A Strengthened PAKE Protocol with Identity-Based Encryption*

SeongHan SHIN†(a), Member

SUMMARY In [2], Choi et al. proposed an identity-based password-authenticated key exchange (iPAKE) protocol using the Boneh-Franklin IBE scheme, and its generic construction (UKAM-PiE) that was standardized in ISO/IEC 11770-4/AMD 1. In this paper, we show that the iPAKE and UKAM-PiE protocols are insecure against passive/active attacks by a malicious PKG (Private Key Generator) where the malicious PKG can find out all clients’ passwords by just eavesdropping on the communications, and the PKG can share a session key with any client by impersonating the server. Then, we propose a strengthened PAKE (for short, SPAIBE) protocol with IBE, which prevents such a malicious PKG’s passive/active attacks. Also, we formally prove the security of the SPAIBE protocol in the random oracle model and compare relevant PAKE protocols in terms of efficiency, number of passes, and security against a malicious PKG.

key words: PAKE, IBE, PKG, Online/Offline dictionary attacks, provable security

1. Introduction

Password-based authenticated key exchange protocols provide password authentication and establishment of session keys to be used for protecting subsequent communications. Because password authentication is commonly used in the real world, these protocols have been widely deployed in many applications (e.g., TLS, SSH, IPsec, WPA, HTTP). Since the appearance of EKE[3], [4] (known as PAKE), many applications (e.g., TLS, SSH, IPsec, WPA, HTTP). In the real world, these protocols have been widely deployed in many applications (e.g., TLS, SSH, IPsec, WPA, HTTP). Since the appearance of EKE[3], [4] (known as PAKE), such protocols have been extensively studied (see [5], [6] and references therein) in order to be secure against passive/active attacks as well as offline dictionary attacks on human-memorable passwords. A PAKE protocol is said to be secure if an adversary’s capability is restricted to online dictionary attacks by interacting with a party and testing a possible password candidate one by one. Up until now, PAKE protocols have received much attention in the literature, and some protocols have been standardized in IEEE 1363.2 [7], ISO/IEC 11770-4 [8], IETF [9], ITU-T [10].

At the same time, there are several approaches to strengthen the security of the password-based authenticated key exchange protocols by combining other cryptographic primitives (e.g., [11], [12]). Such an example is using identity-based cryptography [13–16] in which publicly known information (e.g., name, identifier, email/IP address) is used as a party’s public key, and the party’s secret key is issued by a PKG (Private Key Generator). Actually, identity-based cryptography can be well fitted in the PAKE setting where a client, who remembers his/her password, also should know the identity of the corresponding server, which has the client’s password. In [17], Yi et al. proposed an identity-based PAKE protocol using IBE (Identity-Based Encryption). Later, Choi et al. [2] proposed another identity-based PAKE (called, iPAKE) protocol using the Boneh-Franklin IBE scheme [15], [18], and its generic construction using an identity-based KEM/DEM scheme [19]. Also, Hwang et al. [20] proposed identity-based PAKE protocols constructed from IBS (Identity-Based Signature). In addition to the security of password authentication, these protocols [2], [17], [20] provide another layer of security, meaning that it is not possible for an adversary, who gets the client’s password, to impersonate the server.

1.1 Motivation and Our Contributions

In [2], Choi et al. proposed the iPAKE protocol and its generic construction where the latter construction named as ‘Unbalanced Key Agreement Mechanism with Password and Identity-based Encryption (UKAM-PiE)’ was standardized in ISO/IEC 11770-4/AMD 1 [21]. So, it is of significant importance to thoroughly analyze the security of these protocols.

In this paper, we first revisit the iPAKE protocol [2] using the Boneh-Franklin IBE scheme [15], [18], and the UKAM-PiE protocol in ISO/IEC 11770-4/AMD 1 [21]. After describing the iPAKE and UKAM-PiE protocols, we show that they are insecure against passive/active attacks by a malicious PKG (Private Key Generator) where the malicious PKG can find out all clients’ passwords by just eavesdropping on the communications, and the PKG can share a session key with any client by impersonating the server. Then, we propose a strengthened PAKE (for short, SPAIBE) protocol with IBE, which provides security against passive/active attacks by a malicious PKG. Also, we formally prove the security of the SPAIBE protocol in the random oracle model [22] and compare the PAKE protocols using the BF-IBE scheme (PAKE-CS [17], iPAKE [2], UKAM-PiE [21], SPAIBE) in terms of efficiency, number of passes and security against a malicious PKG.

DOI: 10.1587/transinf.2022NGP0009
2. Preliminaries

2.1 Notation

Let $k \in \mathbb{N}$ and $\lambda \in \mathbb{N}$ be the security parameters. Let $\{0, 1\}^*$ be the set of finite binary strings and $\{0, 1\}^k$ be the set of binary strings of length $k$. Let $A || B$ be the concatenation of $A$ and $B$. If $U$ is a set, then $u \in U$ indicates the process of selecting $u$ at random and uniformly over $U$. If $U$ is a function (whatever it is), then $u = U$ indicates the process of assigning the result to $u$. Let $\mathbb{D}_{pw}$ be a dictionary size of passwords whose cardinality is $N$. Let $C$ and $S$ be the identities of client and server, respectively, with each $ID \in \{0, 1\}^*$. Also, let $G_1$ and $G_2$ be two (multiplicative) cyclic groups of order $q$ for some large prime $q$. An admissible (symmetric) bilinear map $e : G_1 \times G_1 \rightarrow G_2$ has the following properties: 1) Bilinear: For all $g_1, g_2 \in G_1$ and all $\alpha, \beta \in \mathbb{Z}_q^*$, $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha \beta}$; 2) Non-degenerate: For all pairs $g_1, g_2 \in G_1$, $e(g_1, g_2) \neq 1$. If $g$ is a generator of $G_1$, then $e(g, g)$ is a generator of $G_2$; and 3) Computable: There is an efficient algorithm to compute $e(g_1, g_2)$ for any $g_1, g_2 \in G_1$. Note that the modified Weil and Tate pairings on elliptic curves are examples of the admissible bilinear map [18].

2.2 Computational Assumptions

First, we define the Computational Diffie-Hellman (CDH) problem. Let $G_1$ be the group generation algorithm that takes as input $1^\lambda$ and outputs a group description $(G_1, q, g)$ where $G_1$ is a finite cyclic group of prime order $q$ with $g$ as a generator.

Definition 1: (CDH Problem) Let $G_1$ be the group generation algorithm described above. A $(t_1, \epsilon_1)$-CDH$_{G_1}$ adversary is a probabilistic polynomial time (PPT) machine $B$, running in time $t_1$, such that its success probability $\text{Succ}_{\text{CDH}}(B)$, given random elements $g^a$ and $g^b$ to output $g^{ab}$, is greater than $\epsilon_1$. We denote by $\text{Succ}_{\text{CDH}}(t_1)$ the maximal success probability over every adversaries, running within time $t_1$. The CDH problem states that $\text{Succ}_{\text{CDH}}(t_1) \leq \epsilon_1$ for any $t_1/\epsilon_1$ not too large.

Next, we define the Bilinear Diffie-Hellman (BDH) problem. Let $G_2$ be the BDH group generation algorithm that takes as input $1^\lambda$ and outputs a group description $(G_1, G_2, e, q, g)$ where $G_1$ and $G_2$ are two groups of prime order $q$, $e : G_1 \times G_1 \rightarrow G_2$ is an admissible bilinear map and $g$ is a generator of $G_1$.

Definition 2: (BDH Problem) Let $G_2$ be the BDH group generation algorithm described above. A $(t_2, \epsilon_2)$-BDH$_{G_1, G_2, e}$ adversary is a probabilistic polynomial time (PPT) machine $B$, running in time $t_2$, such that its success probability $\text{Succ}_{\text{BDH}}(B)$, given random elements $g^a$, $g^b$ and $g^c$ to output $e(g, g)^{ab}$, is greater than $\epsilon_2$. We denote by $\text{Succ}_{\text{BDH}}(t_2)$ the maximal success probability over every adversaries, running within time $t_2$. The BDH problem states that $\text{Succ}_{\text{BDH}}(t_2) \leq \epsilon_2$ for any $t_2/\epsilon_2$ not too large.

2.3 An Identity-Based Encryption Scheme

In this subsection, we define the syntax of identity-based encryption (IBE) and describe the Boneh-Franklin IBE (BF-IBE) scheme [15], [18] that is the most efficient construction among IBE schemes (see [19]).

Definition 3: (Identity-Based Encryption) An identity-based encryption (IBE) scheme is a quadruple of probabilistic polynomial time algorithms $(\text{Setup}_{\text{IBE}}, \text{Extract}, \text{Encrypt}, \text{Decrypt})$ such that:

- $\text{Setup}_{\text{IBE}}$: The setup algorithm takes as input $1^\lambda$ and outputs public parameters $pp_{\text{IBE}}$ and a master secret key $msk$ where $(mpk, msk)$ is a pair of master public/secret keys and $mpk$ is included in $pp_{\text{IBE}}$. Also, the public parameters include descriptions of a finite message space $M$ and a finite ciphertext space $C$. The $pp_{\text{IBE}}$ will be publicly known, while the $msk$ will be known only to the Private Key Generator (PKG).
- $\text{Extract}$: The key extraction algorithm takes as input $pp_{\text{IBE}}$, $msk$ and an identity $ID \in \{0, 1\}^*$, and outputs a private key $d_{ID}$ corresponding to the user with this identity.
- $\text{Encrypt}$: The encryption algorithm takes as input $pp_{\text{IBE}}$, an identity $ID$ and a message $M \in M$, and outputs a ciphertext $C \in C$.
- $\text{Decrypt}$: The decryption algorithm takes as input $pp_{\text{IBE}}$, $C$ and a private key $d_{ID}$, and outputs $M \in M$.

It is required that $M = \text{Decrypt}(pp_{\text{IBE}}, C, d_{ID})$ for all $ID \in \{0, 1\}^*$ and $M \in M$ where $C = \text{Encrypt}(pp_{\text{IBE}}, ID, M)$.

Also, we define the semantic security (i.e. semantic security against adaptively chosen identity and message attacks) for an IBE scheme.

Definition 4: (Semantic Security of IBE) We say that an IBE scheme $IBE$ is semantically secure (IND-CPA secure) if, for any PPT adversary $B$, an advantage

$$\text{Adv}_{IBE}^{\text{ind.cpaa}}(B) = 2 \Pr[b^* = b] - 1$$

of the adversary $B$ against the scheme $IBE$ is negligible in the following game:

- $\text{Setup}_{\text{Challenger}}$: A challenger runs the setup algorithm $\text{Setup}_{\text{IBE}}$ and gives the adversary public parameters $pp_{\text{IBE}}$. The master secret key $msk$ is kept by the challenger.
- $\text{Phase 1}$: The adversary issues key extraction queries $\text{ID}_1, \ldots, \text{ID}_m$. The challenger responds with a private key $d_{\text{ID}_i}$ corresponding to $\text{ID}_i$ by running the key extraction algorithm $\text{Extract}$. These queries may be asked adaptively.
- $\text{Challenge}$: When Phase 1 is over, the adversary outputs two equal length messages $M_0, M_1 \in M$ and an identity $\text{ID}$. The only constraint is that $\text{ID}$ did not
appear in any key extraction query in Phase 1. The challenger chooses a random bit \( b \in \{0, 1\} \) and sends \( C = \text{Encrypt}(pp_{\text{IBE}}, \text{ID}, M_b) \) as a challenge to the adversary.

- Phase 2: The adversary issues more key extraction queries \( \text{ID}_{m+1}, \cdots, \text{ID}_n \). The only constraint is that \( \text{ID}_i \neq \text{ID} \). The challenger responds as in Phase 1.
- Guess: Finally, the adversary outputs a guess \( b' \in \{0, 1\} \) and wins the game if \( b = b' \).

### 2.3.1 Boneh-Franklin IBE (BF-IBE)

Here, we describe the Boneh-Franklin IBE (BF-IBE) scheme [15], [18] that is proven to be IND-CPA secure (i.e., \( \text{Adv}_{\text{BF-IBE}}^{\text{IND-CPA}}(B) \)) in the random oracle model [22] under the BDH problem in Definition 2.

- Setup: The setup algorithm on input \( 1^\lambda \) outputs public parameters \( pp_{\text{IBE}} \) and a master secret key \( msk \)

  \( \text{ Setup}_{\text{IBE}}(\lambda) \) is run by running the setup algorithm \( \text{Setup}_{\text{IBE}} \) of Sect. 2.3.1 where \( H', H : \{0, 1\}^n \rightarrow \{0, 1\}^k \) are descriptions of additional cryptographic hash functions. It outputs \( pp = (pp_{\text{IBE}}, H', H) \).

- Extract: The key extraction algorithm, on input \( pp_{\text{IBE}}, \text{ID} \) and an identity \( \text{ID} \), computes \( Qz = G(\text{ID}) \) and outputs a private key \( d_{\text{ID}} \) as in \( \text{Setup}_{\text{IBE}} \).

- Encrypt: The encryption algorithm, on input \( pp_{\text{IBE}}, \text{ID}, M \), encrypts \( M \) and outputs ciphertext \( C = C_0 \).

In the above, the consistency of the BF-IBE scheme can be easily checked by

\[
\delta = e(d_{\text{ID}}, U_1) = e(Q_{\text{ID}}^\ast, g^\ast) = e(Q_{\text{ID}}, g)\left(\ast\right)
\]

\[= e(Q_{\text{ID}}, g^\ast) = g_{\text{ID}}^\ast. \tag{1}
\]

### 3. The iPAKE Protocol

In this section, we describe the iPAKE protocol [2] which consists of Initialization and Key Establishment phases.

#### 3.1 Initialization

In this phase, it executes the following three processes Setup, Extract and Registration.

#### 3.1.1 Setup

The Setup on input \( 1^\lambda \) outputs public parameters \( pp \) and a master secret key \( msk \) by running the setup algorithm \( \text{Setup}_{\text{IBE}} \) of Sect. 2.3.1 where \( H', H : \{0, 1\}^n \rightarrow \{0, 1\}^k \) are descriptions of additional cryptographic hash functions. It outputs \( pp = (pp_{\text{IBE}}, H', H) \).

#### 3.1.2 Extract

The key extraction algorithm Extract (run by PKG), on input the public parameters \( pp_{\text{IBE}} \), the master secret key \( msk \) and an identity \( \text{ID} \), computes \( Qz = G(\text{ID}) \) and outputs a private key \( d_{\text{ID}} \) as in \( \text{Setup}_{\text{IBE}} \).

#### 3.1.3 Registration

In this phase, \( C \) randomly chooses his/her password \( pw \) from a password file and sends \( (C, H'(pw)) \) to server \( S \). Then, the server stores \( (C, H'(pw)) \) to a password file. This registration process should be done securely between client \( C \) and server \( S \).

#### 3.2 Key Establishment

In this phase, client \( C \) and server \( S \) execute the iPAKE protocol over insecure networks in order to share a session key. This phase of iPAKE has three steps as below:

- **Step 1.** The client \( C \) runs the encryption algorithm Encrypt of Sect. 2.3.1 on input \( pp_{\text{IBE}} \), an identity \( \text{ID} \) and a message \( H'(pw) \). Then, \( C \) sends \( (C, U_1, U_2) \) to server \( S \).

- **Step 2.** After receiving a message \( (C, U_1, U_2) \) from client \( C \), server \( S \) runs the decryption algorithm Decrypt of Sect. 2.3.1 on input \( pp_{\text{IBE}}, (U_1, U_2) \) and a private key \( d_{\text{ID}} \). If \( (U_2 \oplus H(\delta)) \neq H'(pw) \), the server aborts the protocol. Otherwise, \( S \) computes a session key \( SK = H_{3}(C||\text{ID}||sid|||Z) \), where \( sid = U_1||U_2||Y \), and then sends \( (S, Y) \) to the client.

- **Step 3.** After receiving a message \( (S, Y) \) from server \( S \), client \( C \) computes \( Z = Y' \) and a session key \( SK = H_{3}(C||\text{ID}||sid|||Z) \) where \( sid = U_1||U_2||Y \).

**Remark 1:** In [2], Choi et al. proposed the iPAKE protocol and its generic construction using an identity-based KEM/DEM scheme where the latter construction named as ‘Unbalanced Key Agreement Mechanism with Password and Identity-based Encryption (UKAM-PiE)’ was standardized in ISO/IEC 11770-4/AMD 1 [21]. The UKAM-PiE protocol, instantiated with the BF-IBE scheme [15], [18], is somewhat different from the iPAKE protocol as follows:

**Step 1’:** First, client \( C \) chooses a random element \( x \leftarrow Z_q^* \) and server \( S \) sends \( (C, x) \) to client \( C \).

Note that IBE schemes (e.g., [18], [23]–[26]) can be represented in the identity-based KEM/DEM framework [19].
4. Passive/Active Attacks on iPAKE and UKAM-PiE

This section shows passive and active attacks by a malicious PKG (Private Key Generator) on the iPAKE [2] and UKAM-PiE [21] protocols.

4.1 A Passive Attack on iPAKE

Here, we show that a malicious PKG can find out all clients’ passwords by just eavesdropping on the communications of the iPAKE protocol. After eavesdropping on the first message \((C, U_1, U_2)\) in the Key Establishment phase, the malicious PKG who has the master secret key \(z\) can decrypt the ciphertext \((U_1, U_2)\) since \(d_S \equiv (G(S))^z\). With all possible password candidates, the PKG can find out the client’s password \(pw\) by performing offline dictionary attacks on \(H’(pw) = U_2 \oplus H(e(d_S, U_1))\). Of course, these offline dictionary attacks can be used for all clients who registered to server \(S\).

4.2 An Active Attack on iPAKE

Here, we show that a malicious PKG can share a session key with any client by impersonating the server in the iPAKE protocol. After receiving the first message \((C, U_1, U_2)\) in the Key Establishment phase, the malicious PKG who has the master secret key \(z\) just executes Step 2 except the check of \(W \neq H’(pw)\) and then can share the same session key \(SK_C = H_3(C||S||sid||b||Z)\) with client \(C\). Note that in this active attack the PKG does not need to perform offline dictionary attacks at all.

4.3 Passive/Active Attacks on UKAM-PiE

As in the same way of Sects. 4.1 and 4.2, a malicious PKG can perform passive and active attacks on the UKAM-PiE protocol [21].

5. A Strengthened PAKE Protocol with IBE

In this section, we propose a strengthened PAKE (for short, SPAIBE) protocol with IBE which provides security against passive/active attacks by a malicious PKG (Private Key Generator). A main idea of the SPAIBE protocol is 1) to double mask a Diffie-Hellman public key on client \(C\) where the first mask is performed with the password verification data and the second one is with the encryption algorithm Encrypt of BF-IBE [15], [18], and 2) to make server \(S\) to send its authenticator. Note that security against a malicious PKG’s passive/active attacks is a stronger security guarantee than security against an outside adversary’s passive/active attacks. The SPAIBE protocol consists of Initialization and Key Establishment phases.

5.1 Initialization

In this phase, it executes the following three processes Setup, Extract and Registration.

5.1.1 Setup

The Setup on input \(1^d\) outputs public parameters \(pp\) and a master secret key \(msk\) by running the setup algorithm \(Setup_{IBE}\) of Sect. 2.3.1 where \(h\) is another random generator of \(G_1\) and \(H_1 : (0, 1)^* \rightarrow Z_q^*\) and \(H_2, H_3 : (0, 1)^* \rightarrow (0, 1)^k\) are descriptions of additional cryptographic hash functions. It outputs \((pp, msk) = ((pp_{IBE}, h, H_1, H_2, H_3), z)\).

5.1.2 Extract

The key extraction algorithm Extract (run by PKG), on input the public parameters \(pp_{IBE}\), the master secret key \(msk(=z)\) and an identity \(S\), computes \(Q_S = G(S)\) and outputs a private key \(d_S \equiv Q_S^z\) that is securely transmitted to the corresponding server \(S\).

5.1.3 Registration

First, client \(C\) randomly chooses his/her password \(pw\) from a dictionary \(D_{pw}\) and sends \((C, h^{-H_1(pw)})\) to server \(S\). Then, the server stores \((C, h^{-H_1(pw)})\) to a password file. Note that password \(pw\) is kept by client \(C\) secretly, and \((d_S, (C, h^{-H_1(pw)}))\) are held by server \(S\) secretly. This registration process should be done securely between client \(C\) and server \(S\).

5.2 Key Establishment

In this phase, client \(C\) and server \(S\) execute the SPAIBE protocol over insecure networks (e.g., the Internet) in order to share a session key to be used for protecting subsequent communications. This phase of SPAIBE has three steps as below (see also Fig. 1).

Step 1. First, client \(C\) chooses two random elements \(x, r \leftarrow S\) and computes \(X \equiv g^x\). Next, the client runs the encryption algorithm \(Encrypt\) of Sect. 2.3.1 on input \(pp_{IBE}\), an identity \(S\) and a message \(H’(pw)||X\). Then, client \(C\) sends \((C, U_1, U_2)\) to server \(S\).

Step 2. After receiving a message \((C, U_1, U_2)\) from client \(C\), server \(S\) runs the decryption algorithm Decrypt of Sect. 2.3.1 on input \(pp_{IBE}\), \((U_1, U_2)\) and a private key \(d_S\). If the first component of \((U_2 \oplus H(\delta))\) is not equal to \(H’(pw)\), the server aborts the protocol. Otherwise, server \(S\) chooses a random element \(y \leftarrow Z_q^*\), and computes \(Y \equiv g^y\) and \(Z \equiv X^y\) where \(X\) is the second component of \((U_2 \oplus H(\delta))\). Also, the server computes a session key \(SK_C = H_3(C||S||X||Y||Z)\) and then sends \((S, Y)\) to the client.

Step 3. After receiving a message \((S, Y)\) from server \(S\), client \(C\) computes \(Z \equiv Y^s\) and a session key \(SK_C = H_3(C||S||X||Y||Z)\).
A strengthened PAKE (for short, SPAIBE) protocol with IBE

\[ \mathbb{G}_1, \mathbb{G}_2, e, q, g, h, g^x, G, H, H_1, H_2, H_3 \]

\[ \text{Client } C \text{ (pw)} \]

\[ \begin{align*}
& x \leftarrow \mathbb{Z}_q^*, X \equiv g^x \\
& W \equiv X \cdot H^1(\text{pw}) \\
& r \leftarrow \mathbb{Z}_q^*, g_S = e(G(S), g^r) \\
& U_1 = g^r, U_2 = W \oplus H(g_S^r) \\
& \text{Encrypt}(pp_{IBE}, S, W) \text{ of BF-IBE} \\
& \text{sid} = C||S||U_1||U_2||Y \\
& K \equiv Y^r \\
& \text{If } V_S \neq H_2(\text{sid}||X||K), \text{ abort.} \\
& \text{Otherwise, } SK_C = H_3(\text{sid}||X||K')
\end{align*} \]

\[ \text{Server } S \left( d_S, (C, h^{-H_1(\text{pw})}) \right) \]

\[ \begin{align*}
& y \leftarrow \mathbb{Z}_q^*, Y \equiv g^y \\
& \text{Decrypt}(pp_{IBE}, (U_1, U_2), d_S) \text{ of BF-IBE} \\
& \begin{align*}
& \delta = e(d_S, U_1) \\
& W = U_2 \oplus H(\delta) \\
& X' = W \cdot h^{-H_1(\text{pw})}, K' \equiv (X')^y \\
& \text{sid} = C||S||U_1||U_2||Y \\
& V_S = H_2(\text{sid}||X'||K') \\
& SK_S = H_3(\text{sid}||X'||K')
\end{align*} \]

From the above construction, one might think of using only one random element \(x\) or \(r\) instead of two random elements in Step 1 for efficiency improvements. However, such constructions can result in offline dictionary attacks by a malicious PKG who has the master secret key \(z\). For example, if the random element \(r\) is reused for \(K\) (i.e. \(X \equiv g^r\)) the malicious PKG can find out the password \(\text{pw}\) by performing offline dictionary attacks. This is the reason why we use two random elements \(x, r\) where \(x\) is used for the first mask and \(r\) is for the second mask in the SPAIBE protocol.

If an adversary gets the password \(\text{pw}\) (not the master secret key), the SPAIBE protocol is secure against server impersonation attacks thanks to the security of the BF-IBE scheme [15, 18].

6. Security Model

Here, we extend the security model [27, 28] in order to capture attacks of a malicious PKG, and define the semantic security of session keys.

Let \(C\) and \(S\) be sets of clients and servers, respectively. We denote by \(C \in C\) and \(S \in S\) two parties that participate in an authenticated key exchange protocol \(P\). Each of them may have several instances called oracles involved in distinct, possibly concurrent, executions of \(P\). We denote \(C\) (resp., \(S\) instances by \(C^i\) (resp., \(S^j\) where \(i, j \in \mathbb{N}\), or by \(I\) in the case of any instance. During the protocol execution, an adversary has the entire control of networks and has access to the master secret key. Let us show the capability of adversary \(A\) each query captures:

- **Execute** \((C^i, S^j)\): This query models passive attacks, where the adversary gets access to honest executions of \(P\) between the instances \(C^i\) and \(S^j\) by eavesdropping.
- **Send** \((I, msg)\): This query models active attacks by

---

\[^1\text{From an implementation perspective, data type conversion functions for } W \text{ can be used. See Annex of [8].}\]

\[^{11}\text{Such password compromise always allows client impersonation attacks in PAKE protocols.}\]
having $\mathcal{A}$ send a message to instance $I$. The adversary $\mathcal{A}$ gets back the response $I$ generates in processing the message $msg$ according to the protocol $P$. A query $\text{Send}(C', \text{Start})$ initializes the protocol, and thus the adversary receives the first flow message.

- **Reveal($I$):** This query handles misuse of the session key (e.g., use in a weak symmetric-key encryption) by any instance $I$. The query is only available to $\mathcal{A}$, if the instance actually holds a session key, and the latter is released to $\mathcal{A}$.

- **Reveal($\mathcal{PKG}$):** This query handles compromise of the PKG, and the adversary receives the master secret key.

- **Test($I$):** This oracle is used to see whether or not the adversary $\mathcal{A}$ can not distinguish the session key $SK$ from some oracles and then is allowed to invoke any number of query by outputting this guess (e.g., with the $\text{Reveal}(I)$-query).

We say that an instance $I$ is fresh only if an adversary $\mathcal{A}$ can not distinguish the session key $SKB$ in a trivial way (e.g., with the $\text{Reveal}(I)$-query).

The adversary $\mathcal{A}$ is provided with random coin toses, some oracles and then is allowed to invoke any number of queries as described above, in any order. The aim of the adversary is to break the privacy of the session key (a.k.a., semantic security) in the context of executing $P$.

**Definition 5:** (AKE Security) The AKE security is defined by the game $\text{Game}^{\text{ake}}(\mathcal{A}, P)$, in which the ultimate goal of the adversary is to guess the bit $b$ involved in the $\text{Test}$-query by outputting this guess $b'$. We denote the AKE advantage, by $\text{Adv}^{\text{ake}}(\mathcal{A}) = 2 \Pr[b = b'] - 1$, as the probability that $\mathcal{A}$ can correctly guess the value of $b$. The protocol $P$ is said to be $(t, e)$-AKE-secure if $\mathcal{A}$’s advantage is smaller than $e$ for any adversary $\mathcal{A}$ running time $t$.

7. Security Proof of SPAIBE

In this section, we show that the SPAIBE protocol of Fig. 1 is provably secure in the random oracle model [22].

**Theorem 1:** Let $P$ be the SPAIBE protocol of Fig. 1 where passwords are chosen from a dictionary of size $N$. For any adversary $\mathcal{A}$ within a polynomial time $t$, with less than $q_{se}$ active interactions with the parties (Send-queries), $q_{ex}$ passive eavesdroppings (Execute-queries) and asking $q_{hs}$ hash queries to any $H_j$, $\text{Adv}^{\text{ake}}(\mathcal{A}) \leq e$, with $e$ upper-bounded by

$$
\frac{6q_{se}}{N} + 6q_{h}^2 \times \text{Succ}_{D_1}(t_1 + 3\tau_r) + \frac{3(q_{ex} + q_{se})^2}{q} + \frac{2q_{se}}{2k} + 4nq_{se} \times \text{Adv}^{\text{ind-id-CPA}_{IBE}}(B),
$$

where $n = |S|$, $k$ is the output length of $H$, and $\tau_r$ denotes the computational time for an exponentiation in $G_1$.

This theorem indicates that the SPAIBE protocol is secure against offline dictionary attacks since the advantage of the adversary essentially grows with the ratio of interactions to the number of passwords. Also, one can notice that this security result holds with any IBE scheme that is IND-ID-CPA secure.

**Proof 1:** In this proof, we define a sequence of games starting at the real protocol $G_0$ and ending up at $G_6$ where we bound the probability of each event by using Shoup’s difference lemma [29]. For clarity, we denote by $\text{Event}$, an event $\text{Event}$ considered in Game $G_i$.

**Game $G_0$:** This is the real protocol in the random oracle model. We are interested in the following event:

- $S_0$ which occurs if the adversary correctly guesses the bit $b$ involved in the $\text{Test}$-query

$$
\text{Adv}^{\text{ake}}(\mathcal{A}) = 2 \Pr[S_0] - 1.
$$

**Game $G_1$:** In this game, we simulate the hash oracles ($H_j$, but as well additional hash functions, for $j = 1, 2, 3$ which will appear in the game $G_4$) by maintaining hash lists $\Lambda_H$ and $\Lambda_{H'}$ (see below). We also simulate all the instances, as the real parties would do, for the Send-queries and for the Execute, Reveal, and Test-queries (see further below). From this simulation, we can easily see that the game is perfectly indistinguishable from the real attack.

**Simulation of the hash functions: $H_j$ oracles**

- For a hash-query $H_j(q)$ (resp., $H'_j(q)$), such that a record $(j, q, r)$ appears in $\Lambda_H$ (resp., $\Lambda_{H'}$), the answer is $r$. Otherwise, one chooses a random element $r \leftarrow [0, 1]_q$, answers with it, and adds the record $(j, q, r)$ to $\Lambda_H$ (resp., $\Lambda_{H'}$).

**Simulation of the SPAIBE protocol**

Send-queries to $C$

We answer to the Send-queries to a C-instance as follows:

- A $\text{Send}(C', \text{Start})$-query is processed according to the following rules:

  - **Rule C1:**

    Choose a random element $\theta \leftarrow \mathbb{Z}_{p^t}^*$, and compute $X \equiv g^\theta$, $w = H_1(pw)$, $W \equiv X \cdot h^w$ and $(U_1, U_2) = \text{Encrypt}(pw_{IBE}, S, W)$. Then the query is answered with $(C, U_1, U_2)$, and the instance goes to an expecting state.

  - **Rule C2:**

    If the instance $C$ is in an expecting state, a query $\text{Send}(C', (S, Y, V_S))$ is processed by computing the Diffie-Hellman key, authenticator and session key. We apply the following rules.
Compute \( K \equiv Y^g \).

- **Rule C3\(^{(1)}\)**
  Compute the expected authenticator and session key:
  \[
  V'_S = H_2(C||S||U_1||U_2||Y||X||K),
  S K_C = H_3(C||S||U_1||U_2||Y||X||K).
  \]
  If \( V'_S \neq V_S \), it terminates. Otherwise, the instance accepts.

**Send-queries to S**
We answer to the Send-queries to a S-instance as follows:
- A Send\((S^0, (C, U_1, U_2))\)-query is processed according to the following rule:
  - **Rule S1\(^{(1)}\)**
    Choose a random element \( \varphi \triangleq S \ast \mathbb{Z}_q^* \) and compute \( Y \equiv g^\varphi \).
    Then, the instance computes the Diffie-Hellman key and authenticator. We apply the following rules:
  - **Rule S2\(^{(1)}\)**
    Compute \( w = H_1(pw), W = \text{Decrypt}(pp_{IBE}, (U_1, U_2), d_S), X' \equiv W \cdot h^w \) and \( K' \equiv (X')^g \).
  - **Rule S3\(^{(1)}\)**
    Compute the authenticator and session key:
    \[
    S K_S = H_2(C||S||U_1||U_2||Y||X'||K'),
    S K_S = H_3(C||S||U_1||U_2||Y||X'||K').
    \]
    The query is answered with \((S, Y, V_S)\) and the instance accepts.

**Other queries**
- An Execute\((C', S^0)\)-query is processed using successively the above simulations of the Send-queries: \((C, U_1, U_2) \leftarrow \text{Send}(C', \text{Start}), (S, Y, V_S) \leftarrow \text{Send}(S^0, (C, U_1, U_2))\), and then outputting the transcript \(((C, U_1, U_2), (S, Y, V_S))\).
- A Reveal\((I)\)-query returns the session key \( SK \) computed by the instance \( I \) (if the former has accepted).
- A Reveal\((PKG)\)-query returns the master secret key.
- A Test\((I)\)-query first gets \( SK \) from Reveal\((I)\), and flip a coin \( b \). If \( b = 1 \), we return the value of the session key \( SK \), otherwise we return a random value drawn from \([0, 1]^k\).

**Game G2:** For an easier analysis in the following, we cancel games in which some collisions (Coll\(_2\)) are unlikely to happen:
- Collisions on the partial transcripts \(((C, U_1, U_2), (S, Y, V_S))\): Any adversary tries to find out one pair \((U_1, Y)\), coinciding with the challenge transcript, and then obtain the corresponding session key using the Reveal-query. However, at least one party involves with the transcripts, and thus one of \( U_1 \) and \( Y \) is truly uniformly distributed.

The probability is bounded by the birthday paradox:
\[
\Pr[\text{Coll}_2] \leq \frac{(q_{\text{ex}} + q_{\text{se}})^2}{2q}.
\] (4)

**Game G3:** Here, we consider the following
\[
\Pr[S_3] = \Pr[S_3] \triangleq \text{BreakIBE} \lor \text{BreakIBE}
= \Pr[S_3] \triangleq \text{BreakIBE} \lor \Pr[S_3] \triangleq \text{BreakIBE}
\]
where BreakIBE is an event that an adversary gets the master secret key by asking the \( \text{Reveal}(PKG) \)-query, and its complement is denoted by \( \neg \text{BreakIBE} \). For the first term, we construct an adversary \( B \) who breaks the IND-ID-CPA security of the IBE scheme IBE by using an adversary \( A \) who breaks the semantic security of the SPAIBE protocol. Let \( n = |S| \) and \( S \) be the challenge identity. \( B \) chooses two random elements \( \theta_0, \theta_1 \triangleq S \ast \mathbb{Z}_q^* \) and composes \( W_0 \equiv g^{\theta_0} \cdot h^w \) and \( W_1 \equiv g^{\theta_1} \cdot h^w \), and forwards \((W_0, W_1, S)\) to the IBE challenger. The challenge Encrypt\((pp_{IBE}, S, W_j)\) is introduced in the simulation of the Send\((C', \text{Start})\)-query. Finally, the output of \( B \) is the same as the output of \( A \), and the probability of \( \Pr[S_3] \triangleq \text{BreakIBE} \) is bounded by
\[
\Pr[S_3] \triangleq \text{BreakIBE} \leq 2n^{q_{\text{se}}} \times \text{Adv}_{\text{IBE}}^{\text{ind-id-cpa}}(B).
\] (5)
Note that all the bounds in the subsequent games are conditioned on the BreakIBE.

**Game G4:** In order to make the authenticator and session key unpredictable to any adversary, we compute them using the private oracles \( H'_j \) (instead of \( H_j \)) so that the values are completely independent from the random oracles. We reach this aim by using the following rules:
- **Rule C3\(^{(4)}\)**
  Compute the authenticator and session key:
  \[
  V'_S = H_2(C||S||U_1||U_2||Y),
  S K_C = H_3(C||S||U_1||U_2||Y).
  \]
- **Rule S3\(^{(4)}\)**
  Compute the authenticator and session key:
  \[
  V_S = H_2(C||S||U_1||U_2||Y),
  S K_S = H_3(C||S||U_1||U_2||Y).
  \]
Since we do no longer need to compute the value \( K \), we can simplify the following rules:
- **Rule C2\/^{\text{S2}}\(^{(4)}\)**
  Do nothing.
Finally, the mask \( h^w \) is not used anymore either so that we can also simplify the generation of \( W \) using the group property of \( G_1 \).
- **Rule C1\(^{(4)}\)**
  Choose a random element \( x \triangleq S \ast \mathbb{Z}_q^* \) and compute \( W \equiv g^x \).
The games \( G_4 \) and \( G_3 \) are indistinguishable unless some specific hash queries are asked, denoted by event AskH\(_4\) = AskH\(_2\_4\) \lor AskH\(_3\_2\_4\):
- **AskH\(_4\):** \( H_2(C||S||U_1||U_2||Y||X||K) \) has been
queried by $\mathcal{A}$ to $H_2$ for some execution transcripts $((C, U_1, U_2), (S, Y))$;
- $\text{AskH3w2}_4$: $H_3(C||S||U_1||U_2||Y||X||K)$ has been queried by $\mathcal{A}$ to $H_2$ for some execution transcripts $((C, U_1, U_2), (S, Y))$, but event $\text{AskH2}_4$ did not happen.

The authenticator is computed with random oracles that are private to the simulator, then one can remark that it cannot be guessed by the adversary, better than at random for each attempt, unless the same partial transcript $((C, U_1, U_2), (S, Y))$ appeared in another session with a real instance $(C', S')$. But such a case has already been excluded (in game $G_2$). A similar remark holds on the session key:

$$\Pr[S_\lambda] \leq \frac{q_{se}}{2^k} + \frac{1}{2}.$$  \hfill (6)

When collisions of the partial transcripts have been excluded, the event $\text{AskH2}$ can be split in three disjoint sub-cases:

- $\text{AskH2-Passive}_4$: The transcript $((C, U_1, U_2), (S, Y))$ comes from an execution between instances of $C$ and $S$ (Execute-queries or forward of Send-queries, relay of part of them). This means that both $(U_1, U_2)$ and $Y$ have been simulated;
- $\text{AskH2-WithC}_4$: The execution involved an instance of $C$, but $Y$ has not been sent by any instance of $S$. This means that $(U_1, U_2)$ has been simulated, but $Y$ has been produced by the adversary;
- $\text{AskH2-WithS}_4$: The execution involved an instance of $S$, but $(U_1, U_2)$ has not been sent by any instance of $C$. This means that $Y$ has been simulated, but $(U_1, U_2)$ has been produced by the adversary.

In the above, an adversary can get the value $W$ from the simulated $(U_1, U_2)$ since the adversary has the master secret key with the $\text{Reveal}(PKG)$-query.

**Game $G_3$:** In order to evaluate the above events, we introduce a random Diffie-Hellman instance $(P, Q)$ where both $P$ and $Q$ are generators of $\mathbb{G}_1$. (Otherwise, the Diffie-Hellman problem is easy.) We first set the element $Q$ as follows: $h = Q$.

By the isomorphic property from $\mathbb{G}_1$ to $\mathbb{G}_1$, it is perfectly indistinguishable from before since there exists a unique discrete logarithm for $Q$. Also, we introduce the other part $P$ of the Diffie-Hellman instance in the simulation of $S$.  

- **Rule $S_1^{(5)}$**

  Choose a random element $y \xleftarrow{\$} \mathbb{Z}_q^*$ and compute $Y = P^y$.

Note that we excluded the case $Y \equiv 1$. By Shoup’s difference lemma, the probability is upper-bounded by a random guess for $Y$:

$$|\Pr[\text{AskH}_3] - \Pr[\text{AskH}_4]| \leq \frac{q_{se} + q_{ex}}{q}.$$ \hfill (7)

**Game $G_4$:** It is now possible to evaluate the probability of the event $\text{AskH}$ (or more precisely, the sub-cases). Indeed, one can see that the password is never used during the simulation. It does not need to be chosen in advance, but at the very end only. Then, an information-theoretic analysis can be done which simply uses cardinalities of some sets.

To this aim, we first cancel a few more games, wherein for some pairs $(W, Y) \in \mathbb{G}_1^2$, involved in a communication between an instance $S'$ and either the adversary or an instance $C'$, there are two distinct elements $h^{w}$ such that the tuple $(U_1, U_2, Y, W/h^{w}, \text{CDH}_{\lambda_1}(W/h^{w}, Y))$ is in $\Lambda_4$, and one can solve the computational Diffie-Hellman problem:

$$\Pr[\text{CollH}_8] \leq \frac{q_{s}^2}{|\mathbb{G}_1|} \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + \tau_e).$$ \hfill (8)

The event $\text{CollH}_8$ can be upper-bounded, granted the following lemma:

**Lemma 1:** If for any pair $(W, Y) \in \mathbb{G}_1^2$, involved in a communication with an instance $S'$, there are two elements $h^{w_0}$ and $h^{w_1}$ such that $(U_1, U_2, Y, W/h^{w_0}, Z_i = \text{CDH}_{\lambda_1}(W/h^{w_0}, Y))$ is in $\Lambda_4$, one can solve the computational Diffie-Hellman problem:

$$\Pr[\text{CollH}_8] \leq \frac{q_{s}^2}{|\mathbb{G}_1|} \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + \tau_e).$$ \hfill (9)

**Proof.** We prove this lemma by showing the reduction to the CDH problem when event $\text{CollH}_8$ happens. We assume that there exist $(W, Y \equiv P^y) \in \mathbb{G}_1^2$ involved in a communication with an instance $S'$, and two elements $h^{w_0} \equiv Q^{w_0}$ and $h^{w_1} \equiv Q^{w_1}$ such that the tuple $(U_1, U_2, Y, W/h^{w_0}, Z_i = \text{CDH}_{\lambda_1}(W/h^{w_0}, Y))$ is in $\Lambda_4$, for $i = 0, 1$. Then,

$$Z_i = \text{CDH}_{\lambda_1}(W/h^{w_i}, Y)$$

$$= \text{CDH}_{\lambda_1}(W \times Q^{-w_i}, Y)$$

$$= \text{CDH}_{\lambda_1}(W, Y) \times \text{CDH}_{\lambda_1}(Q, Y)^{-w_i}$$

$$= \text{CDH}_{\lambda_1}(W, Y) \times \text{CDH}_{\lambda_1}(P, Q)^{-w_i}.$$ \hfill (10)

As a consequence, $Z_1/Z_0 = \text{CDH}_{\lambda_1}(P, Q)^{(w_0-w_1)}$, and thus $\text{CDH}_{\lambda_1}(P, Q) = (Z_1/Z_0)^{y}$ where $\psi$ is the inverse of $y(w_0 - w_1)$ in $\mathbb{Z}_q^*$. The latter exists since $h^{w_0} \neq h^{w_1}$ and $y \neq 0$. By guessing the two queries asked to the $H_j$, one concludes the proof. \hfill \qed

In order to conclude the proof, let us study separately the three sub-cases of $\text{AskH}_2$, and then $\text{AskH}_3w2$ (keeping in mind the absence of several kinds of collisions for partial transcripts, and for $h^{w}$ in $H$-queries):

- $\text{AskH2-Passive}$: About the passive transcripts (in which both $W$ and $Y$ have been simulated), one
can state the following lemma:

**Lemma 2:** If for any pair \((W, Y) \in \mathbb{G}_1^2\), involved in a passive transcript, there is an element \(h^w\) such that \((U_1, U_2, Y, W/h^w, Z = \text{CDH}_{\mathbb{G}_1}(W/h^w, Y))\) is in \(\Lambda_H\), one can solve the computational Diffie-Hellman problem:

\[
\Pr[\text{AskH2-Passive}_\theta] \leq q_h \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + 2\tau_e).
\]

(11)

**Proof.** We prove this lemma by showing the reduction to the CDH problem when event \(\text{AskH2-Passive}_\theta\) happens. We assume that there exist \((W \equiv g^y, Y \equiv \mathbb{P}^w) \in \mathbb{G}_1^2\) involved in a passive transcript and \(h^w \equiv Q^w\) such that the tuple \((U_1, U_2, Y, W/h^w, Z)\) is in \(\Lambda_H\). As above,

\[
Z = \text{CDH}_{\mathbb{G}_1}(W, Y) \times \text{CDH}_{\mathbb{G}_1}(Q, Y)^{-w} = \mathbb{P}^w \times \text{CDH}_{\mathbb{G}_1}(P, Q)^{-w}.
\]

(12)

As a consequence, \(\text{CDH}_{\mathbb{G}_1}(P, Q) = (Z/\mathbb{P}^w)^{\bar{w}}\) where \(\bar{w}\) is the inverse of \(-yw\) in \(\mathbb{Z}_q^*\). The latter exists since we have excluded the cases where \(y = 0\) or \(w = 0\). By guessing the query asked to the \(H_p\), one concludes the proof.

\(\square\)

- **AskH2-WithC:** This corresponds to an attack where the adversary tries to impersonate \(S\) to \(C\). But each authenticator sent by the adversary has been computed with at most one \(h^w\) value:

\[
\Pr[\text{AskH2-WithC}_\theta] \leq \frac{q_{se}}{N}.
\]

(13)

- **AskH2-WithS:** The Lemma 1, applied to games where the event \(\text{CollH}_0\) did not happen, states that for each pair \((W, Y)\) involved in a transcript with an instance \(S\), there is at most one element \(h^w\) such that \(h^w \equiv Q^w\), the corresponding tuple is in \(\Lambda_H\): The probability for the adversary over a random password is as above:

\[
\Pr[\text{AskH2-WithS}_\theta] \leq \frac{q_{se}}{N}.
\]

(14)

About AskH3w2 (when the above three events did not happen), it means that only executions with an instance of \(S\) (and either \(C\) or the adversary) may lead to acceptance. Exactly the same analysis as for \(\text{AskH2-Passive}\) and \(\text{AskH2-WithS}\) leads to:

\[
\Pr[\text{AskH3w2}_\theta] \leq \frac{q_{se}}{N} + q_h \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + 2\tau_e).
\]

(15)

As a conclusion, we get an upper-bound for the probability of \(\text{AskH}_0\) by combining all the cases:

\[
\Pr[\text{AskH}_\theta] \leq \frac{3q_{se}}{N} + 2q_h \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + 2\tau_e).
\]

(16)

Combining inequalities (4), (5), (6), (7), (9) and (16), one gets

\[
\Pr[S_0] \leq \frac{q_{se}}{2q} + \frac{1}{2} + \Delta,
\]

(17)

where

\[
\Delta \leq \frac{3q_{se}}{N} + 2q_h \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + 2\tau_e) + \frac{q_{se} + q_{ex}}{q} + q_h^2 \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + \tau_e) + \frac{(q_{ex} + q_{se})^2}{2q} + 2q_{se} \times \text{Adv}_{\text{IBE}}^{\text{ind-id-cpa}}(B)
\]

\[
\leq 3\frac{q_{se}}{N} + 3q_{se}^2 \times \text{Succ}_{\mathbb{G}_1}^{\text{cdh}}(t_1 + 3\tau_e) + \frac{3(q_{ex} + q_{se})^2}{2q} + 2q_{se} \times \text{Adv}_{\text{IBE}}^{\text{ind-id-cpa}}(B).
\]

(18)

Finally, one can get the result as desired by noting that \(\text{Adv}_{\mathcal{P}}^{\text{atk}}(\mathcal{A}) = 2\Pr[S_0] - 1\).

8. Comparison

In this section, we compare the PAKE protocols using the BF-IBE scheme [15], [18] (PAKE-CS [17], iPake [2], UKAM-PiE [21], SPAIBE of Sect. 5) in terms of efficiency, number of passes and security against a malicious PKG.

We summarize a comparative result in Table 1 where

| Protocols     | Computation costs                  | Communication costs | # of passes | Security against a malicious PKG |
|--------------|-----------------------------------|---------------------|-------------|---------------------------------|
| PAKE-CS [17] | 1Pairing + 5Exp, l_1 + 1Exp, l_2 | ||C|| + ||S|| + 4||G_1|| + ||H|| | 2           | No                              |
| iPake [2]    | 1Pairing + 2Exp, l_1 + 1Exp, l_2 | ||C|| + ||S|| + 2||G_1|| + ||H|| | 2           | No                              |
| UKAM-PiE [21]| 1Pairing + 3Exp, l_1 + 1Exp, l_2 | ||C|| + ||S|| + 2||G_1|| + ||H|| | 2           | No                              |
| SPAIBE (Sect. 5) | 1Pairing + 3.17Exp, l_1 + 1Exp, l_2 | ||C|| + ||S|| + 2||G_1|| + 2||H|| | 2           | Yes                             |
Exp$_{G_1}$ (resp., Exp$_{G_2}$) indicates a modular exponentiation in $G_1$ (resp., $G_2$), and Pairing is a pairing operation. Let $|l|$ be a bit-length of $l$. The number of modular exponentiations for one simultaneous calculation of two bases (i.e. $g_1^{x_1} \cdot g_2^{x_2}$) is counted to 1.17 due to the Strauss’s algorithm [30], [31] (also known as Shamir’s trick).

In the SPAIBE protocol, the computation cost of client C (resp., server S) is reduced to 2Exp$_{G_1}$ (resp., 1Pairing + 1Exp$_{G_1}$) if pre-computation is allowed. Though the iPAKE protocol [2] is more efficient than the SPAIBE protocol with respect to both computation costs of client C and communication costs, it is not secure against passive/active attacks by a malicious PKG (as in Sect. 4). Compared to the UKAM-PiE protocol [21], the SPAIBE protocol has almost same efficiency while providing security against passive/active attacks by a malicious PKG.

9. Conclusions

In this paper, we have revisited the iPAKE protocol [2] using the Boneh-Franklin IBE scheme [15], [18], and the UKAM-PiE protocol in ISO/IEC 11770-4/AMD 1 [21]. That is, the iPAKE and UKAM-PiE protocols are insecure against passive/active attacks by a malicious PKG where the malicious PKG can find out all clients’ passwords by just eavesdropping on the communications, and the PKG can share a session key with any client by impersonating the server. Then, we have proposed a strengthened PAKE (SPAIBE) protocol with IBE, which provides security against passive/active attacks by a malicious PKG. Also, we have formally proved the security of the SPAIBE protocol in the random oracle model [22] and compared the PAKE protocols using the BF-IBE scheme (PAKE-CS [17], iPAKE [2], UKAM-PiE [21], SPAIBE) in terms of efficiency, number of passes and security against a malicious PKG.

Acknowledgments

We sincerely appreciate the anonymous reviewers’ constructive and valuable comments on this paper.

References

[1] S.H. Shin, “Strengthened PAKE Protocols Secure Against Malicious Private Key Generator,” WISA 2019, vol.11897, pp.192–205, Springer, 2019.
[2] K.Y. Choi, J. Cho, J.Y. Hwang, and T. Kwon, “Constructing Efficient PAKE Protocols from Identity-Based KEM/DEM,” WISA 2015, vol.9503, pp.411–422, Springer, 2015.
[3] S.M. Bellovin and M. Merritt, “Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks,” IEEE Computer Society Symposium on Research in Security and Privacy, pp.72–84, IEEE, 1992.
[4] S.M. Bellovin and M. Merritt, “Augmented Encrypted Key Exchange: a Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise,” CCS’93, pp.244–250, ACM, 1993.
[5] “Research Papers on Password-based Cryptography.” http://www.jablon.org/passwordlinks.html
[6] F. Hao and P.C. van Oorschot, “SoK: Password-Authenticated Key Exchange – Theory, Practice, Standardization and Real-World Lessons.” https://eprint.iacr.org/2021/1492, 2021. Cryptology ePrint Archive: Report 2021/1492.
[7] IEEE 1363.2, “IEEE Standard Specification for Password-Based Public-Key Cryptographic Techniques.” https://ieeexplore.ieee.org/document/4773330, Jan. 2009.
[8] ISO/IEC 11770-4:2017, “Information technology — Security techniques — Key management — Part 4: Mechanisms based on weak secrets.” https://www.iso.org/standard/67933.html, Nov. 2017.
[9] IETF RFC 2945, “The SRP Authentication and Key Exchange System.” https://www.rfc-editor.org/rfc/rfc2945.html, Sept. 2000.
[10] ITU-T Recommendation X.1035, “Password-authenticated key exchange (PAK) protocol.” https://www.itu.int/rec/T-REC-X.1035-200702-I/en, Feb. 2007.
[11] L. Gong, M.A. Lomas, R.M. Needham, and J.H. Saltzer, “Protecting Poorly Chosen Secrets from Guessing Attacks,” IEEE Journal on Selected Areas in Communications, vol.11, no.5, pp.648–656, 1993.
[12] S. Halevi and K. Krawczyk, “Public-Key Cryptography and Password Protocols,” ACM Transactions on Information and System Security, vol.2, no.3, pp.230–268, 1999.
[13] A. Shamir, “Identity-Based Cryptosystems and Signature Schemes,” CRYPTO’84, pp.47–53, Springer, 1984.
[14] R. Sakai and M. Kasahara, “Cryptosystems Based on Pairings,” Symposium on Cryptography and Information Security, 2000.
[15] D. Boneh and M. Franklin, “Identity-Based Encryption from the Weil Pairing,” CRYPTO 2001, vol.2139, pp.213–229, Springer, 2001.
[16] C. Cocks, “An Identity Based Encryption Scheme Based on Quadratic Residues,” Cryptography and Coding 2001, vol.2260, pp.360–363, Springer, 2001.
[17] X. Yi, R. Tso, and E. Okamoto, “Identity-based Password-Authenticated Key Exchange for Client/Server Model,” SECRYPT 2012, pp.45–54, SCITEPRESS, 2012.
[18] D. Boneh and M. Franklin, “Identity-Based Encryption from the Weil Pairing,” SIAM Journal on Computing, vol.32, no.3, pp.586–615, 2003.
[19] X. Boyen, “A Tapestry of Identity-Based Encryption: Practical Frameworks Compared,” International Journal of Applied Cryptography, vol.1, no.1, pp.3–21, 2008.
[20] J.Y. Hwang, S.-H. Kim, D. Choi, S.-H. Jin, and B. Song, “Robust Authenticated Key Exchange Using Passwords and Identity-Based Signatures,” International Conference on Research in Security Standardisation (SSR) 2015, vol.9497, pp.43–69, Springer, 2015.
[21] ISO/IEC 11770-4:2017/AMD 1:2019, “Information technology — Security techniques — Key management — Part 4: Mechanisms based on weak secrets — Amendment 1: Unbalanced Password-Authenticated Key Agreement with Identity-Based Cryptosystems (UAPA/IBC),” https://www.iso.org/standard/73902.html, Sept. 2019.
[22] M. Bellare and P. Rogaway, “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols,” CCS’93, pp.62–73, ACM, 1993.
[23] D. Boneh and X. Boyen, “Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles,” EUROCRYPT 2004, vol.3027, pp.223–238, Springer, 2004.
[24] R. Canetti, S. Halevi, and J. Katz, “Chosen-Ciphertext Security from Identity-Based Encryption,” EUROCRYPT 2004, vol.3027, pp.207–222, Springer, 2004.
[25] B. Waters, “Efficient Identity-Based Encryption Without Random Oracles,” EUROCRYPT 2005, vol.3494, pp.114–127, Springer, 2005.
[26] C. Gentry, “Practical Identity-Based Encryption Without Random Oracles,” EUROCRYPT 2006, vol.4004, pp.445–464, Springer, 2006.
[27] M. Bellare, D. Pointcheval, and P. Rogaway, “Authenticated Key Exchange Secure against Dictionary Attacks,” EUROCRYPT 2000, vol.1807, pp.139–155, Springer, 2000.
SeongHan Shin  received his B.S. and M.S. degrees in computer science from Pukyong National University, Busan, Korea, in 2000 and 2002, respectively. In 2005, he received his Ph.D. degree in information and communication engineering, information science and technology from The University of Tokyo, Tokyo, Japan. In 2006, he joined the National Institute of Industrial Science and Technology (AIST), Japan. Currently, he is a senior research scientist at the Cyber Physical Security Research Center (CPSEC), AIST. He received the CSS 2003 Student Paper Award, IWS2005/WPMC’05 Best Student Paper Award, and WISA 2019 Best Paper Premium Award. His research interests include information security, cryptography, and wireless security.