptions/decryptions in AG are not shown. Paring computations are the most expensive among the four kinds computation, and normally scalar points multiplication on an elliptic curve is faster than exponentiation and finding a square root. Barreto et al.[12] indicated that a 512 bit pairing takes about 2.5 times as long as a 1024 bit exponentiation with a 1007 bit exponent(20ms for pairing compared to 7.9 ms for a RAS signature). Considering this assumption, the sum of pairings and exponentiations of our proposed protocol is about equal to the number of exponentiations of BCP. But AG and HCLB are more efficient in communications than the others. Our protocol requires extra computations, i.e., 3 scalar multiplications and 18 square root computations. Before the message transmission, each participant computes 1 scalar multiplication to generate a random elliptic curve point and performs 6 operations of finding square root to compute a message blinding data $\lambda_A, \lambda_B,$ and λ_C. However, $\lambda_A, \lambda_B,$ *and* λ_C can be pre-computed by each participant. Thus $m, \mu,$ and v can be pre-computed by participant A, participant B and participant C respectively. In terms of message length and the number of round and pass, our protocol is better than the others.

Table 2. The Complexity of Protocols

Protocols		Proposed	BCP	AG	HCLB
Computation	Pairings	3	-	-	-
	Scalar Multiplications	3	-	-	-
	Exponentiations	3	12	9	9
	SQRT*	18	-	-	-
Communication	Round/ Passes	1/3	3/3	4/6	2/6
	Message Length**	3|p|	9|g|	6|g|	6|g|
Precomputation		Y	N	N	Partially

* Assume one application of $f_i(A, B, C, \pi)$ loops twice in the Appendix A algorithm.

** |p| is the bit size of the finite field over which the elliptic curve is defined and |g| is the bit size of the finite field over which the DLP is defined. When the base field is GF(2), normally |p|≈ 250, and |g| ≈ 1024.

VI. Conclusion

In this paper we proposed a provably secure one round password-based tripartite key agreement protocol, which builds on Joux's protocol and adapts the PAK-EC scheme for password-based authentication.

We proved the security of the proposed protocol using the random oracle model. Our proposed protocol is better than existing protocols by BCP, AG, and HCLB in terms of message length and the number of round and pass. Although our protocol requires extra computation compared to the others, the extra part can be precomputed.

References

[1] S. Al-Riyami and K. Paterson, "Tripartite authenticated key agreement protocols from pairings," IMA Conference on Cryptography and Coding, *LNCS* vol. 2898, Springer-Verlag, pp. 332-359. 2003.

[2] P. Barreto, H. Kim, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," Advances in Cryptology - Crypto 2002, LNCS 2442, Springer-Verlag, pp.354-368, 2002.

[3] R. Barua, R. Dutta and P. Sarkar, "Provably secure authenticated tree based group key agreement protocol using pairing," *Cryptology ePrint Archive*, Report 2004/122, 2002.

[4] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks." In *EUROCRYPT 2000, LNCS* vol. 1807, pp.139-155, Springer-Verlag, 2000.

[5] M. Bellare, and P. Rogaway. "Entity authentication and key distribution." In *CRYPTO'93, LNCS* vol. 773, pp.62-73. 1993.

[6] M. Bellare, and P. Rogaway, "Provably secure session key distribution-the three party case." In *27thACM Symposium on the Theory of computing*, pp.57-66, 1995.

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

[8] S. Blake-Wilson, D. Johnson, and A. Menezes, "Key agreement protocols and their security analysis." In *proceedings of the sixth IMA International Conferences on Cryptography and Coding, LNCS* vol.1355, pp.30-45, Springer-Verlag, 1997.

[9] S. Blake-Wilson and A. Menezes, "Authenticated Diffie-Hellman key agreement protocols." In S. Tacares and H. Meijer, editors, 5th*Annual Workshop on Selected Areas in Cryptography (SAC'98), LNCS*1556, pp.339-361, Springer-Verlag, *1998*.

[10] V. Boyko, P. MacKenzie, and S. Patel, "Provably secure passwordauthentication and key exchange using Diffie-Hellman." In EUROCRYPT 2000, *LNCS* vol. *1807*, pp.156-171, *2000*.

[11] E. Bresson, O. Chevassut and D. Pointcheval, "Group Diffie-Hellman key exchange secure against dictionary attacks," Proceedings of Asiacrypt '02, *LNCS* vol. 2501, Springer-Verlag, pp. 497-514, 2002.

[12] S. Galbraith, K. Harrison and D. Soldera, "Implementing the Tate pairing," Algorithm Number Theory Symposium - ANTS V, LNCS vol. 2369, Springer-Verlag, pp. 324-337, 2002.

[13] IEEE. *IEEE1363 Standard Specifications for public key cryptography*, 2000.

[14] A. Joux, "A one round protocol for tripartite Deffie-Hellman." In W. Bosma, editor, *Proceedings of Algorithmic Number Theory Symposium - ANTS IV, LNCS* vol. 1838, pp.385-394, Springer-Verlag, 2000.

[15] J. Kate, R. Ostrovsky, ans M. Young, "Practical password-authenticated key

exchange provably secure under standard assumptions." In EUROCRYPT 2001, *LNCS* vol. 2045, pp.475-494, 2001

[16] Y. Kim, A. Perrig and G. Tsudik, "Communication-efficient group key agreement," IFIP SEC 2001, Jun 2001.

[17] L. Law, A. Menezes, M. Qu, J. Solinas, and S.A. Vanstone, "An efficient protocol for authenticated key agreement." *Technical Report CORR 98-05*, Department of C & O, University of Waterloo, 1998.

[18] L. Law, A. Menezes, M. Qu, J. Solinas, and S.A. Vanstone. "An efficient protocol for authenticated key agreement," Designs, Codes and Cryptography, vol. 28, no. 2, pp.119-134, 2003.

[19] S. Lee, Y. Kim, K. Kim and D. Ryu, "An Eficient tree based group key-agreement using bilinear map," *ACSN 2003*, China, LNCS vol. 2846, Springer-Verlag, pp.357-371, 2003

[20] P. MacKenzie, "More efficient password-authenticated key exchange." *Proceedings The Cryptographer's Track at RSA Conference, LNCS* vol. 2020, pp. 361-377, Springer-Verlag 2001.

[21] P. MacKenzie, "The PAK suit: Protocols for password-authenticated key exchange," *DIMACS Technical report* 2002-46, October 2002.

[22] D. Nalla, "ID-based tripartite key agreement with signatures," Cryptology e-Print Archive, Report 2003/144.

[23] D. Nalla and K.C. Reddy, "ID-based tripartite authenticated key agreement protocols from pairings," Cryptology ePrint Archive, Report 2003/004.

[24] K. Shim, "Efficient one-round tripartite authenticated key agreement protocol from Weil pairing," Electronic Letters 39, pp.208-209, 2003.

[25] K. Shim, "A Man-in-the-middle attack on Nalla-Reddy's ID-based tripartite authenticated key agreement protocol," Cryptology ePrint Archive, Report 2003/115.

[26] F. Zhang, S. Liu and K. Kim, "ID-based one-round authenticated tripartite key agreement protocol with pairings." Cryptology ePrint archive, Report 2002/122.

[27] N. Asokan and Philip Ginzboorg, "Key Agreement in Ad-Hoc Networks", Computer Communications, vol. 23, pp. 1627-1637, 2000.

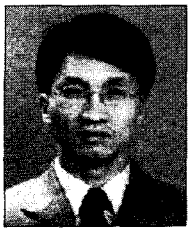
[28] J. Y. Hwang, G. Y. Choe, D. H. Lee, and J. M. Baeg, "Efficient password based Group Key Exchange Protocol," Journal of KIISC, vol. 14, no.1, pp. 59-69, 2004.

A. Computation of $f_i(A,B,C,\pi)$

The following description of $f_i(A,B,C,\pi)$ is from MacKenzie,[20] who adapted it from IEEE standards 1363[13, Appendix A.2.5].

1. Set j = 1.
2. Compute $w' = H_i(A,B,C,\pi,j)$ and $w = [w' \; AND \; (2^{|+\kappa}-2)]/2 \bmod p$
   (i.e., remove the least significant bit from w' to make w: the least order bit will be used later).
3. set $a = w^3 + aw + b \bmod p$.
4. if $a = 0$ then $f_i(A,B,C,\pi) = (w,0)$
5. find the "minimum" square root of $a \bmod p$ (for instance, using the method in [13][Appendix A.2.5]) and if it exist, call it β
6. if no square root exist, set $j := j+1$ and go to step 2.
7. Let $\gamma = w' \; AND 1$, and let $f_i(A,B,C,\pi) = (w,(-1)^{\gamma}\beta) \bmod p$.

〈著者紹介〉



이 상 곤(Sang-Gon Lee) 정회원
1986년 2월: 경북대학교 전자공학과 졸업
1988년 2월: 경북대학교 전자공학과 석사
1993년 2월: 경북대학교 전자공학과 박사
1991년 3월~1997년 2월: 창신대학 정보통신과 조교수
1997년 3월~현재: 동서대학교 인터넷공학부 조교수
2003년 8월~2004년 7월: 호주 ISRC, QUT 방문교수
〈관심분야〉 암호 프로토콜, 네트워크 보안, 자바기술

Yvonne Hitchcock
Ph.D QUT
Research Associate, Information Security Institute, QUT, QUT.
〈관심분야〉 Cryptographic Protocol

박 영 호 (Young-Ho Park) 종신회원
1989년 2월: 경북대학교 전자공학과 졸업
1991년 2월: 경북대학교 전자공학과 석사
1995년 8월: 경북대학교 전자공학과 박사
1996년 3월~현재: 상주대학교 전자전기공학부 부교수
2003년 8월~2004년 7월: Oregon State University 방문 교수
〈관심분야〉 정보보호 이론, 네트워크 보안, 광통신 보안

문 상 재(Sang-Jae Moon) 종신회원
1972년 2월: 서울대학교 공업교육과 졸업
1974년 2월: 서울대학교 전자공학과 석사
1984년 6월: 미국 UCLA 전자공학과 박사
1984년 7월~1985년 6월: UCLA Postdoc. 근무
1984년 7월~1985년 6월: 미국 OMNET 컨설턴트
1974년 2월~현재: 경북대학교 전자전기컴퓨터학부 교수
2002년 2월~현재: 한국정보보호학회 명예회장
〈관심분야〉 정보보호, 디지털 통신, 이동 네트워크