Password Based Key Exchange With Mutual Authentication

Shaoquan Jiang and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario N2L 3G1, CANADA
Email:{jiangshq,ggong}@calliope.uwaterloo.ca

August 14, 2004

Abstract. A reasonably efficient password based key exchange (KE) protocol with provable security without random oracle was recently proposed by Katz, *et al.* [18] and later by Gennaro and Lindell [13]. However, these protocols do not support mutual authentication (MA). The authors explained that this could be achieved by adding an additional flow. But then this protocol turns out to be 4-round. As it is known that a high entropy secret based key exchange protocol with MA[1] is optimally 3-round (otherwise, at least one entity is not authenticated since a replay attack is applicable), it is quite interesting to ask whether such a protocol in the password setting (without random oracle) is achievable or not. In this paper[2], we provide an affirmative answer with an efficient construction in the common reference string (CRS) model. Our protocol is even simpler than that of Katz, *et al.* Furthermore, we show that our protocol is secure under the DDH assumption (*without* random oracle).

1 Introduction

In the area of secure communications, key exchange (KE) is one of the most important issues. In this scenario, two interactive parties are assumed to hold long-term secrets. Through an interactive procedure, they establish a temporary session key and then use it to encrypt and authenticate the subsequent communication. There are two types of KE protocols in the literature. In the first case, each party holds a high entropy secret (e.g., a signing key of a digital signature). Research along this line has been well studied, see [1, 6, 8, 12]. The other case is a password authenticated key exchange protocol (see [20] for a detailed description), in which it is assumed that the two parties only share a human-memorable (low entropy) password. Unlike a high entropy secret, it is believed that an exhaustive search attack (or a dictionary attack) is feasible. In fact, it is this attack that makes a construction of a secure password based KE protocol more difficult than the high entropy secret based one.

1.1 Related Work

Password authenticated key exchange was first studied by Bellovin and Merritt [4]. Since then, it has been extensively studied in literature [5, 16]. However, none of these solutions had provable security. The first effort to achieve provable security was due to Lucks [19]. Halevi and Krawczyk [15] proposed a password KE protocol in an asymmetric setting: a user only holds a password while the

[1] we do not consider a protocol with a time stamp or a stateful protocol (e.g., a counter based protocol). In other words, we only consider protocols in which a session execution within an entity is independent of its history, and in which the network is asynchronous.

[2] An extended abstract appeared in [17]. This is the full version.

server additionally has a *private key* of a public key cryptosystem. Password KE protocols without this asymmetric assumption were proposed in [2, 7]. However, these protocols including [19] were proved in the random oracle model. It is known [9] that a random oracle based cryptographic construction could be insecure when the oracle is replaced by any real function. In the password setting, it is even worse since a minor weakness of the real function might open the door to a dictionary attack. The first solution without random oracle was due to Goldreich and Lindell [14]. Actually, their protocol was based on a general assumption only (i.e., the existence of trapdoor permutation). But this solution is very inefficient. A reasonably efficient construction in CRS model without random oracle was proposed by Katz, *et al.* [18]. We shall refer to this as the KOY protocol. An abstract framework for this protocol was proposed by Gennaro and Lindell [13]. Nevertheless, these protocols do not support mutual authentication (MA). Katz, *et al.* mentioned in their paper that a mutual authentication can be made by adding an additional flow. This is indeed true. However, the resulting protocol is then 4-round. It is known that a high entropy secret based KE protocol with MA is optimally 3-round. Thus, it is quite interesting to ask whether there exists such a protocol in the password setting *without* random oracles.

1.2 Contribution

In this paper, we provide an affirmative answer to the above problem with an explicit construction. Our construction is in the CRS model (as in [13, 18]), where all the parties have access to a set of public parameters drawn from a predetermined distribution, but nobody knows the corresponding secret key if any. Our construction is optimally 3-round. Comparing with work in [13, 18], it additionally supports mutual authentication and is also simpler than KOY protocol in the sense of exponentiation cost. Nevertheless, their work has been instructive to us. In fact, one technique in their construction helps us in authenticating the initiator. As our important contribution, we formally prove the security under the Decisional Diffie-Hellman (DDH) assumption (*without random oracles*).

2 Security Model

In this section, we introduce a formal model of security. This model is mainly adopted from Bellare, *et al.* [2] and [3]. Our difference is in the mutual authentication where we feel our definition is more reasonable. Details are provided later. The basic security model without MA was previously adopted by Katz, *et al.* [18] and Gennaro and Lindell [13]. We start with the following notations, which will be used throughout the paper.

- D: a password dictionary with a polynomial size (otherwise, it becomes a KE problem with high entropy secrets). WOLG, we assume that $D = \{1, \cdots, N\}$ with a uniform distribution for some $N > 0$.
- P_i: party i, either a client or a server. If it is a server, then it could individually share a password with a set of clients.
- $\Pi_i^{l_i}$: protocol instance l_i within party P_i. We require that l_i be unique within P_i in order to distinguish local instances. However, we do not require it is globally unique, which reflects the practical concern for possible independence of different parties.
- $Flow_i$: The ith message exchanged between two particular instances.

– $\mathbf{sid}_i^{l_i}$: the session identifier of a particular instance $\Pi_i^{l_i}$.
– $\mathbf{pid}_i^{l_i}$: the party with which instance $\Pi_i^{l_i}$ *believes* that he has been interacting.

Partnering. We say two protocol instances $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ are partnered if (1) $\mathbf{pid}_i^{l_i} = P_j$ and $\mathbf{pid}_j^{l_j} = P_i$; (2) $\mathbf{sid}_i^{l_i} = \mathbf{sid}_j^{l_j}$.

Adversarial Model. Roughly speaking, the adversary is allowed to fully control the external network. He can inject, modify, block and delete messages at will. He can also request any session keys adaptively. Formally, he can adaptively query the following oracles.

- $\mathbf{Execute}(i, l_i, j, l_j)$: When this oracle is called, it checks whether instances $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ are fresh. If either of them is old, it outputs $\perp$. Otherwise, a protocol execution between $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ takes place. At the end of the execution, a complete transcript (messages exchanged between the two instances) is returned. This oracle call models a threat from an eavesdropping adversary.
- $\mathbf{Send}(d, i, l_i, M)$: When this oracle is called, message M is sent to instance $\Pi_i^{l_i}$ as $Flow_d$. If instance $\Pi_i^{l_i}$ does not exist but $d \geq 2$, or if oracle $\mathbf{Send}(d, i, l_i, *)$ was called before, or if instance $\Pi_i^{l_i}$ already exists but either $\mathbf{Send}(d-2, i, l_i, *)$ was not previously called or its output was $\perp$ if called, then the oracle output is set to $\perp$; otherwise, the oracle output is whatever $\Pi_i^{l_i}$ returns. We stress that the oracle response needs to be consistent with $\mathbf{Send}(d-2t, i, l_i, *)$ for all $t > 0$. Furthermore, when $\mathbf{Send}(0, i, l_i, null)$ is called, it first checks whether instance $\Pi_i^{l_i}$ is fresh. If it is old, then the output is set to $\perp$; otherwise, $\Pi_i^{l_i}$ is initiated within P_i, and the output is whatever $\Pi_i^{l_i}$ returns as $Flow_1$. Similarly, when $\mathbf{Send}(1, i, l_i, M)$ is called, it first checks whether instance $\Pi_i^{l_i}$ is fresh. If it is old, then the output is set to $\perp$; otherwise, an instance $\Pi_i^{l_i}$ is initiated within party P_i as a responsor with input M. The output is whatever $\Pi_i^{l_i}$ outputs as $Flow_2$. The oracle call reflects a threat from man-in-the-middle attack.
- $\mathbf{Reveal}(i, l_i)$: When this oracle is called, it outputs the session key of instance $\Pi_i^{l_i}$ if it has accepted and completed with a session key derived; otherwise, it outputs $\perp$. This oracle reflects the threat from a session key loss.
- $\mathbf{Test}(i, l_i)$: This oracle does not reflect any real concern. However, it provides a security test. The adversary is allowed to query it once. The queried session must be completed and accepted. Furthermore, this session as well as its partnered session (if it exists) should not be issued a **Reveal** query. When this oracle is called, it flips a fair coin b. If $b = 1$, then the session key of $\Pi_i^{l_i}$ is provided to adversary; otherwise, a random number of the same length is provided. The adversary then tries to output a guess bit b'. He is successful if $b' = b$.

Having defined adversary behavior, we come to define the protocol security. It contains two conditions: correctness and privacy. The mutual authentication is considered in the privacy condition.

Correctness. If two partnered instances both accept, then they conclude with the same session key except for a negligible probability.

Privacy. We define two types of adversary success:

◇ If at any moment, an instance $\Pi_i^{l_i}$ with $\mathbf{pid}_i^{l_i} = P_j$ has accepted and completed with a session key derived while there does not exist an instance $\Pi_j^{l_j}$ with $\mathbf{pid}_j^{l_j} = P_i$ such that the exchanged messages seen by $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ prior to this moment (especially not including the currently generated message by $\Pi_i^{l_i}$ if any) are equal, then we announce the success of adversary. Furthermore, if such an instance $\Pi_j^{l_j}$ indeed exists, then we require it is unique except for a negligible probability.

◇ If the above event does not happen but the adversary succeeds in the test session, we also announce its success.

We use random variable **Succ** to denote the above success events. We define the advantage of adversary $\mathcal{A}$ as $\mathbf{Adv}(\mathcal{A}) := 2\Pr[\mathbf{Succ}] - 1$.

Now we are ready to provide a formal definition of security.

Definition 1. *A password authenticated key exchange protocol with mutual authentication is said to be secure if it satisfies*

- *Correctness.*
- *Privacy.*
  If adversary $\mathcal{A}$ makes Q_{send} queries to **Send** *oracle, then*

$$\mathbf{Adv}(\mathcal{A}) < \frac{Q_{send}}{|D|} + \mathbf{negl}(n), \tag{1}$$

  where D is the password dictionary, n is the security parameter.

Remarks. Here we give two comments on our definition and that in [2].

1. From our first privacy condition, whenever an instance $\Pi_i^{l_i}$ with $\mathbf{pid}_i^{l_i} = P_j$ accepts and completes, there exists an (essentially) unique instance in P_j (say, $\Pi_j^{l_j}$) with $\mathbf{pid}_j^{l_j} = P_i$ interacting with it and also the exchanged messages prior to the moment $\Pi_i^{l_i}$ accepts are not tampered. This is indeed our intuition about "$\Pi_j^{l_j}$ is authenticated".
2. In Bellare, *et al.* [2], MA is said to be violated if one instance terminates while no partner instance exists. This definition is not always satisfactory. Indeed, *session identifier* $\mathbf{sid}_i^{l_i}$ for instance $\Pi_i^{l_i}$ is popularly [13, 18] defined as a complete transcript seen by $\Pi_i^{l_i}$. Under this SID, their version of MA is always violated since once the adversary holds on the last message the partnership is never established. However, this problem does not occur for our version of MA since we only consider the messages exchanged before the considered instance (i.e., $\Pi_i^{l_i}$) accepts and completes. We stress that a provable MA property of a particular protocol in [2] does not contradict our remark here since their SID is defined as a *partial* transcript.
3. As pointed out in [11], our definition of MA could overkill some secure protocols. Consider any (secure) protocol Σ. We append 0 to each message in the Σ protocol but the protocol action remains unchanged (i.e., the redundant bit is ignored). It is easy to see that the revised protocol is no longer secure according to our definition of MA. Thus, a more concise MA should be defined to be a match of the minimal sub-transcript that uniquely determines the session key. However, we keep our definition. The reasons are as follows. First, our definition seems more natural and it is waived of a problem to determine the min-sub-transcript. Second, the overkilled protocol seems only due to the *redundant* bits which thus can be removed. Third, a protocol secure under our definition is also secure under the concise definition (since the first privacy condition (i.e, MA) in the latter is violated then it is violated in the former too).

3 Our Protocol

In this section, we introduce our 3-round construction under the common reference string (CRS) model, where all the parties have access to the public parameters that are drawn from a predetermined distribution. In reality, this condition could be realized by a trusted third party or a threshold scheme. Assume p, q are large primes with $q|(p-1)$; G_q is the (unique) multiplicative subgroup of F_p^* of order q; g, h are uniformly random generators of G_q; H is a collision resistant hash function; $e \leftarrow \mathcal{G}enPK(1^n)$ is the public key for a chosen ciphertext attack (in the postprocessing model) (CCA2) secure public key cryptosystem E (we stress that *nobody* knows the secret key of E_e); $\mathcal{F}$ is a pseudorandom function family and its realization with secret key σ is denoted by $F_\sigma()$. Our protocol is presented as Figure 1. Assume that password π_{ij} is ideally shared between party P_i and

$P_i(\pi_{ij})$		$P_j(\pi_{ij})$
	Pub: $g, h, H, \mathcal{F}, e, p, q$	
$x \leftarrow Z_q$ $A = g^x$ $C = h^x g^{\pi_{ij}}$	$\xrightarrow{A\|C\|P_i}$	
		$\lambda_1, \lambda_2 \leftarrow Z_q$ $\mu = g^{\lambda_1} h^{\lambda_2}$ $C' = Cg^{-\pi_{ij}}, \sigma = A^{\lambda_1} C'^{\lambda_2}$ $r = F_\sigma(3), \omega = E_e[\Sigma; r]$ $\Sigma = H(\mu\|A\|C'\|P_j\|P_i)$
	$\xleftarrow{\mu\|\omega\|P_j}$	
$C' = h^x, \sigma = \mu^x, r = F_\sigma(3)$ $\Sigma = H(\mu\|A\|C'\|P_j\|P_i)$ Verify: $\omega \stackrel{?}{=} E_e[\Sigma; r]$ If it holds, $\tau = F_\sigma(1), sk = F_\sigma(2)$	$\xrightarrow{\tau}$	
		Verify: $\tau \stackrel{?}{=} F_\sigma(1)$. If it holds, $sk = F_\sigma(2)$

Fig. 1. Key Exchange Protocol Execution between P_i and P_j

P_j. In order to establish a session key, P_i and P_j interact as follows. Assume P_i speaks first. He picks $x \leftarrow Z_q$ uniformly, computes a plain ElGalmal ciphertext $A|C$ and sends it together with id P_i to P_j as $Flow_1$. When P_j receives $Flow_1$, he chooses $\lambda_1, \lambda_2 \leftarrow Z_q$, and computes $\mu, C', \sigma, r, \omega, \Sigma$ properly, where r is used as the random input in encryption of Σ, and if it requires a longer string, r can be defined as $F_\sigma(3)|F_\sigma(4)|\cdots$ until it is long enough. We prefer the simple case since the security proof under this modification is essentially identical. Then he sends $\mu|\omega|P_j$ back to P_i (as $Flow_2$). Using μ, P_i is able to compute σ since $\sigma = \mu^x$. Then he verifies whether ω is a ciphertext of $H(\mu|A|C'|P_j|P_i)$ using random bits r. If the verification is successful, then he believes P_j is authentic and therefore returns an authentication tag $\tau = F_\sigma(2)$ as $Flow_3$. Furthermore, he outputs a session key $sk = F_\sigma(1)$ and terminates. When P_j receives τ, he checks whether τ is correct. If the verification succeeds, he believes P_i is authentic. Therefore, he accepts and outputs a session key $sk = F_\sigma(1)$. If the verification fails, it rejects. Note in the above interaction, *implementation issues (e.g., a validity check whether appropriate elements belong to G_q) are omitted for simplicity.*

3.1 Comparison with KOY Protocol

Now we provide a more detailed comparison with KOY protocol. As mentioned before, KOY protocol does not support MA, or it is 4-round if an additional flow is added. In contrast, our protocol is 3-round with MA. Each party in their construction needs 15 exps while ours needs at most 4 exps plus one ciphertext of a CCA2-secure PKE(note it is easy to find such a PKE with a ciphertext cost less than 11 exps). Their construction employs a one-time signature to "bind" the whole transcript while we do not use such a technique since it requires the responsor to store the whole transcript, which might be more vulnerable to denial of service (DoS) attack. However, we stress their construction is instructive to us. Specifically, in authentication of initiator, we use a technique that if $A|C$ is not an ElGamal ciphertext of $g^{\pi_{ij}}$, then σ is uniformly random in G_q. This technique is essentially from KOY protocol with a relaxation of Cramer-Shoup ciphertext [10] to ElGamal ciphertext.

4 Security

In this section, we prove the security of our protocol.

Theorem 1. *Let Γ be the password authenticated key exchange protocol in Figure 1. Let a, b, c be polynomially related to the security parameter n. Assume $e \leftarrow \mathcal{G}enPK(1^n)$ is the public key of a CCA2 secure public key cryptosystem E; $H : \{0,1\}^* \rightarrow \{0,1\}^a$ is a collision resistant hash function uniformly taken from a family $\mathcal{H}$; p, q are large primes with $q|(p-1)$; $\mathcal{F}$ is a pseudorandom function family from $\{0,1\}^b$ to $\{0,1\}^c$; G_q is the (unique) multiplicative subgroup of order q in F_p^*; g, h are random generators of G_q. Then under* DDH *assumption, protocol Γ is secure.*

Proof. Define $\mathbf{sid}_i^{l_i}$ to be the whole transcript seen by instance $\Pi_i^{l_i}$. Assume $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ are partnered and both accept. Then, $\mathbf{pid}_i^{l_i}$ and $\mathbf{pid}_j^{l_j}$ are consistent and the messages are faithfully exchanged. Thus, P_i and P_j derive the same σ: $\sigma = \mu^x = A^{\lambda_1} C'^{\lambda_2}$. Thus, the correctness follows.

In the rest, we concentrate on the proof of the privacy condition. We look the protocol execution as a game between a simulator and an adversary $\mathcal{A}$. The simulator picks large prime p, q with $q|(p-1)$ and takes $g \leftarrow G_q, u \leftarrow Z_q$, $(e, d) \leftarrow \mathcal{G}en(1^n)(= (\mathcal{G}enPK, \mathcal{G}enSK)(1^n))$, $\mathcal{F}$ a pseudorandom function family from $\{0,1\}^b$ to $\{0,1\}^c$ and H uniformly from a family of a collision resistant hash function (CRHF). He lets $h = g^u$. Then he sets the public parameters as $g, h, H, \mathcal{F}, e, p, q$ and assigns passwords to parties as in the real protocol. He simulates the protocol execution with adversary $\mathcal{A}$.

We construct a sequence of slightly modified protocols $\Gamma_1, \Gamma_2, \cdots$ from Γ and show that the success probability of $\mathcal{A}$ in Γ_i is no less than that in Γ_{i-1} except for a negligible gap for any $i \geq 1$, where $\Gamma_0 := \Gamma$. And then we bound the success probability of $\mathcal{A}$ in the last variant. Before our actual proof, *we assume that in response to any oracle query, the basic validity check in its definition has already been successfully verified thus the output is never* $\perp$.

For given two parties P_i and P_j with common password π_{ij}, we say $A|C$ is *inconsistent* if $\log_g A \neq \log_h Cg^{-\pi_{ij}}$. We first introduce the following simple fact, where the proof is mainly due to the fact that λ_1, λ_2 are both uniform in Z_q (independent of anything else).

Fact 1 *If $A|C$ is inconsistent, then σ is uniformly random in G_q, given $A|C|\mu$ where σ and μ are derived according to the responsor's execution.*

Game Γ_1. Now we modify Γ_0 to Γ_1 with the only difference in **Execute** query, where C in Γ_1 is chosen uniformly random. Using a hybrid argument or a better proof similar to Lemma 2 in [18], both with reduction to DDH assumption, we have

Lemma 1. *Under* DDH *assumption in G_q, the success probabilities of $\mathcal{A}$ in Γ and Γ_1 are negligibly close.*

Game Γ_2. We modify Γ_1 to Γ_2 with only difference in **Execute** queries where r, τ and $sk_i^{l_i}(= sk_j^{l_j})$ in any **Execute**(i, l_i, j, l_j) are chosen uniformly random from $\{0,1\}^{3c}$. Note $A|C$ is inconsistent in **Execute** queries of Γ_1 (and Γ_2) except for a negligible probability. By **Fact 1**, one can conclude the following lemma using a standard hybrid argument with reduction to the pseudorandomness of $\mathcal{F}$.

Lemma 2. *The success probabilities of $\mathcal{A}$ in Γ_1 and Γ_2 are negligibly close.*

Game Γ_3. Now we modify Γ_2 to Γ_3 with the only difference in computing ω in **Execute** query, where Simulator picks $C^* \leftarrow G_q$ randomly and defines $\omega = E_e(H(\mu|A|C^*|P_j|P_i); r)$ instead of a ciphertext of $\Sigma = H(\mu|A|C'|P_j|P_i)$. Here r is uniformly random (as in Γ_2). By a standard hybrid argument with reduction to the semantic security[3] of cryptosystem E (note the challenge template should be set according to the above modification), we have the following lemma.

Lemma 3. *The success probabilities of $\mathcal{A}$ in Γ_2 and Γ_3 are negligibly close.*

Game Γ_4. Till now, we have finished modifying **Execute** oracle. Next, let us consider **Send** oracle. Before that, we introduce some notations. We say that a message is *adversary-generated* if it is not exactly equal to the output of a **Send** oracle or a *Flow* in a response of an **Execute** oracle; otherwise, we say it is an *oracle-generated* message. Consider any query **Send**$(2, i, l_i, \mu|\omega|P_j)$. If there *exists* **Send**$(1, j, l_j, A|C|P_i)$ such that it outputs $\mu|\omega|P_j$ and that $A|C|P_i$ is exactly the output of **Send**$(0, i, l_i, null)$, then we say that **Send**$(2, i, l_i, \mu|\omega|P_j)$ *matches* with **Send**$(1, j, l_j, A|C|P_i)$; otherwise, we say that a *none-match* event happens to **Send**$(2, i, l_i, \mu|\omega|P_j)$. Now we modify Γ_3 to Γ_4 with the only difference: upon any query **Send**$(2, i, l_i, \mu|\omega|P_j)$, if a *none-match event* happens to it (note Simulator can check this since it controls all the oracles), then deciding accept/reject only depends on whether ω can be decrypted to $\Sigma = H(\mu|A|C'|P_j|P_i)$ or not, where $A|C|$ is in the output of **Send**$(0, i, l_i, null)$ and $C' = Cg^{-\pi_{ij}}$. If it accepts in this case, it announces the success of $\mathcal{A}$ and halts. Note in case of a *match event* it responses as in Γ_3.

[3] Here semantic security suffices and CCA2 security will be required later to deal with **Send** oracle.

Lemma 4. *The success probability of $\mathcal{A}$ in $\varGamma_4$ is no less than that in $\varGamma_3$.*

Proof. Note in case of a none-match event, if $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$ in $\varGamma_4$ rejects, then it rejects in $\varGamma_3$ too. Therefore, before a none-match event is accepted in $\varGamma_4$, adversary view in $\varGamma_4$ is identically distributed as that in $\varGamma_3$. On the other hand, an accepted none-match event in $\varGamma_4$ already announces the success of $\mathcal{A}$. Thus, the conclusion follows. $\square$

Game $\varGamma_5$. Now we modify $\varGamma_4$ to $\varGamma_5$ such that C in any $\mathbf{send}(0, i, l_i, null)$ is taken uniformly random from G_q. In order of consistency (in view of $\mathcal{A}$), we need to take care of other oracle definitions. $\mathbf{Send}(1, j, l_j, M)$ remains unchanged. Since there does not exist x in $A|C$ such that the normal action can be executed, $\mathbf{Send}(2, i, l_i, A|C|P_j)$ is modified as follows.

i) If there exists a unique l_j such that $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$ matches with $\mathbf{Send}(1, j, l_j, M)$, then it *accepts* (without verification of ω) and computes $\tau = F_\sigma(2)$ using σ defined in $\mathbf{Send}(1, j, l_j, M)$. Then, he outputs τ and defines the session key $sk_i^{l_i} = F_\sigma(1)$. If there are two or more l_j, $l'_j, \cdots$ such that the above match event holds simultaneously (in the future, we call it a *multi-match* event), then it chooses one match randomly and follows the same procedure.
ii) If a none-match event happens to $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$, then it responses as in $\varGamma_4$ (i.e. it decrypts ω, and decides to announce the success of $\mathcal{A}$ or to reject).

The $\mathbf{Send}(3, j, l_j, M)$ answers normally. The rest oracles remain unchanged (note the validity follows from the fact that their actions do not depend on the above modification).

Lemma 5. *The success probabilities of $\mathcal{A}$ in $\varGamma_4$ and $\varGamma_5$ are negligibly close.*

Proof. To relate $\varGamma_4$ and $\varGamma_5$, we define a slightly modified $\varGamma_4$ as $\varGamma'_4$. The only difference is that in case of a *match event* in $\varGamma'_4$, $\mathbf{Send}(2, i, l_i, \mu|\omega|P_i)$ responses as i) in definition of $\varGamma_5$. On the one hand, if l_j is always unique (whenever a match event happens), then adversary views in $\varGamma_4$ and $\varGamma'_4$ are identically distributed since a unique match event is always accepted in $\varGamma_4$. On the other hand, the probability that a multi-match event happens throughout the simulation is negligible since μ is uniform in G_q. Thus, the success probabilities of $\mathcal{A}$ in $\varGamma_4$ and $\varGamma'_4$ are negligibly close. Notice that executions of Games $\varGamma'_4$ and $\varGamma_5$ are different only in that C is real or random. Thus, if the conclusion were wrong, a standard hybrid argument directly would reduce to break DDH assumption, a contradiction. Details are omitted. $\square$

Game $\varGamma_6$. Now we modify $\varGamma_5$ to $\varGamma_6$ with the only difference in oracle $\mathbf{Send}(1, j, l_j, A|C|P_i)$. If $A|C$ is consistent: $C = A^u g^{\pi_{ij}}$, it announces the success of adversary $\mathcal{A}$ and exits (recall Simulator knows $u := \log_g h$; recall normally $C \neq A^u g^{\pi_{ij}}$ since C is chosen uniformly random in oracle $\mathbf{Send}(0, *, *, null)$); otherwise, it answers normally (as in $\varGamma_5$). The rest oracle definitions remain unchanged as in $\varGamma_5$. Note this modification only increases the success probability of $\mathcal{A}$. Indeed, if $A|C$ is always inconsistent, then the adversary view in $\varGamma_6$ is identically distributed as in $\varGamma_5$; otherwise, $\mathcal{A}$ already succeeds in $\varGamma_6$. Thus, we have

Lemma 6. *The success probability of $\mathcal{A}$ in $\varGamma_6$ is no less than that in $\varGamma_5$.*

Game $\varGamma_7$. $\varGamma_7$ is modified from $\varGamma_6$ as follows. In order to answer oracle $\mathbf{Send}(1, j, l_j, A|C|P_j)$ in $\varGamma_7$, Simulator chooses σ uniformly random from G_q instead of $A^{\lambda_1} C'^{\lambda_2}$. Other oracle definitions remain unchanged as in $\varGamma_6$ (here the validity is due to the fact that the state information λ_1, λ_2 is not required in these oracle definitions).

Lemma 7. *The success probabilities of $\mathcal{A}$ in Γ_6 and Γ_7 are equal.*

Proof. Whenever σ is defined in Γ_6 (and Γ_7), this implies that $\mathcal{A}$ is not announced to succeed in $\mathbf{Send}(1, j, l_j, A|C|P_i)$ and thus $A|C$ is inconsistent. Thus, from **Fact 1**, the adversary view in Γ_6 and Γ_7 is identically distributed. The conclusion follows immediately. □

Game Γ_8. Now we modify Γ_7 to Γ_8 with the only difference: $(r, \tau, sk_i^{l_i})$ in **Send** oracles are chosen uniformly random from $\{0, 1\}^{3c}$, which is the range of $\mathcal{F}$. Details are as follows. Whenever any $\mathbf{Send}(1, j, l_j, A|C|P_i)$ is called, Simulator follows the oracle definition in Γ_7 except r is random in $\{0, 1\}^c$. When any $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$ oracle is called, Simulator responses as in $\Gamma_5 - \Gamma_7$ with the following exception: in case of a match event, $\tau, sk_i^{l_i}$ are taken uniformly random in $\{0, 1\}^c$ and furthermore he saves tuple $(\mu, \tau, sk_i^{l_i}, i, j)$ in his memory. Whenever any $\mathbf{Send}(3, j, l_j, \tau')$ is called, Simulator searches for $(\mu, *, *, *, j)$ in his memory. If a unique tuple is found, then it recovers $(\tau, sk_i^{l_i}, i)$ from this tuple and checks whether $\tau' = \tau$. If it holds, $\mathbf{Send}(3, j, l_j, \tau')$ accepts and concludes the session key $sk_j^{l_j} := sk_i^{l_i}$. If more than one such a tuple are found, then it chooses one randomly and follows the same procedure. Otherwise, if either of the above two checks (i.e., search and comparison) fails, it rejects. The rest oracle definitions (**Reveal, Test, Execute**) remain unchanged (the validity follows since such definitions are independent of the way **Send** chooses $(r, \tau, sk_i^{l_i})$).

Lemma 8. *The success probabilities of $\mathcal{A}$ in Γ_7 and Γ_8 are negligibly close.*

Proof. Consider a slightly modified Γ_7, denoted as Γ_7'. Oracle definitions in Game Γ_7' are identical to those in Γ_8 except that $(r, \tau, sk_i^{l_i}(= sk_j^{l_j}))$ is computed as $F_\sigma(3), F_\sigma(2), F_\sigma(1)$. We show that the success probabilities of $\mathcal{A}$ in Γ_7 and Γ_7' are negligibly close. Indeed, for $\mathbf{Send}(1, *, *, *)$ and $\mathbf{Send}(2, *, *, *)$, adversary views in Γ_7' and Γ_7 are identical since their *outgoing* messages are computed from the same definitions. When $\mathbf{Send}(3, j, l_j, \tau')$ in Γ_7' is called, there are two cases.
Case 1: A tuple $(\mu, *, *, *, j)$ is found: Suppose such a tuple is recorded by $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$, based on the match with $\mathbf{Send}(1, j, l_j', M)$. Then since μ is uniform in G_q, it follows $l_j = l_j'$ except for a negligible probability. Thus, the decision based on $\tau' = \tau(= F_\sigma(2))$ is well consistent with that in Γ_7 except for a negligible probability.
Case 2: The tuple $(\mu, *, *, *, j)$ is not found: We show that the probability of wrong decision (i.e., $\tau' = F_\sigma(2)$) is negligible. If this is incorrect, we build a distinguisher $\mathcal{D}_7$ for $\mathcal{F}$. Let η_7 be the upper bound of the number of $\mathbf{Send}(1, *, *, *)$ queries. $\mathcal{D}_7$ simulates Γ_7' as done by Simulator except for lth query $\mathbf{Send}(1, j, , l_j, M)$, where r is provided by his function oracle $\mathcal{O}$ with input 3. If at some moment, $\mathcal{A}$ makes a query $\mathbf{Send}(2, *, *, \mu|\omega|P_j)$ that matches with $\mathbf{Send}(1, j, l_j, M)$, then $\mathcal{D}_7$ terminates the simulation and outputs 0, 1 randomly. When $\mathbf{Send}(3, j, l_j, \tau')$ is called, $\mathcal{D}_7$ first looks up $(\mu, *, *, *, j)$ in his memory. If it is found, then it terminates the simulation and outputs 0, 1 equally likely; otherwise, it feeds 2 to $\mathcal{O}$ and gets back τ. In this case, if $\tau' = \tau$, then he outputs 1; otherwise, he outputs 0. If $\mathcal{O} = \mathcal{F}$, adversary view in the simulation by $\mathcal{D}_7$ is identically distributed as that in Γ_7' since σ in Γ_7' (and Γ_7) is uniformly random from G_q (thus the function oracle outputs are perfectly consistent with Γ_7'). An easy calculation shows that the probability $\mathcal{A}$ outputs 1 is $\frac{p_0}{2\eta_7} + \frac{1}{2}$, where p_0 is the probability of the wrong rejection event. If $\mathcal{O}$ is purely random function family, then τ is uniform in $\{0, 1\}^c$ (independent of anything else). Thus, $\mathbf{Send}(3, j, l_j, \tau')$ wrongly accepts with probability at most 2^{-c} (sufficient to consider $\tau' = \tau$). Thus, $\mathcal{D}_7$ has a non-negligible advantage, contradiction.

Other oracle definitions in Γ_7' and Γ_7 are identical. Thus, the success probabilities of $\mathcal{A}$ in Γ_7 and Γ_7' are negligibly close. Furthermore, the success probabilities of $\mathcal{A}$ in Γ_7' and Γ_8 are negligibly close, because their executions are identical only except that $(r, \tau, sk_i^{l_i})$ in Γ_8 are taken uniformly random and thus a standard hybrid argument with reduction to the pseudorandomness of $\mathcal{F}$ can be applied. □

Game Γ_9. Γ_8 is modified to Γ_9 so that ω in $\mathbf{Send}(1, j, l_j, A|C|P_i)$ is defined as $E_e[\Sigma'; r]$, where r is uniform in $\{0,1\}^c$ and $\Sigma' = H(\mu|A|C^*|P_j|P_i)$ for $C^* \leftarrow G_q$. The rest oracles are unchanged. We have the following result.

Lemma 9. *The success probabilities of $\mathcal{A}$ in Γ_8 and Γ_9 are negligibly close.*

Proof. We define $\Gamma_8^{(l)}$ to be the variant of Γ_8 such that the first l $\mathbf{Send}(1, *, *, *)$ queries are answered according to Γ_9 and the rest queries are answered according to Γ_8. It follows $\Gamma_8^{(0)} = \Gamma_8$ and $\Gamma_8^{(\eta_9)} = \Gamma_9$, where η_9 is the upperbound of number of queries $\mathbf{Send}(1, *, *, *)$. If the success gap in Γ_8 and Γ_9 were non-negligible, then there would exist $z \in \{1, \cdots, \eta_9\}$ such that the success gap between $\Gamma_8^{(z-1)}$ and $\Gamma_8^{(z)}$ would be non-negligible. We build a CCA2 breaker $\mathcal{D}_9$ for E_e as follows. He takes l randomly from $\{1, \cdots, \eta_9\}$ and initializes public parameters as done by Simulator except e provided by his challenger. Then, $\mathcal{D}_9$ simulates $\Gamma_8^{(l)}$ except for the lth $\mathbf{Send}(1, *, *, *)$ query, say $\mathbf{Send}(1, j, l_j, A|C|P_i)$. In this case, he computes Σ and gives $(\Sigma, \mu|A|P_j|G_q)$ to his encryption oracle, requesting that a random message has a pattern $\Sigma' = H(\mu|A|C^*|P_j|P_i)$ for $C^* \leftarrow G_q$. Then, he will receive ω^*, that is an encryption of either Σ or a random message Σ' of that pattern. $\mathbf{Send}(1, j, l_j, A|C|P_i)$ outputs $\mu|\omega^*|P_j$. Different from Simulator, $\mathcal{D}_9$ does not have a private key d for E. Thus, we need to guarantee his action is consistent. Upon any $\mathbf{Send}(2, s, l_s, \mu'|\omega'|P_t)$,

1. If a match event happens to this oracle, it responses as in Γ_8, Γ_9.
2. If $\omega^* \neq \omega'$ and a none-match event happens to $\mathbf{Send}(2, s, l_s, \mu'|\omega'|P_t)$, then $\mathcal{D}_9$ asks his oracle to decrypt ω' in order to make acceptance/reject decision.
3. If $\omega^* = \omega'$ and a none-match event happens to $\mathbf{Send}(2, s, l_s, \mu'|\omega'|P_t)$, then $\mathcal{D}_9$ simply rejects. We show that the wrong decision probability is negligible only; otherwise, H is not collision resistant. Suppose that $\tilde{A}|\tilde{C}|P_s$ is the output by $\mathbf{Send}(0, s, l_s, null)$ and that ω^* decrypts to $H(\mu|A|C^\dagger|P_j|P_i)$ where $C^\dagger$ is chosen by its challenger and thus is $Cg^{-\pi_{ij}}$ or a random C^* in G_q. If the decision is wrong, we have $H(\mu|A|C^\dagger|P_j|P_i) = H(\mu'|\tilde{A}|\tilde{C}g^{-\pi_{st}}|P_t|P_s)$. By the collision resistant property of H, $\mu|A|C^\dagger|P_j|P_i \neq \mu'|\tilde{A}|\tilde{C}g^{-\pi_{st}}|P_t|P_i$ only with negligible probability (Otherwise, we can build a breaker of H that simulate the game by playing the roles of both $\mathcal{D}_9$ and the challenger of $\mathcal{D}_9$ and waiting for the collision to happen). Thus, $\mu = \mu', A = \tilde{A}, C^\dagger = \tilde{C}g^{-\pi_{st}}, t = j, s = i$, except for a negligible probability. If $C^\dagger = Cg^{\pi_{ij}}$, then $C = \tilde{C}$ thus $\mathbf{Send}(2, s, l_s, \mu'|\omega'|P_t)$ in fact matches with $\mathbf{Send}(1, j, l_j, A|C|P_i)$, contradiction to the none-match assumption for the former. If $C^\dagger = C^*$ a random in G_q, then $\tilde{C}g^{-\pi_{st}} = C^\dagger$ holds only with negligible probability. Thus, as a summary, the decision is wrong only with a negligible probability.

The rest oracles are answered normally as in Γ_8 (or Γ_9) since no decryption is required any more. Thus, in case ω^* is a ciphertext of Σ, then adversary view in the simulation is negligibly close to that in $\Gamma_8^{(l-1)}$; otherwise it is negligibly close to $\Gamma_8^{(l)}$. Thus, a correct guess for z, which is non-negligible, immediately implies non-negligible advantage of $\mathcal{D}_9$, a contradiction. □

Bounding Success Probability in Γ_9. Now let us consider protocol Γ_9. The adversary succeeds only possibly (1) at $\mathbf{Send}(1, j, l_j, A|C|P_i)$ where he inputs a consistent ElGamal ciphertext $A|C$, or (2) at oracle $\mathbf{Send}(2, i, l_i, \mu|\omega|P_j)$ where a none-match event occurs, but the oracle decrypts ω to $\Sigma = H(\mu|A|C'|P_j|P_i)$, or (3) at $\mathbf{Send}(3, j, l_j, \tau)$, where the oracle accepts but τ is not the output by a $\mathbf{Send}(2, i, l_i, *)$ that is matched to $\mathbf{Send}(1, j, l_j, *)$, or (4) at **Test** query. Here we stress that mutual authentication in Definition 1 is fully covered by (2) and (3). For case (3), since τ will be compared with the value in the memory, success here happens only when there are two $\mathbf{Send}(2, *, *, *)$ that match with $\mathbf{Send}(1, j, l_j, *)$. This implies that $\exists$ two $\mathbf{Send}(0, *, *, null)$ generate the same output. This happens with only negligible probability since A is uniform in G_q. We thus only consider cases (1), (2), and (4). We say the adversary attempt to succeed in cases (1) (2) is an impersonation trial, denoted by **ITri**. In case (1), no input can be successful in two protocol executions with different password candidates (recall that $D = \{1, \cdots, N\}$ with $N < q$). In case (2), no input can be accepted with non-negligible probability in two password candidates (otherwise, we can break H in two steps: Step 1. Simulate the protocol execution and record all the events in case (2); Step 2. Check whether the collision in case (2) happens by trying to find two passwords that accept some recorded event simultaneously)[4]. Thus, we assume each input at case (1) or (2) can be accepted by at most one password candidate. Notice that just before **ITri** happens, the adversary view in Γ_9 is completely independent of password. Thus, immediately after the first **ITri** is rejected, the adversary view is distributed identically among a password dictionary of size at least $|D| - 1$. The reason is: it has the same reject event for at least $|D| - 1$ password candidates. Furthermore, using a simple induction, we have the probability that the first l **ITri** events are rejected but it succeeds in $l + 1$th **ITri** event is $\frac{1}{|D|-l}\prod_{i=1}^{l}(1 - \frac{1}{|D|-(i-1)}) = \frac{1}{|D|}$. Thus, suppose the number of **Send** queries is upperbounded by Q_{send}. Then the success in **ITri** happens with probability at most $\frac{Q_{send}}{|D|}$ except for a negligible gap. Now we consider case (4), this success event happens only if the success event in **ITri** does not happen. In this case, since the session key is chosen uniformly random independent of anything else. Thus, the success probability is exactly $\frac{1}{2}$ except that the session key was seen at a previous moment, which is only possible by **Reveal** query. Note the test session is not allowed to issue **Reveal** query. We show the revealed session must be its partnered session, which is not allowed by definition. To this end, let $\Pi_i^{l_i}$ be the test session with $\mathbf{pid}_i^{l_i} = P_j$. Since $\mathbf{Send}(2, i, l_i, *)$ accepts with $sk_i^{l_i}$ derived, there must exist a matched $\mathbf{Send}(1, j, l_j, *)$ and a tuple $(\mu, \tau, sk_i^{l_i}, i, j)$ is stored in the memory. And later only $\mathbf{Send}(3, j, l'_j, \tau')$ with μ in the output of $\mathbf{Send}(1, j, l'_j, M)$ will access this tuple and define $sk_j^{l'_j} = sk_i^{l_i}$. Note in this case, $l_j = l'_j$ except for a negligible probability since μ is uniform in G_q. The exchanged messages seen by $\Pi_i^{l_i}$ and $\Pi_j^{l_j}$ (unique except for negligible probability) are identical by definition of *match*, and they see the same τ (as in the tuple). Thus, $\mathbf{pid}_i^{l_i} = P_j$, $\mathbf{pid}_j^{l_j} = P_i$ and $\mathbf{sid}_i^{l_i} = \mathbf{sid}_j^{l_j}$. That is, they are partnered sessions.

[4] Here in order for our attack to be polynomial time, we use the fact that $|D|$ is polynomially bounded. If $|D|$ is super polynomial, although it is not the setting for password KE protocol, the conclusion should be revised as: no input from $\mathcal{A}$ can be accepted by non-negligible fraction of password candidates; otherwise, an adversary $\mathcal{O}$ for H can be built as follows. He simulates Γ_9 and runs $\mathcal{A}$ against it. And he records the events in case (2). And for each event, he randomly picks a password $\pi \in D$ and tests whether this event can also be accepted by π. If yes, he obtains a collision of H. An easy calculation shows that if the conclusion is wrong, the success probability of $\mathcal{O}$ is non-negligible. Details are omitted. Due to the above modification, the subsequent proof content should be adjusted accordingly by allowing a negligible gap.

As a summary, the success probability of adversary in **Test** session is exactly $\frac{1}{2}$. Let α be the probability of **ITri** event. Then the total success probability of adversary is $\alpha+(1-\alpha)\frac{1}{2} \leq \frac{1}{2}+\frac{Q_{send}}{2|D|}$.

Proof of Theorem 1 Summarizing the results in Lemmas 1- 9 and success probability of $\mathcal{A}$ in Γ_9, we have $\mathbf{Adv}(\mathcal{A}) < \frac{Q_{send}}{|D|} + \mathbf{negl}(n)$. ♠

Acknowledgement The authors would like to thank anonymous referees for valuable comments. S. Jiang would like to thank Mihir Bellare for kind response upon query on mutual authentication, and he especially feels grateful to David Pointcheval for an instructive discussion on the definition of mutual authentication.

References

1. Mihir Bellare, Ran Canetti, and Hugo Krawczyk, A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols, *STOC 98*: 419-428.
2. Mihir Bellare, David Pointcheval, Phillip Rogaway: Authenticated Key Exchange Secure against Dictionary Attacks. *EUROCRYPT 2000*: 139-155.
3. Mihir Bellare, Phillip Rogaway: Entity Authentication and Key Distribution. *CRYPTO 1993*: 232-249.
4. Bellovin, S.M.; Merritt, M., Encrypted key exchange: password-based protocols secure against dictionary attacks, In *Proceedings of the 1992 IEEE Computer Society Symposium on Research in Security and Privacy,* 72-84.
5. Steven M. Bellovin, Michael Merritt: Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise. *ACM Conference on Computer and Communications Security 1993*: 244-250.
6. Simon Blake-Wilson, Don Johnson, Alfred Menezes: Key Agreement Protocols and Their Security Analysis. *IMA Int. Conf.* 1997: 30-45.
7. Victor Boyko, Philip D. MacKenzie, Sarvar Patel: Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. *EUROCRYPT 2000*: 156-171.
8. Ran Canetti and Hugo Krawczyk, Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels, *Eurocrypt 2001*: 453-474.
9. Ran Canetti, Oded Goldreich, Shai Halevi: The Random Oracle Methodology, Revisited (Preliminary Version). *STOC 1998*: 209-218.
10. Ronald Cramer, Victor Shoup: A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. *CRYPTO 1998*: 13-25.
11. David Pointcheval, private communication, April 2004.
12. W. Diffie, P.C. van Oorschot, and M.J. Wiener, Authentication and Authenticated Key Exchanges, *Designs, Codes and Cryptography,* vol. 2, no. 2, 1992, pp. 107-125.
13. Rosario Gennaro, Yehuda Lindell: A Framework for Password-Based Authenticated Key Exchange. *EUROCRYPT 2003*: 524-543.
14. Oded Goldreich, Yehuda Lindell: Session-Key Generation Using Human Passwords Only. *CRYPTO 2001*: 408-432.
15. Shai Halevi, Hugo Krawczyk: Public-Key Cryptography and Password Protocols. *ACM Conference on Computer and Communications Security 1998*: 122-131.
16. David P. Jablon, Extended Password Key Exchange Protocols Immune to Dictionary Attacks. *WETICE 1997*: 248-255.
17. Shaoquan Jiang and Guang Gong, Password based Key Exchange with Mutual Authentication, *SAC 2004*. The current paper is the full version.
18. Jonathan Katz, Rafail Ostrovsky, Moti Yung: Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. *EUROCRYPT 2001*: 475-494.
19. Stefan Lucks, Open Key Exchange: How to Defeat Dictionary Attacks Without Encrypting Public Keys. *Security Protocols Workshop* 1997: 79-90. Available at http://th.informatik.uni-mannheim.de/People/Lucks/papers.html
20. Alfred Menezes, Paul C. van Oorschot, Scott A. Vanstone: *Handbook of Applied Cryptography.* CRC Press 1996.