Constructing Efficient PAKE Protocols from Identity-Based KEM/DEM

Kyu Young Choi[1], Jihoon Cho[1], Jung Yeon Hwang[2], and Taekyoung Kwon[3]

[1] Samsung SDS, Inc., Seoul, Korea
{ky12.choi,jihoon1.cho}@samsung.com
[2] ETRI, Daejeon, Korea
videmot@etri.re.kr
[3] Yonsei University, Seoul, Korea
taekyoung@yonsei.ac.kr

Abstract. In this paper, we propose an efficient identity-based password authenticated key exchange (IBPAKE) protocol using identity-based KEM/DEM. In IBPAKE, a client conducts authentication based on a human-memorable password and a server's identity. A distinctive feature of IBPAKE protocols, compared to the well-known EKE-type PAKE protocols, is that an adversary who even acquired a user's password cannot impersonate a server to further investigate user's sensitive information.

We first construct the new IBPAKE protocol using the Boneh-Franklin Identity-based encryption (IBE) scheme, and then generalize the protocol by presenting a generic method to yield an efficient IBPAKE protocol from identity-based KEM/DEM. Our fine-grained approach has concrete advantages in terms of performance. First, unnecessary parameters can be removed easily. This allows a straightforward improvement on computational cost and communication bandwidth. In addition, using the essential feature of identity-based KEM/DEM, we can construct an IBPAKE protocol which runs in a single pass. Our protocol gives better performance, compared to prior known IBPAKE protocols.

1 Introduction

A key exchange (KE) protocol is run by parties who want to communicate privately over a public network. After completing the protocol, communicating parties can obtain cryptographic keys to be used for cryptographic tasks later. In order to share correct keys only between intended participants, it is required to achieve two security notions, the secrecy of a key and authentication of participants. The secrecy of a key is used to defeat passive adversaries who eavesdrop communication messages. Two-party Diffie-Hellman (DH) KE protocol [13] is a well-known single round KE protocol to provide it. However, in an ordinary public network, there exist active adversaries who can control traffic messages in an adaptive way. The adversaries may impersonate protocol participants by deleting, inserting or modifying traffic messages at will. To be secure against

active attacks, a KE protocol should achieve appropriate authentication, that is, some assurance to know each other's true identities.

Among various factors of authentication, a password is commonly used because it can be memorized by human without any specific device. In practice, most of IT services use ID/password as a log-in method. A lot of password-authenticated KE (PAKE) protocols have been suggested extending the Diffie-Hellman protocol in a client-server model. In principle, due to low entropy of a password, PAKE can be easily vulnerable to *dictionary* attacks in which an adversary tries a word from a dictionary exhaustively. On-line dictionary attacks can be simply prevented by applying a limit on the number of an on-line authentication trial. One of the main challenges for PAKE is to design a protocol which is immune to off-line dictionary attacks. In a PAKE protocol, a client and a server transmit transcripts containing a password. An off-line dictionary attack can be mounted when some information to verify a true password, called *password verifier*, is revealed from the transcripts.

For a PAKE protocol, so-called 'Encrypted Key Exchange (EKE)' is presented in [5]. Also, various PAKE protocols extending EKE have been constructed. The essential idea of EKE is to encrypt ephemeral DH keys by a password. Since ephemeral DH keys are generated from a random distribution, decryption by a guessed password does not reveal meaningful information about the original one. Off-line dictionary attacks can be prevented effectively in EKE.

However there are various situations where a password can be leaked; for example, by a malware, hacking, shoulder surfing attacks and information leakage from lost/stolen portable devices. When a password of a client is revealed, it is inevitable that an adversary can impersonate the client. It will be also quite dangerous if an adversary is able to impersonate a server to the client, using a password stolen from a client. A server may provide a client with useful services and important information, e.g., financial service, healthcare information, etc. In EKE, a client and a server authenticate each other by a shared password (or a password verifier). Therefore it is vulnerable to server impersonation attacks when a password is revealed.

As an approach to resolve the problem, we can apply a hybrid method combining password authentication and asymmetric cryptographic schemes. For example, [15] and [17] use a public key encryption scheme in conjunction with password authentication. It uses a password for a client and a public key encryption scheme for a server. In a usual client-server model, a client is a human who can memorize a password and a server is a powerful machine which can store a high-entropy secret key. The hybrid structure fits for such *unbalanced computational* environments. A public key is set to be a random string for security. A client must check if the random public key is corresponding to a specific server by a certificate [18]. The maintenance of certificates entails additional computation and communication costs.[4] To simplify the certificate-based public key management, one can build an identity-based cryptosystem [23, 4]. Here, a

[4] For example, each client must verify a server's certificate (e.g., X.509 Certificates) via CRL (Certificate Revocation Lists) or OCSP (Online Certificate Status Protocol).

public key can be replaced with an arbitrary public string that a user chooses, such as an e-mail address or IP address.

Recently, an identity-based password authenticated key exchange (IBPAKE) protocol is introduced in a client-server model [26]. In the protocol, a server's public identity is additionally used to encrypt a password. Since the server's public identity such as a company or brand name is typically well remembered, a client can perform a convenient authentication. However, the IBPAKE protocol of [26] is generically constructed from an identity-based encryption (IBE) scheme. Though it gives a conceptually simple design principle, efficiency is further studied.

Our Results. In this paper, we propose an efficient IBPAKE protocol using identity-based key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM) [2]. Basically, identity-based KEM/DEM works in identity-based cryptosystem, that is, a public key is defined as an arbitrary string. Thus a client can do an easy authentication based on a human-memorable password and server's identity.

In contrast to the approach of [26], our approach has various advantages in terms of the performance. Intuitively, we can control a keying material from identity-based KEM/DEM more precisely and remove unnecessary parameters. This gives straightforward improvement on computational cost and communication bandwidth. By a fine-grained design based on identity-based KEM/DEM, we can construct an IBPAKE protocol which runs in a single pass. With respects to the performance of a PAKE protocol, reducing the number of communication passes(or rounds) is an important issue because it fundamentally affects communication latency. Note that this is impossible in [26] because of structural limitation of the approach.

Our first IBPAKE protocol is constructed by using the well-known Boneh-Franklin IBE scheme [4]. Next, generalizing our first protocol, we presents a generic method to yield an efficient IBPAKE protocol using identity-based KEM/DEM. Our idea is to combine an identity-based KEM/DEM and a DH KE protocol. We formally prove its security. Using the generic method, we can flexibly and independently construct an IBPAKE protocol by combining any pair of identity-based KEM/DEM and KE protocols irrespective of their underlying structures or hardness assumptions. For example, an integer factorization-based identity-based KEM and a pairing-based IBKS can be combined together.

As shown in our performance analysis, our IBPAKE protocol gives better performance, compared to [26].

Related Work. Since Diffie-Hellman key agreement protocol [13], KE protocols have been proposed to achieve various authentication goals [7, 8, 6, 20]. Authenticated KE protocols have been developed largely according to two authentication types, i.e., symmetric and asymmetric.

Symmetric authentication type assumes that participants have a same secret key in advance before running a KE protocol [7, 8, 6]. For example, we can consider password-based KE. Refer to [22] for a recent survey. Since the formal work of [6, 10] for PAKE, lots of research has been conducted to provide useful

features, e.g. resilience to server compromise [16], construction under standard assumption [21], and multi-party PAKE [1]. Recent research on PAKE protocols [9] focuses on meeting highly theoretical security requirement such as UC model [11] and the protocols are known to be relatively inefficient.

Asymmetric authentication type assumes that a participant has a secret key and its corresponding public key. The secret key is kept secret by the participant while the public key is set to be public and so anyone can access it. By construction, no information about the secret key should be extracted from the public key. For example, we can consider a standard public key based KE and identity-based KE [12].

Organization. The remainder of this paper is organized as follows. In Section 2, we briefly review some preliminaries. In Section 3 we give a security model for IBPAKE. In Section 4 we present an IBPAKE protocol over gap Diffie-Hellman groups and prove its security. In Section 5 we present a generalization of the IBPAKE protocol, i.e., a generic method to generate an IBPAKE protocol and its security. Finally we conclude in Section 6.

2 Preliminaries

In this section, we review bilinear maps and some assumptions related to our protocol. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two (multiplicative) cyclic group of prime order p. We assume that the discrete logarithm problems (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable.

Admissible Bilinear Map. We call $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ an *admissible bilinear* map if it satisfies the following properties:

- Bilinearity: $e(g^a, h^b) = e(g, h)^{ab}$ for all $g, h \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p^*$.
- Non-degenerancy: There exists $P \in \mathbb{G}_1$ such that $e(g, g) \neq 1$.
- Computability: There exists an efficient algorithm to compute $e(g, h)$ for all $g, h \in \mathbb{G}_1$.

The modified Weil and Tate pairings in elliptic curve are examples of the admissible bilinear maps.

Computational Diffie-Hellman (CDH) problem. A CDH problem in $\mathbb{G}_1$ is to compute g^{ab} when given g, g^a and g^b for some $a, b \in \mathbb{Z}_p^*$. More formally, the advantage of $\mathcal{A}$ is defined to be:

$$\mathsf{Adv}^{\mathrm{CDH}}_{\mathcal{A},\mathbb{G}_1}(\lambda) = \Pr\left[\mathcal{A}(g, g^a, g^b) = g^{ab} \mid a, b \leftarrow \mathbb{Z}_p^*; g \leftarrow \mathbb{G}_1\right].$$

Bilinear Diffie-Hellman (BDH) problem. A BDH problem in $[\mathbb{G}_1, \mathbb{G}_2, e]$ is to compute $e(g, g)^{abc}$ when given g, g^a, g^b, and g^c for some $a, b, c \in \mathbb{Z}_p^*$. More formally, the advantage of $\mathcal{A}$ is defined to be:

$$\mathsf{Adv}^{\mathrm{BDH}}_{\mathcal{A},[\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}]}(\lambda) = \Pr\left[\mathcal{A}(g, g^a, g^b, g^c) = e(g, g)^{abc} \mid a, b, c \leftarrow \mathbb{Z}_p^*; g \leftarrow \mathbb{G}_1\right].$$

We assume that the above CDH and BDH problems are intractable. That is, there is no PPT algorithm that solves these problems with non-negligible probability.

3 Security Model

We extend the security model of Bellare et al. [6], which is formal security of PAKE protocols in the realistic setting of concurrent sessions, to define a security model for our identity-based PAKE protocol.

Initialization. We assume that client C and server S have unique identities ID_C and ID_S from $\{0,1\}^\ell$, respectively. In the model, we allow client C to execute a protocol repeatedly with server S. An instance of C (resp. S) is represented by an oracle Π_C^s (resp. Π_S^s) for any $s \in \mathbb{N}$. The client C holds a password space $\mathsf{Password}$ of size $\mathcal{PW}$. The server S holds a vector (the so-called a verifier) $f(pw_C)$ and a secret key corresponding to ID_S from IBE, where f is a one-way function. The public parameters p and identities $ID_i (i \in \{C, S\})$ are known to client and server (and also to an adversary).

Partnering. Let sid_C^s be the concatenation of all messages sent and received by an oracle Π_C^s during the execution. For the concatenation the messages are ordered according to the sender's identity. Let partner identifier pid_C^s for instance Π_C^s be a set of the identities of the users with whom Π_C^s intends to establish a session key. The oracles Π_C^s and Π_S^t are *partnered* if and only if $\mathsf{pid}_C^s = \mathsf{pid}_S^t$ and $\mathsf{sid}_C^s = \mathsf{sid}_S^t$.

Adversarial model. An adversary $\mathcal{A}$ is a PPT algorithm that controls all the communications. The types of attacks that $\mathcal{A}$ can make are modelled in the following queries.

- $\mathsf{Extract}(ID_i)$: This query allows $\mathcal{A}$ to obtain the long-term secret key of ID_i where $i \notin \{C, S\}$.

- $\mathsf{Execute}(ID_C, ID_S)$: This query models passive attacks, where $\mathcal{A}$ eavesdrops an execution of the protocol. $\mathcal{A}$ retrieves the complete transcripts of an honest execution between C and S.

- $\mathsf{Send}(\Pi_i^s, m)$: This query is used to send a message m to instance Π_i^s (this models active attack on the part of the adversary). When Π_i^s receives m, it responds according to key agreement protocol.

- $\mathsf{Reveal}(\Pi_i^s)$: This query models *known key* attacks in the real system. $\mathcal{A}$ is given the session key for instance Π_i^s.

- $\mathsf{Corrupt}(ID_i)$: This query models exposure of the long-term secret key held by ID_i where $i \notin \{C, S\}$. $\mathcal{A}$ is assumed to be able to obtain the full private key, but cannot control the behaviour of ID_i directly (of course, once $\mathcal{A}$ has asked a query $\mathsf{Corrupt}(ID_i)$, the adversary may impersonate ID_i in subsequent Send queries.)

- $\mathsf{Test}(\Pi_i^s)$: This query is used to define the advantage of $\mathcal{A}$. When $\mathcal{A}$ asks this query to an instance Π_i^s, a random bit b is chosen; if $b = 1$ then the session key is returned. Otherwise a random string is returned. $\mathcal{A}$ is allowed to make a single Test query, at any time during the game.

In the model, we consider two types of adversaries according to their attack types. The attack types are simulated by the queries issued by an adversary. A *passive adversary* is allowed to issue Execute, Reveal, Corrupt, and Test queries, while an *active adversary* is additionally allowed to issue Send and Extract queries. Even though Execute query can be using Send queries repeatedly, we use Execute query for more concrete analysis.

FRESHNESS. An oracle Π_i^s is said *fresh* (or holds a *fresh* key ssk) if the following conditions hold:

1. $\mathsf{Corrupt}(ID_j)$ is not asked for all $j \in \mathsf{pid}_i^s$,
2. a session secret key ssk of Π_i^s is not revealed, and
3. if Π_i^s and Π_j^t are partnered, ssk of Π_j^t is not revealed.

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

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

- `Validity`: if all oracles in a session are partnered, the session keys of all oracles are same.
- `Key secrecy`: $\mathsf{Adv}_{\mathcal{P}}(\lambda)$ is bounded by $\frac{q_S}{\mathcal{PW}} + \epsilon(\lambda)$, where $\epsilon(\lambda)$ is negligible. q_S is the number of Send queries and $\mathcal{PW}$ is the size of the password space.

4 Proposed IBPAKE Protocol and Security Analysis

In this section, we propose an IBPAKE protocol, called iPAKE using the Boneh-Franklin IBE (BF-IBE) scheme [4]. The protocol makes an asymmetric key setting for a client and a server. That is, the server S has a long-term secret key generated from an IBE scheme, while the client C has no long-term secret key of high-entropy but only uses a human-memorizable password. In the following description, we denote by $x \leftarrow_R X$ the operation that picks an element x of set X uniformly at random.

4.1 Protocol Description

Our protocol, iPAKE consists of two phases, initialization and key establishment. In the initialization phase, system parameters and keys of iPAKE are generated. We assume that there exists a trusted key generation system (KGS) to generate a secret key for a given server's identity. In the key establishment phase, a client and a server execute a key exchange process using a password and the BF-IBE scheme.

Initialization Phase. A server, S obtains a secret key corresponding to its identity. A client, C registers a password as its authentication key.

- Setup: To generate IBE system parameters, the KGS chooses a random $\kappa \in \mathbb{Z}_q$ and a generator g in $\mathbb{G}_1$ and sets $g_{pub} = g^{\kappa}$. KGC also chooses four cryptographic hash functions $H_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1$, $H_2 : \mathbb{G}_2 \longrightarrow \{0,1\}^t$, $H_3 : \{0,1\}^* \longrightarrow \{0,1\}^t$, and $H_4 : \{0,1\}^* \longrightarrow \{0,1\}^l$ where t is a security parameter and l is the bit length of a session key. The system parameters p and the master secret key msk are given by

$$\mathsf{msk} = \kappa, \qquad \mathsf{p} = (\lambda, p, e, \mathbb{G}_1, \mathbb{G}_2, g, g_{pub}, H_1, H_2, H_3, H_4).$$

- Extract: To generate a private key for a server S with identity ID_S, using the master key msk, KGS computes $q_S = H_1(ID_S)$ and the private key $k_S = (q_S)^{\kappa}$. KGS finally sends k_S to S over a secure channel.

- Registration: A client C chooses a password $pw_C \in$ Password and sends $H_3(pw_C)$ to S over a secure channel.

Key Establishment Phase. When C and S want to establish a session key, they execute the following protocol (See Fig. 1).

1. The client C picks a random number x in Z_p. C computes $X = g^x$, $q_S = H_1(ID_S)$, $d_S = e(q_S, p_{pub})$, and $\delta = (d_S)^x$. Also, C computes $W = H_2(\delta) \oplus H_3(pw_C)$ by using his/her own password. C then sends $\langle ID_C, W, X \rangle$ to the server S.

Client C		Server S
$[ID_C, pw_C]$		$[ID_S, k_S], [ID_C, H_3(pw_C)]$
$x \xleftarrow{R} Z_p$		
$X = g^x,\ q_S = H_1(ID_S)$		
$d_S = e(q_S, g_{pub}),\ \delta = (d_S)^x$		
$W = H_2(\delta) \oplus H_3(pw_C)$	$\xrightarrow{ID_C, W, X}$	
		$\delta' = e(X, k_S)$
		$H_3(pw_C)' = W \oplus H_2(\delta')$
		$H_3(pw_C)' \stackrel{?}{=} H_3(pw_C)$
		$y \xleftarrow{R} Z_p$
	$\xleftarrow{Y}$	$Y = g^y$
$Z = Y^x$		$Z = X^y$
$\mathsf{sid}_C = W\|\|X\|\|Y$		$\mathsf{sid}_S = W\|\|X\|\|Y$
$ssk = H_4(ID_C\|\|ID_S\|\|\mathsf{sid}_C\|\|\delta\|\|Z)$		$ssk = H_4(ID_C\|\|ID_S\|\|\mathsf{sid}_S\|\|\delta\|\|Z)$

Fig. 1. Our iPAKE protocol

2. The server S computes $\delta' = e(X, k_S)$ and $H_3(pw_C)' = W \oplus H_2(\delta')$ by using received messages and its secret key k_S. S then checks if $H_3(pw_C)' = H_3(pw_C)$. If it is not true, S outputs FAIL and aborts. Otherwise, S picks a random number y in Z_N^*. S computes ($Y = g^y$, $Z = X^y$), and sends Y to the client C. Finally, S computes the session secret key $ssk = H_4(ID_C||ID_S||\mathsf{sid}_S||\delta||Z)$, where $\mathsf{sid}_S = W||X||Y$.
3. The client C computes the session secret key $ssk = H_4(ID_C||ID_S||\mathsf{sid}_C||\delta||Z)$, where $\mathsf{sid}_C = W||X||Y$ and $Z = Y^x$.

In the above protocol, the transcript (W, X) is a ciphertext which is the result of encryption performed on the plaintext $H_3(pw_C)$ using BF-IBE scheme, i.e,

$$CT = (W, X) \leftarrow \mathsf{BF\text{-}IBE.Enc}(ID_S, H_3(pw_C)) \text{ and}$$
$$H_3(pw_C) \leftarrow \mathsf{BF\text{-}IBE.Dec}(k_S, CT).$$

We focus on IBPAKE with implicit authentication. A KE protocol is said to achieve implicit key authentication if an entity (client or server) is assured that no other entities except partners can possibly learn the values δ and Z of particular secret keys. Note that implicit key authentication does not necessarily mean that partners have actually obtained the key. For explicit key authentication, we can easily apply the known techniques [20].

4.2 Security Analysis

Theorem 1. *Let $\mathcal{A}$ be an adversary for given ID_C and ID_S attack to iPAKE in the random oracle model. Suppose $\mathcal{A}$ makes at most q_{H_4}, q_E, and q_S queries*

to the H_4, Execute, and Sand oracle, respectively. Then,

$$\mathsf{Adv}^{\mathrm{IBPAKE}}_{\mathcal{A},\mathsf{iPAKE}}(t) \leq q_E \mathsf{Adv}^{\mathrm{BDH}}_{\mathcal{A},[\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}]}(t) + \frac{1}{2} q_{H_4} \mathsf{Adv}^{\mathrm{CDH}}_{\mathcal{A},\mathbb{G}_1}(t) + q_S \mathsf{Adv}_{\mathcal{A},\mathsf{BF\text{-}IBE}}(t) + \frac{q_S}{\mathcal{PW}}$$

where t is the adversary's running time and $\mathsf{Adv}_{\mathcal{A},\mathsf{BF\text{-}IBE}}(t)$ is the maximum advantage of $\mathcal{A}$ against the BF-IBE scheme.

Proof. Let $\mathcal{A}$ be an active adversary that gets an advantage in attacking iPAKE. The adversary $\mathcal{A}$ can get the advantage by following cases:

- Case 1. Finding the password, namely impersonating the client.
- Case 2. Computing a server's secret value, namely impersonating the server.
- Case 3. Breaking the protocol without altering transcripts.

In Case 1, the adversary $\mathcal{A}$ can get information about a particular session key by finding the password pw_C. There are two ways $\mathcal{A}$ can get information about the password; either $\mathcal{A}$ executes on-line dictionary attacks using Sand queries or $\mathcal{A}$ breaks a ciphertext (W, X) of the BF-IBE scheme. Let Succ_{pw} be the event that $\mathcal{A}$ succeeds in Case 1, and we assume that passwords are uniformly distributed. We then may obtain the probability of Succ_{pw} as follows:

$$\mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_{pw}] \leq q_S \mathsf{Adv}_{\mathcal{A},\mathsf{BF\text{-}IBE}}(t) + \frac{q_S}{\mathcal{PW}}.$$

In Case 2, to impersonate the server or to get information about a session key, $\mathcal{A}$ may try to compute $\delta = e(q_S, g_{pub})^x$ from the transcript X and the public values (q_S, g_{pub}) of the protocol from Execute queries. It is the same as solving the BDH problem. Therefore, the upper bound about the advantage of $\mathcal{A}$ from Case 2 is $q_E \mathsf{Adv}^{\mathrm{BDH}}_{\mathcal{A},[\mathbb{G}_1,\mathbb{G}_2,e]}(t)$.

Next, we consider the advantage from Case 3. Note that, to get any information of a session secret key ssk in the random oracle model, $\mathcal{A}$ has to ask $\langle ID_C || ID_S || \mathsf{sid} || \delta || Z \rangle$ to the hash oracle H_4. We can construct $\mathcal{B}$ which succeeds in solving the CDH problem using $\mathcal{A}$ as a subroutine. $\mathcal{B}$ receives a CDH instance $(\mathbb{G}, N, g, U = g^u, V = g^v)$. $\mathcal{B}$ chooses two identities (ID_C, ID_S), a master secret key $\kappa \in \mathbb{Z}_p^*$, and a password $pw_C \in \mathsf{Password}$ of the client's identity ID_C. $\mathcal{B}$ sets $g_{pub} = g^\kappa$, and gives (ID_C, ID_S) and system parameters to $\mathcal{A}$. $\mathcal{B}$ then runs $\mathcal{A}$, answering its oracle queries as follows:

- For queries $H_1(ID_i)$ proceed as follows: if $[ID_i, q_i]$ exist in a list $\mathsf{h_1}$-tuples, return q_i. Otherwise, return a random $q_i \in \mathbb{G}_1$ and store $[ID_i, q_i]$ in $\mathsf{h_1}$-tuples. (If $ID_i = ID_S$, we denote $q_i = q_S$.)
- For queries $H_2(\delta_i)$ proceed as follows: if $[\delta_i, \alpha_i]$ exist in a list $\mathsf{h_2}$-tuples, return α_i. Otherwise, return a random $\alpha_i \in \{0,1\}^t$ and store $[\delta_i, \alpha_i]$ in $\mathsf{h_2}$-tuples.
- For queries $H_3(pw_i)$ proceed as follows: if $[pw_i, \beta_i]$ exist in a list $\mathsf{h_3}$-tuples, return β_i. Otherwise, return a random $\beta_i \in \{0,1\}^t$ and store $[pw_i, \beta_i]$ in $\mathsf{h_3}$-tuples.

- For queries $H_4(ID_C||ID_S||\mathsf{sid}_i||\delta_i||Z_i)$, return a random $\gamma_i \in \{0,1\}^l$. Store $[\mathsf{sid}_i, \gamma_i]$ in a list $\mathsf{h_4}$-tuples.
- For queries $\mathsf{Extract}(ID_i)$ proceed as follows: find $[ID_i, q_i]$ in $\mathsf{h_1}$-tuples and return $k_i = q_i^\kappa$. (We assume Extract queries are preceded by H_1 queries.)
- For queries $\mathsf{Execute}(ID_C, ID_S)$ proceed as follows: choose random $a, b \in \mathbb{Z}_p^*$ and $\alpha, \beta \in \{0,1\}^t$. Compute $X = Ug^a$, $\delta = e(X, k_S)$, $W = \alpha \oplus \beta$, and $Y = Vg^b$. Return $\langle ID_C, W, X, Y\rangle$ and store $[\delta, \alpha]$ and $[pw_C, \beta]$ in $\mathsf{h_2}$-tuples and $\mathsf{h_3}$-tuples, respectively. (We assume Execute queries are not preceded by H_2, H_3 queries.)
- Send, Corrupt, Reveal, and Test queries are answered honestly.

The success probability of $\mathcal{B}$ depends on the event query that $\mathcal{A}$ issues H_4 oracle query on $ID_C||ID_S||\mathsf{sid}||\delta||Z$, where sid is a return value of Execute query and $Z = g^{(u+a)(v+b)}$. (Note that, $g^{uv} = Z/U^bV^ag^{ab}$.) If the advantage of $\mathcal{A}$ in Case 3 is ϵ, then $\mathcal{A}$ issues a query for $H_4(ID_C||ID_S||\mathsf{sid}||\delta||Z)$ with probability at least 2ϵ, i.e., $\mathsf{Pr}_{\mathcal{A}}[\mathsf{query}] \leq 2\epsilon$. (The details are in [4].) Thus, the provability that $\mathcal{B}$ outputs g^{uv} from the list $\mathsf{h_4}$-tuples is at least $2\epsilon/q_{H_4}$. Therefore, the upper bound about the advantage of $\mathcal{A}$ from Case 3 is $\frac{1}{2}q_{H_4}\mathsf{Adv}^{\mathrm{BDH}}_{\mathcal{A},[\mathbb{G}_1,\mathbb{G}_2,e]}(t)$. Finally, we have

$$\mathsf{Adv}^{\mathrm{IBPAKE}}_{\mathcal{A},\mathsf{iPAKE}}(t) \leq q_E Adv^{\mathrm{BDH}}_{\mathcal{A},[\mathbb{G}_1,\mathbb{G}_2,\mathrm{e}]}(t) + \frac{1}{2}q_{H_4} Adv^{\mathrm{CDH}}_{\mathcal{A},\mathbb{G}_1}(t) + q_S Adv_{\mathcal{A},\mathsf{BF\text{-}IBE}}(t) + \frac{q_S}{\mathcal{PW}}.$$

□

5 Generic Construction

In this section, we present a generic method to construct an IBPAKE protocol from an identity-based KEM/DEM scheme. Before describing our construction in detail, we present an identity-based KEM/DEM scheme which extends the identity-based KEM scheme [2].

5.1 Identity-based KEM/DEM scheme

An identity-based KEM/DEM scheme is specified by six polynomial time algorithms, Setup, Extract, IBKEM.Enc, IBDEM.Enc, IBKEM.Dec, and IBDEM.Dec.

- $\mathsf{Setup}(\lambda)$. This algorithm takes a security parameter λ as input and returns a master secret key, msk and its corresponding public parameter, p. $\mathcal{K}_P$ is a plaintext space associated with p.
- $\mathsf{Extract}(\mathsf{msk}, \mathsf{p}, ID)$. This algorithm takes the master secret key, msk, public parameter, p, and an identity, ID as input. It returns a secret key, sk_{ID}.
- $\mathsf{IBKEM.Enc}(\mathsf{p}, ID)$. This algorithm takes p and an identity, ID as input. It returns a random *one-time* key, $k_p \in \mathcal{K}_P$ and its ciphertext, Δ_{KEM}.
- $\mathsf{IBDEM.Enc}(k_p, \mathsf{p}, m)$. This algorithm takes p, a key k_p, and a message, m as input. It returns a ciphertext Δ_{DEM} for m.

- $\mathsf{IBKEM.Dec}(sk_{ID}, \mathsf{p}, \Delta_{\mathsf{KEM}})$. This algorithm takes p, a private key sk_{ID}, and a ciphertext Δ_{KEM} as input. It returns a key, k_p.
- $\mathsf{IBDEM.Dec}(k_p, \mathsf{p}, \Delta_{\mathsf{DEM}})$. This algorithm takes p, a key k_p, and a ciphertext Δ_{DEM} as input. It returns a message m.

In the above identity-based KEM/DEM scheme, the full ciphertext for m is $(\Delta_{\mathsf{KEM}}, \Delta_{\mathsf{DEM}})$. Several IBE schemes [4, 24, 3, 25, 14] can be represented in the identity-based KEM/DEM framework [2].

5.2 Our Generic Construction for IBPAKE

The generic method to construct an IBPAKE protocol from an identity-based KEM/DEM scheme is described as follows:

1. C runs $\mathsf{IBKEM.Enc}(\mathsf{p}, ID_S)$ to obtain a random one-time key k_p and its ciphertext Δ_{KEM}. C also runs $\mathsf{IBDEM.Enc}(k_p, \mathsf{p}, f(pw_C))$ to obtain a ciphertext Δ_{DEM} for a message $f(pw_C)$, where f is a one-way function. C then sends ID_C and $\Delta = (\Delta_{\mathsf{KEM}}, \Delta_{\mathsf{DEM}})$ to S.
2. S runs $\mathsf{IBKEM.Dec}(sk_{IDs}, \mathsf{p}, \Delta_{\mathsf{KEM}})$ to obtain the one-time key k_p, where sk_{IDs} is a server's secret key generated by $\mathsf{Extract}(\mathsf{msk}, \mathsf{p}, ID_S)$. S obtains $f(pw_C)'$ by running $\mathsf{IBDEM.Dec}(k_p, \mathsf{p}, \Delta_{\mathsf{DEM}})$ and then checks if $f(pw_C)' = f(pw_C)$. If it is not true, S outputs FAIL and aborts. Otherwise, S generates a random number r_S from a random number generator, and sends r_S to C. Finally, S computes a session secret key $ssk = H(ID_C||ID_S||\Delta||r_S||k_p)$.
3. C computes a session secret key, $ssk = H(ID_C||ID_S||\Delta||r_S||k_p)$.

Client C		Server S
$[ID_C, pw_C]$		$[ID_S, sk_{IDs}], [ID_C, f(pw_C)]$
$(k_p, \Delta_{\mathsf{KEM}}) \leftarrow \mathsf{IBKEM.Enc}(\mathsf{p}, ID_S)$		
$\Delta_{\mathsf{DEM}} \leftarrow \mathsf{IBDEM.Enc}(k_p, \mathsf{p}, f(pw_C))$		
$\Delta = (\Delta_{\mathsf{KEM}}, \Delta_{\mathsf{DEM}})$	$\xrightarrow{ID_C, \Delta}$	
		$k_p \leftarrow \mathsf{IBKEM.Dec}(sk_{IDs}, \mathsf{p}, \Delta_{\mathsf{KEM}})$
		$f(pw_C)' \leftarrow \mathsf{IBDEM.Dec}(k_p, \mathsf{p}, \Delta_{\mathsf{DEM}})$
		Check $f(pw_C)' \stackrel{?}{=} f(pw_C)$
	$\xleftarrow{r_S}$	$r_S \leftarrow \mathsf{RanGen}(\lambda)$
$ssk = H(ID_C\|\|ID_S\|\|\Delta\|\|r_S\|\|k_p)$		$ssk = H(ID_C\|\|ID_S\|\|\Delta\|\|r_S\|\|k_p)$

Fig. 2. Generic construction for IBPAKE

The above generic protocol provides *half forward secrecy*. That is, if the secret key of the server is compromised, then all session keys are revealed using protocol transcripts. However, the exposure of client's password is not helpful to get the

information about previous session keys. In practice, it is reasonable to assume that low-power devices held by clients are vulnerable to attacks, while a server is powerful and so more secure.

However, our generic construction can be modified to provide *forward secrecy* by using additional ephemeral DH KE. In fact, iPAKE can be viewed as a specific version of our generic protocol, in which BF-IBE is used as an identity-based KEM/DEM scheme. Note that BF-IBE can be represented as a identity-based KEM/DEM scheme as follows:

IBE-Enc

$$(k_p = e(q_S,\, g_{pub})^x,\, \Delta_{\mathsf{KEM}} = g^x) \leftarrow \mathsf{IBKEM.Enc}(\mathsf{p}, ID)$$
$$\Delta_{\mathsf{DEM}} = m \oplus H_2(k_p) \leftarrow \mathsf{IBDEM.Enc}(k_p, \mathsf{p}, m)$$

IBE-Dec

$$k_p = e(\Delta_{\mathsf{KEM}},\, sk_{ID}) \leftarrow \mathsf{IBKEM.Dec}(sk_{ID}, \mathsf{p}, \Delta_{\mathsf{KEM}})$$
$$m = \Delta_{\mathsf{DEM}} \oplus H_2(k_p) \leftarrow \mathsf{IBDEM.Dec}(k_p, \mathsf{p}, \Delta_{\mathsf{DEM}}).$$

The ciphertext element $X(= \Delta_{\mathsf{KEM}} = g^x)$ of BF-IBE is a public ephemeral DH value. Our iPAKE performs DH KE to compute g^{xy} by using g^y, instead of use of the random value r_S. It is known that the ephemeral key, g^{xy} is sufficient for perfect forward secrecy [20]. Similarly, by using such identity-based KEM/DEM schemes [24, 3, 25, 14] that have public DH values in ciphertexts, we can also construct an IBPAKE protocol with perfect forward secrecy.

Considering only half forward secrecy, we can construct *one-pass* IBPAKE protocol by omitting the second pass. A client does not receive a server response, r_S and thus a session key is defined as $ssk = H(ID_C||ID_S||\Delta||k_p)$. We can show that this one-pass protocol provides *key indistinguishability* (KI). A brief proof sketch is given as follows (For more details, refer to the full version of this paper). Assume that $\mathcal{B}$ is an IND-ID-CPA adversary to an identity-based KEM/DEM scheme. Also, $\mathcal{A}$ is a probabilistic polynomial time adversary attacking the one-pass IBPAKE protocol:

> Initially, $\mathcal{B}$ sets up system parameter and gives it with identities (ID_C, ID_S) to $\mathcal{A}$. $\mathcal{B}$ simulates an attack environment for the one-pass IBPAKE by providing Extract, Execute, Send, Reveal, Corrupt and hash queries. $\mathcal{A}$ get transcripts of an honest execution of the protocol or a common key computed from an execution of the protocol according to queries. When $\mathcal{A}$ makes Test query for a fresh oracle described in Section 3, $\mathcal{B}$ gives a session key ssk_b after selecting a bit $b \in \{0, 1\}$, where ssk_0 is an honest session key of the protocol and ssk_1 is a random element from a key space. If $\mathcal{A}$ succeeds the attack then $\mathcal{A}$ issued hash oracle query on $H(ID_C||ID_S||\Delta||k_p)$. (The details are in [4].) Therefore, if a KI adversary exists then an IND-ID-CPA adversary exists.

To enhance the security while preserving one-pass, a time stamp can be used [19]. It will prevent so called 'known session key attacks' from replay of protocol transcripts.

5.3 Comparison

We now compare performance between our IBPAKE protocol and the previous IBPAKE protocol [26][5]. To be fair, we assume that the IBPAKE protocols are constructed by using the BF-IBE [4] and Gentry IBE [14], respectively. The following table summarizes the results.

Protocol		Client			Server		
		e	`Exp`	`Mul`	e	`Exp`	`Mul`
IBPAKE [26]	based on [4]	1	6	1	1	4	1
	based on [14]	2	8	3	1	5	3
Ours	based on [4](iPAKE)	1	3	0	1	2	0
	based on [14]	2	5	2	1	3	2

Table 1. Comparison of IBPAKE protocols
(e: pairing operation, `Exp`: modular exponentiation, `Mul`: modular multiplication, w.l.o.g, `Exp` and `Mul` contain the multiplication and the addition in a gap Diffie-Hellman group, respectively.)

As shown in Table 1, although the IBPAKE protocol [26] can be constructed by using the IBE schemes [4, 14], our protocols are more efficient.

6 Conclusion

We have proposed efficient IBPAKE protocols using identity-based KEM/DEM. A client can do an easy authentication based on only a human-memorable password and server's public identity. Our protocols give resistance to server impersonation attacks. That is, even if a password is revealed from a client, a server impersonation attack can be prevented effectively. The proposed protocols outperform the best-known IBPAKE protocol.

References

1. M. Abdalla and D. Pointcheval. A scalable password-based group key exchange protocol in the standard model, Asicrypt'06, LNCS 4284, pp 332-347, Springer, 2006.
2. X. Boyen, A Tapestry of Identity-based Encryption: Practical Frameworks Compared, J. Applied Cryptography, Vol 1, No. 1, pp.3-21, Inderscience, 2008.
3. D. Boneh and X. Boyen, Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles, Eurocrypt??4, LNCS 3027, pp.223-238, Springer, 2004.
4. D. Boneh and M. Franklin, Identity-based encryption from the Weil pairing, Crypto??1, LNCS 2139, pp.213-229, Springer, 2001.
5. S. M. Bellovin and M. Merritt, Encrypted key exchange: Password-based protocol secure against dictionary attack, 1992 IEEE Symposium on Research in Security and Privacy, pp.72-84, 1992.

[5] To the best of our knowledge, this is the only protocol with provable security

6. M. Bellare, D. Pointcheval and P. Rogaway, Authenticated key exchange secure against dictionary attack, Eurocrypt'00, LNCS 1807, pp.139-155, Springer, 2000.
7. M. Bellare and P. Rogaway, Entity authentication and key distribution, Crypto'93, LNCS 773, pp.232-249, Springer, 1994.
8. M. Bellare and P. Rogaway, Provably-Secure Session Key Distribution : The Three Party Case, STOC'95, pp.57-66, 1995.
9. F. Benhamouda, O. Blazy, C. Chevalier, D. Pointcheval1, and D. Vergnaud, New Techniques for SPHFs and Efficient One-Round PAKE Protocols, Crypto'13, LNCS 8042, pp.449-475, Springer, 2013.
10. V. Boyko, P. D. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. Eurocrypt'00, LNCS 1807, pp.156-171. Springer, 2000.
11. R. Canetti, S. Halevi, J. Katz, Y. Lindell, and P. D. MacKenzie. Universally composable password-based key exchange. Eurocrypt'05, LNCS 3494, pp.404-421. Springer, 2005.
12. K. Y. Choi, J. Y. Hwang and D. H. Lee, Efficient ID-based group key agreement with bilinear maps, PKC'04, LNCS 2947, pp.130-144, Springer, 2004.
13. W. Diffie and M. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory 22(6), pp.644-654, 1976.
14. C. Gentry, Practical identity-based encryption without random oracles, Eurocrypt'06, LNCS 4004, pp.445-464, Springer, 2006.
15. L. A. Gong, T. M. Lomas, R. Needham, and J. Saltzwe, Protecting poorly chosen secrets from guessing attacks. IEEE J. Sel. Areas Commun., 11 (5), pp. 648-656, 1993.
16. C. Gentry, P. MacKenzie, and Z. Ramzan, A Method for Making Password-Based Key Exchange Resilient to Server Compromise, Crypto'06, LNCS 4117, pp.142-159, Springer, 2006.
17. S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. ACM Trans. Information and System Security, 2(3), pp.230-268, 1999.
18. R. Housley and T. Polk, Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure, WILEY, 2001.
19. S. Haber and W. Stornetta, How to Time-Stamp a Digital Document, Crypto'90, LNCS 537, pp.437-455, Springer, 1991.
20. J. Katz and M. Yung, Scalable protocols for authenticated group key exchange, Crypto'03, LNCS 2729, pp.85-113, Springer, 2003.
21. J. Katz, R. Ostrovsky, and M. Yung. Efficient password-authenticated key exchange using human-memorable passwords. Eurocrypt'01, LNCS 2045, pp.475-494, Springer, 2001.
22. D. Pointcheval, Password-Based Authenticated Key Exchange, PKC 2012, LNCS 7293, pp. 390-397, Springer, 2012.
23. A. Shamir, Identity based cryptosystems and signature schemes, Crypto'84, LNCS 196, pp. 47-53, Springer, 1984.
24. R. Sakai, and M. Kasahara, ID based cryptosystems with pairing over elliptic curve, Cryptology ePrint Archive, Report 2003/054. http://eprint.iacr.org/2003/054.
25. B. Waters, Efficient Identity-Based Encryption Without Random Oracles, Eurocrypt'05, LNCS 3494, pp.114-127, Springer, 2005.
26. X. Yi, R. Tso, and E. Okamoto, Identity-based password-authenticated key exchange for client/server model, SECRYPT'12, pp.45-54, 2012.