Computer Security Symposium 2017
23 - 25 October 2017

Hybridパスワード認証プロトコルにおけるKGCについて

辛 星漢[1,a)]　古原 和邦[1]

概要：本稿では Hybrid パスワード認証プロトコルにおける KGC について議論する.

キーワード：パスワード, 認証, 鍵生成センター

Reducing the Power of Key Generation Center in Hybrid Password Authentication Protocol

SeongHan Shin[1,a)]　Kazukuni Kobara[1]

Abstract: In this paper, we discuss how to reduce the power of KGC in a hybrid password authentication protocol.

Keywords: Password, authentication, KGC

1. Introduction

The Password-Authenticated Key Exchange (PAKE) protocols provide password-only authentication and establishment of temporal session keys to be used for subsequent cryptographic algorithms. These protocols are designed to be secure against passive/active attacks as well as off-line dictionary attacks on human-memorable passwords, shared by the participating parties. For a long time, PAKE protocols have received much attention because password authentication is commonly used and widely deployed in practice. Since the appearance of Encrypted Key Exchange (EKE) [6], [7], a number of PAKE protocols (see [8], [14] and references therein) have been proposed in the literature. And, some PAKE protocols have been standardized in IEEE 1363.2 [11], ISO/IEC 11770-4 [12], IETF [16] and ITU-T [13].

In general, PAKE protocols can be classified into 'balanced' PAKE and 'augmented' PAKE [11], [12]: in the former case a client and a server share a common password; and in the latter case a client remembers his/her password and a server has password verification data (derived by applying a one-way function to the password). Since password verification data has the same entropy of the password, the off-line dictionary attacks are inevitable if server is compromised in the augmented PAKE protocols. Nonetheless, an augmented PAKE protocol may be preferable because it provides extra protection for server compromise. That is, the ultimate goal in improving resistance to server compromise is to make the off-line dictionary attacks the best one an attacker can do. Actually, there has been a significant amount of works (e.g., [9], [10]) on making PAKE protocols secure even in the case of server compromise.

Though human-memorable passwords (e.g., 4-digit pincode or alphanumerical passwords) are commonly used and very convenient for clients, there exist two major attacks on passwords: on-line and off-line dictionary attacks. Let us take for example a simple challenge-response

1 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)
a) seonghan.shin@aist.go.jp

password authentication protocol [15] where a client and a server share a password pw. In the protocol, the server sends a challenge c to the client, who computes a response $r = \mathsf{H}(c, pw)$ and sends back r to the server where H is a one-way hash function. After receiving r, the server authenticates the client if the received r is equal to its own computation $\mathsf{H}(c, pw)$. The on-line dictionary attacks are performed by an adversary who impersonates one party (i.e., the client in the above example) so that the adversary can sieve out possible password candidates one by one. On the other hand, the off-line dictionary attacks are performed off-line and in parallel where an adversary exhaustively enumerates all possible password candidates, in an attempt to determine the correct one. In the above example, an adversary can find out the correct password pw with off-line dictionary attacks by trying all password candidates pw' until it satisfies $r = \mathsf{H}(c, pw')$. While on-line attacks are applicable to all of the password-based protocols equally, they can be easily prevented by having a server take appropriate countermeasures (e.g., lock-up accounts for 10 minutes after 3 consecutive failures of passwords). But, we cannot avoid off-line attacks by such countermeasures mainly because these attacks can be done off-line and independently of the party.

Recently, Hwang et al., [5] proposed authenticated key exchange (AKE) protocols from a combination of identity-based signature (IBS) and a password-based authentication. Their protocols provide another layer of security meaning that it is not possible for an attacker who gets a password to impersonate the server.

1.1 Our Contributions

In this paper, we focus on the simplified PWIBS-AKE protocol [5] that is based on their modified Galindo-Garcia IBS scheme. After describing the simplified PWIBS-AKE (Sim-PWIBS) protocol [5], we show that it is insecure against active attacks where a malicious KGC (Key Generation Center) can impersonate the server and find out all clients' passwords with off-line dictionary attacks. Then, we propose two secure simplified PWIBS-AKE (PAKEwIBS1 and PAKEwIBS2) protocols both of which provide security against server impersonation and off-line dictionary attacks by a malicious KGC. There is a trade-off between the PAKEwIBS1 and PAKEwIBS2 protocols in terms of computation and communication costs.

2. Preliminaries

In this section, we explain some notations, computational assumptions and an identity-based signature scheme that is secure against existential forgery on adaptively chosen identity and message attacks.

2.1 Notations

Let $k \in \mathbb{N}$ be the security parameter. If U is a set, then $u \overset{\$}{\leftarrow} U$ 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 D be a dictionary size of passwords. Let C and S be the identities of client and server, respectively, with each $\mathsf{id} \in \{0,1\}^*$.

Let $\lambda \in \mathbb{N}$ be the security parameter. And, let Gen be the group generation algorithm that takes as input the security parameter 1^λ and outputs a group description $(\mathbb{G}, q, g)$ where $\mathbb{G}$ is a finite cyclic group of prime order q with g as a generator and its operation is denoted multiplicatively. In the aftermath, all the subsequent arithmetic operations are performed in modulo p unless otherwise stated.

2.2 Computational Assumptions

Here, we explain the discrete logarithm (DL) and computational Diffie-Hellman (CDH) problems.

Definition2.1 (DL Problem) A (t_1, ε_1)-$\mathsf{DL}_{\mathbb{G},q,g}$ attacker is a probabilistic polynomial time (PPT) machine $\mathcal{B}$, running in time t_1, such that its success probability $\mathsf{Succ}^{\mathsf{dl}}_{\mathbb{G},q,g}(\mathcal{B})$, given a random element g^α to output α, is greater than ε_1. We denote by $\mathsf{Succ}^{\mathsf{dl}}_{\mathbb{G},q,g}(t_1)$ the maximal success probability over every adversaries, running within time t_1. The DL problem states that $\mathsf{Succ}^{\mathsf{dl}}_{\mathbb{G},q,g}(t_1) \leq \varepsilon_1$ for any t_1/ε_1 not too large.

Definition2.2 (CDH Problem) A (t_2, ε_2)-$\mathsf{CDH}_{\mathbb{G},q,g}$ attacker is a probabilistic polynomial time (PPT) machine $\mathcal{B}$, running in time t_2, such that its success probability $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},q,g}(\mathcal{B})$, given random elements g^α and g^β to output $g^{\alpha\beta}$, is greater than ε_2. We denote by $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},q,g}(t_2)$ the maximal success probability over every adversaries, running within time t_2. The CDH problem states that $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},q,g}(t_2) \leq \varepsilon_2$ for any t_2/ε_2 not too large.

2.3 An Identity-Based Signature Scheme

In this subsection, we define the syntax of identity-based signature and its security notion (i.e., security against existential forgery on adaptively chosen identity and message attacks).

Definition2.3 (Identity-Based Signature) An identity-based signature (IBS) scheme is a quadruple of probabilistic polynomial-time algorithms $(\mathsf{Setup}_{\mathsf{IBS}}, \mathsf{Extract}, \mathsf{Sign}, \mathsf{Verify})$ such that:

- The setup algorithm $\mathsf{Setup}_{\mathsf{IBS}}$ takes as input the security parameter 1^λ and outputs public parameters pp_{IBS} and a master secret key msk where $(\mathsf{mpk}, \mathsf{msk})$ is a pair of master public/secret keys and mpk is included in pp_{IBS}.
- The key extraction algorithm $\mathsf{Extract}$ takes as input the public parameters pp_{IBS}, the master secret key msk and an identity id, and outputs a private key $\mathsf{sk}_{\mathsf{id}}$ corresponding to the user with this identity.
- The signing algorithm Sign takes as input the public parameters pp_{IBS}, a private key $\mathsf{sk}_{\mathsf{id}}$ and a message m, and outputs a signature σ.
- The deterministic signature verification algorithm Verify takes as input the public parameters pp_{IBS}, an identity id, a message m and a signature σ, and outputs $\{\mathsf{accept}, \mathsf{reject}\}$ indicating whether or not σ is a valid signature of m relative to $(pp_{\mathsf{IBS}}, \mathsf{id})$.

It is required that $\mathsf{Verify}(pp_{\mathsf{IBS}}, \mathsf{id}, m, \sigma) = \mathsf{accept}$ for all $\lambda \in \mathbb{N}$, $\mathsf{id} \in \{0,1\}^*$ and $m \in \{0,1\}^*$.

Definition2.4 (EUF-ID-CMA) An identity-based signature scheme $\Sigma = (\mathsf{Setup}_{\mathsf{IBS}}, \mathsf{Extract}, \mathsf{Sign}, \mathsf{Verify})$ is secure against existential forgery on adaptively chosen identity and message attacks (EUF-ID-CMA) if for a probabilistic polynomial time adversary $\mathcal{B}$ there exists a negligible function $\varepsilon(\cdot)$ in the security parameter λ such that

$$\Pr[IBS^{\mathsf{euf\text{-}id\text{-}cma}}_{\Sigma}(\mathcal{B}) = 1] \le \varepsilon(\cdot) \qquad (1)$$

in the experiment $IBS^{\mathsf{euf\text{-}id\text{-}cma}}_{\Sigma}(\mathcal{B})$ defined as below:

(1) $\mathsf{Setup}_{\mathsf{IBS}}(1^\lambda)$ outputs $(pp_{\mathsf{IBS}}, \mathsf{msk})$.

(2) Adversary $\mathcal{B}$ is given pp_{IBS}. During this experiment, $\mathcal{B}$ has access to two oracles: a key extraction oracle $\mathcal{O}_{\mathsf{Extract}}$ that takes as input an identity id and returns $\mathsf{sk}_{\mathsf{id}} = \mathsf{Extract}(pp_{\mathsf{IBS}}, \mathsf{msk}, \mathsf{id})$; and a signing oracle $\mathcal{O}_{\mathsf{Sign}}$ that takes as input an identity id and a message m, and returns a signature $\sigma = \mathsf{Sign}(pp_{\mathsf{IBS}}, \mathsf{sk}_{\mathsf{id}}, m)$.

(3) $\mathcal{B}$ outputs $(\mathsf{id}^\star, m^\star, \sigma^\star)$.

(4) The output of the experiment is defined to be 1 if $\mathsf{Verify}(pp_{\mathsf{IBS}}, \mathsf{id}^\star, m^\star, \sigma^\star) = \mathsf{accept}$, and 0 otherwise.

In order to prevent trivial attacks, some restrictions apply as follows: $\mathsf{id}^\star$ and $(\mathsf{id}^\star, m^\star)$ should not be equal to any query made to the oracles $\mathcal{O}_{\mathsf{Extract}}(\cdot)$ and $\mathcal{O}_{\mathsf{Sign}}(\cdot,\cdot)$, respectively, and the same id cannot be queried to $\mathcal{O}_{\mathsf{Extract}}(\cdot)$ twice (see [1] for more details). We denote by $\mathsf{Adv}^{\mathsf{euf\text{-}id\text{-}cma}}_{\Sigma}(\mathcal{B}) = \Pr[IBS^{\mathsf{euf\text{-}id\text{-}cma}}_{\Sigma}(\mathcal{B}) = 1]$ the adversary's advantage in attacking the identity-based signature scheme Σ.

2.3.1 Galindo-Garcia IBS (GG-IBS)

Here, we describe Galindo-Garcia IBS (GG-IBS) scheme [4] that is proven to be EUF-ID-CMA secure in the random oracle model [2] under the DL problem.

- The setup algorithm $\mathsf{Setup}_{\mathsf{IBS}}$ on input 1^λ outputs public parameters pp_{IBS} and a master secret key msk where $(\mathbb{G}, q, g)$ is generated by calling the group generation algorithm Gen on input 1^λ, and $G : \{0,1\}^* \to \mathbb{Z}_q^\star$ and $H : \{0,1\}^* \to \mathbb{Z}_q^\star$ are descriptions of hash functions. Let z be a random element from $\mathbb{Z}_q^\star$ and set $(\mathsf{mpk}, \mathsf{msk}) = (g^z, z)$. It outputs $(pp_{\mathsf{IBS}}, \mathsf{msk}) = ((\mathbb{G}, q, g, g^z, G, H), z)$.
- The key extraction algorithm $\mathsf{Extract}$, on input the public parameters pp_{IBS}, the master secret key $\mathsf{msk}(= z)$ and an identity id, chooses $r \xleftarrow{\$} \mathbb{Z}_q^\star$ and computes $w \equiv r + z \cdot H(g^r, \mathsf{id}) \bmod q$. Then, it outputs a private key $\mathsf{sk}_{\mathsf{id}} = (w, g^r)$. Note that g^r is actually public information even though it is part of the private key.
- The signing algorithm Sign, on input the public parameters pp_{IBS}, a private key $\mathsf{sk}_{\mathsf{id}} = (w, g^r)$ and a message m, proceeds as follows. It chooses $a \xleftarrow{\$} \mathbb{Z}_q^\star$ and computes $b \equiv a + w \cdot G(\mathsf{id}, g^a, m) \bmod q$. Then, it outputs a signature $\sigma = (g^a, b, g^r)$.
- The deterministic signature verification algorithm Verify, on input the public parameters $pp_{\mathsf{IBS}} = (\mathbb{G}, q, g, g^z, G, H)$, an identity id, a message m and a signature $\sigma = (g^a, b, g^r)$, proceeds as follows. It outputs $\{\mathsf{accept}, \mathsf{reject}\}$ indicating whether or not the equation $g^b = g^a(g^r \cdot g^{z \cdot c})^d$ holds where $c = H(g^r, \mathsf{id})$ and $d = G(\mathsf{id}, g^a, m)$.

In [5], Hwang et al., presented a modified GG-IBS (we call it modified GG-IBS) scheme that is used as a building block for their Sim-PWIBS protocol. A difference from GG-IBS is that, in the signing algorithm Sign, it computes $b \equiv a - w \cdot G(\mathsf{id}, g^a, m) \bmod q$ and outputs a signature $\sigma = (d, b, g^r)$ where $d = G(\mathsf{id}, g^a, m)$. Accordingly, Verify computes $g^a = g^b(g^r \cdot g^{z \cdot c})^d$ where $c = H(g^r, \mathsf{id})$, and outputs $\{\mathsf{accept}, \mathsf{reject}\}$ indicating whether or not $d = G(\mathsf{id}, g^a, m)$ holds.

3. Simplified PWIBS-AKE Protocol

In this section, we describe the simplified PWIBS-AKE (Sim-PWIBS) protocol [5] which consists of **Initialization** and **Key Establishment** phases.

3.1 Initialization

In this phase, it executes the following three processes Setup, $\mathsf{Extract}$ and $\mathsf{Registration}$.

3.1.1 Setup

The Setup on input 1^λ outputs public parameters pp and a master secret key msk where $(\mathbb{G}, q, g, h)$ is generated by calling the group generation algorithm Gen on input 1^λ, (g, h) are two random generators of $\mathbb{G}$, and $G, H, \mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^\star$ and $\mathsf{H}_2 : \{0,1\}^* \rightarrow \{0,1\}^k$ are descriptions of cryptographic hash functions. Also, it chooses a random element $z \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and sets $(\mathsf{mpk}, \mathsf{msk}) = (g^z, z)$. It outputs $(pp, \mathsf{msk}) = ((\mathbb{G}, q, g, h, g^z, G, H, \mathsf{H}_1, \mathsf{H}_2), z)$.

3.1.2 Extract

The Extract (run by KGC), on input the public parameters pp, the master secret key $\mathsf{msk}(= z)$ and an identity S, chooses $r \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $w \equiv r + z \cdot H(g^r, S) \bmod q$. Then, it outputs a private key $\mathsf{sk}_S = (w, g^r)$ that is securely transmitted to the corresponding server S.

3.1.3 Registration

First, client C randomly chooses his/her password pw from a dictionary $\mathbb{D}_{\mathsf{pw}}$ and sends $(C, h^{-\mathsf{H}_1(pw)})$ to server S securely. Then, the server stores $(C, h^{-\mathsf{H}_1(pw)})$ to a password file. Note that password pw is kept by client C secretly, and $(\mathsf{sk}_S, (C, h^{-\mathsf{H}_1(pw)}))$ are held by server S secretly.

3.2 Key Establishment

In this phase, client C and server S execute the Sim-PWIBS protocol in order to share a session key to be used for protecting subsequent communications. This phase of Sim-PWIBS has three steps as below.

Step 1. The client C chooses a random element $x \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $W \equiv g^x \cdot h^{\mathsf{H}_1(pw)}$ using the password pw. Then, client C sends (C, W) to server S.

Step 2. The server S chooses a random element $y \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $Y \equiv g^y$. Also, using its private key $\mathsf{sk}_S = (w, g^r)$, server S chooses $a \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$, and computes $b \equiv a - w \cdot G(S, g^a, m) \bmod q$ and a signature $\sigma = (d, b, g^r)$ on a message $m = S||Y$ where $d = G(S, g^a, m)$. Then, the server sends (S, Y, σ) to the client. After receiving a message (C, W) from client C, server S computes $X' \equiv W \cdot h^{-\mathsf{H}_1(pw)}$ and a Diffie-Hellman key $K' \equiv (X')^y$. Finally, the server computes a session key $SK_S = \mathsf{H}_2(pid||sid||K')$ where $pid = C||S$ and $sid = C||W||Y||\sigma$.

Step 3. After receiving a message (S, Y, σ) from server S, client C checks whether or not the signature σ is valid. If the equation $d = G(S, g^a, m)$ does not hold, where $g^a = g^b(g^r \cdot g^{z \cdot c})^d$, $c = H(g^r, S)$ and $m = S||Y$, the client aborts the protocol. Otherwise, client C computes a Diffie-Hellman key $K \equiv Y^x$ and a session key $SK_C = \mathsf{H}_2(pid||sid||K)$ where $pid = C||S$ and $sid = C||W||Y||\sigma$.

4. An Attack on Sim-PWIBS

In this section, we show that a malicious KGC (Key Generation Center) can impersonate the server and find out all clients' passwords in the Sim-PWIBS protocol [5].

4.1 An Active Attack on Sim-PWIBS

Here, we show an active attack on the Sim-PWIBS protocol [5] where a malicious KGC can impersonate the server and find out all clients' passwords with off-line dictionary attacks.

Let $\mathsf{SE}(SK, msg)$ and $\mathsf{AE}(SK, msg)$ be any symmetric-key and authenticated encryption schemes, respectively, where SK is a key and msg is a message. In the below, the malicious KGC who has the master secret key $\mathsf{msk}(= z)$ corresponding to the master public key $\mathsf{mpk}(= g^z)$ impersonates the server in the key establishment phase of Sim-PWIBS.

Step 1'. This is the same as in **Step 1** of Section 3.2.

Step 2'. The malicious KGC chooses a random element $y \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $Y \equiv g^y$. Also, it runs the Extract (as in Section 3.1.2) for an identity S so as to generate a private key sk_S. Using its (newly generated) private key sk_S, the malicious KGC computes a signature σ on a message $m = S||Y$ (with the same way as in **Step 2** of Section 3.2). Then, it sends (S, Y, σ) to the client. After receiving a message (C, W) from client C, the malicious KGC computes $\{X' \equiv W \cdot h^{-\mathsf{H}_1(pw')}, K' \equiv (X')^y\}$ for all password candidates $pw' \in \mathbb{D}_{\mathsf{pw}}$.

Step 3'. This is the same as in **Step 3** of Section 3.2.

If client C sends a ciphertext $\mathsf{SE}(SK_C, msg)$ or $\mathsf{AE}(SK_C, msg)$ to the malicious KGC, the latter can find out the client's password pw by applying all possible session keys $SK = \mathsf{H}_2(pid||sid||K')$ to $\mathsf{SE}(SK_C, msg)$ or $\mathsf{AE}(SK_C, msg)$ where $pid = C||S$ and $sid = C||W||Y||\sigma$. Of course, these off-line dictionary attacks can be used for all clients who registered to the server S.

5. A Secure Simplified PWIBS-AKE (PAKEwIBS1) Protocol

In this section, we propose a secure simplified PWIBS-AKE (for short, PAKEwIBS1) protocol that provides security against server impersonation and off-line dictionary attacks by a malicious KGC (Key Generation Center). The PAKEwIBS1 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^λ outputs public parameters pp and a master secret key msk where $(\mathbb{G}, q, g, h)$ is generated by calling the group generation algorithm Gen on input 1^λ, (g, h) are two random generators of $\mathbb{G}$, and $G, H, \mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^\star$ and $\mathsf{H}_2, \mathsf{H}_3 : \{0,1\}^* \rightarrow \{0,1\}^k$ are descriptions of cryptographic hash functions. Also, it chooses a random element $z \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and sets $(\mathsf{mpk}, \mathsf{msk}) = (g^z, z)$. It outputs $(pp, \mathsf{msk}) = ((\mathbb{G}, q, g, h, g^z, G, H, \mathsf{H}_1, \mathsf{H}_2, \mathsf{H}_3), z)$.

5.1.2 Extract

The Extract (run by KGC), on input the public parameters pp, the master secret key $\mathsf{msk}(= z)$ and an identity S, chooses $r \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $w \equiv r + z \cdot H(g^r, S) \bmod q$. Then, it outputs a private key $\mathsf{sk}_S = (w, g^r)$ 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 $\mathbb{D}_{\mathsf{pw}}$ and sends $(C, h^{-\mathsf{H}_1(pw)})$ to server S. Then, the server stores $(C, h^{-\mathsf{H}_1(pw)})$ to a password file. Note that password pw is kept by client C secretly, and $(\mathsf{sk}_S, (C, h^{-\mathsf{H}_1(pw)}))$ 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 PAKEwIBS1 protocol in order to share a session key to be used for protecting subsequent communications. This phase of PAKEwIBS1 has three steps as below.

Step 1. The client C chooses a random element $x \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$, and computes a Diffie-Hellman public value $X \equiv g^x$ and its masked value $W \equiv X \cdot h^{\mathsf{H}_1(pw)}$ using the password pw. Then, client C sends (C, W) to server S.

Step 2. The server S chooses a random element $y \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes a Diffie-Hellman public value $Y \equiv g^y$. After receiving a message (C, W) from client C, server S computes $X' \equiv W \cdot h^{-\mathsf{H}_1(pw)}$ and a Diffie-Hellman key $K' \equiv (X')^y$. Also, the server computes its authenticator $V_S = \mathsf{H}_2(C||S||W||Y||X'||K')$. Using its private key $\mathsf{sk}_S = (w, g^r)$, server S generates a signature σ on a message $m = C||S||W||Y||V_S$ according to the signing algorithm Sign of the (modified) GG-IBS scheme. Then, the server sends (S, Y, V_S, σ) to the client. Finally, server S computes a session key $SK_S = \mathsf{H}_3(sid||X'||K')$ where $sid = m||\sigma$.

Step 3. After receiving a message (S, Y, V_S, σ) from server S, client C first checks whether or not the signature σ on a message $m = C||S||W||Y||V_S$ for an identity S is valid. If the signature verification algorithm Verify of the (modified) GG-IBS scheme outputs Reject, the client aborts the protocol. Otherwise, client C computes a Diffie-Hellman key $K \equiv Y^x$ and checks the validity of V_S. If $V_S \neq \mathsf{H}_2(C||S||W||Y||X||K)$, the client aborts the protocol. Otherwise, client C computes a session key $SK_C = \mathsf{H}_3(sid||X||K)$ where $sid = m||\sigma$.

6. Another Secure Simplified PWIBS-AKE (PAKEwIBS2) Protocol

In this section, we propose another secure simplified PWIBS-AKE (for short, PAKEwIBS2) protocol that also provides security against server impersonation and off-line dictionary attacks by a malicious KGC (Key Generation Center). The PAKEwIBS2 protocol consists of **Initialization** and **Key Establishment** phases.

6.1 Initialization

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

6.1.1 Setup

The Setup on input 1^λ outputs public parameters pp and a master secret key msk where $(\mathbb{G}, q, g, h_1, h_2)$ is generated by calling the group generation algorithm Gen on input 1^λ, (g, h_1, h_2) are three random generators of $\mathbb{G}$, and $G, H, \mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^\star$ and $\mathsf{H}_2 : \{0,1\}^* \rightarrow \{0,1\}^k$ are descriptions of cryptographic hash functions. Also, it chooses a random element $z \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and sets $(\mathsf{mpk}, \mathsf{msk}) = (g^z, z)$. It outputs $(pp, \mathsf{msk}) = ((\mathbb{G}, q, g, h_1, h_2, g^z, G, H, \mathsf{H}_1, \mathsf{H}_2), z)$.

6.1.2 Extract

The Extract (run by KGC), on input the public parameters pp, the master secret key $\mathsf{msk}(= z)$ and an identity S, chooses $r \overset{\$}{\leftarrow} \mathbb{Z}_q^\star$ and computes $w \equiv r + z \cdot H(g^r, S) \bmod q$. Then, it outputs a private key $\mathsf{sk}_S = (w, g^r)$ that is securely transmitted to the corresponding server S.

6.1.3 Registration

First, client C randomly chooses his/her password pw from a dictionary $\mathbb{D}_{\mathsf{pw}}$ and sends $(C, h_1^{-\mathsf{H}_1(pw)}, h_2^{\mathsf{H}_1(pw)})$ to server S. Then, the server stores $(C, h_1^{-\mathsf{H}_1(pw)}, h_2^{\mathsf{H}_1(pw)})$ to a password file. Note that password pw is kept by client

C secretly, and $(\mathsf{sk}_S, (C, h_1^{-\mathsf{H}_1(pw)}, h_2^{\mathsf{H}_1(pw)}))$ are held by server S secretly. This registration process should be done securely between client C and server S.

6.2 Key Establishment

In this phase, client C and server S execute the PAKEwIBS2 protocol in order to share a session key to be used for protecting subsequent communications. This phase of PAKEwIBS2 has three steps as below.

Step 1. The client C chooses a random element $x \xleftarrow{\$} \mathbb{Z}_q^\star$, and computes a Diffie-Hellman public value $X \equiv g^x$ and its masked value $W \equiv X \cdot h_1^{\mathsf{H}_1(pw)}$ using the password pw. Then, client C sends (C, W) to server S.

Step 2. The server S chooses a random element $y \xleftarrow{\$} \mathbb{Z}_q^\star$, and computes a Diffie-Hellman public value $Y \equiv g^y$ and its masked value $Z \equiv Y \cdot h_2^{\mathsf{H}_1(pw)}$. Using its private key $\mathsf{sk}_S = (w, g^r)$, server S generates a signature σ on a message $m = C||S||W||Z$ according to the signing algorithm Sign of the (modified) GG-IBS scheme. Then, the server sends (S, Z, σ) to the client. After receiving a message (C, W) from client C, server S computes $X' \equiv W \cdot h^{-\mathsf{H}_1(pw)}$ and a Diffie-Hellman key $K' \equiv (X')^y$. Finally, the server computes a session key $SK_S = \mathsf{H}_2(sid||X'||Y||K')$ where $sid = m||\sigma$.

Step 3. After receiving a message (S, Z, σ) from server S, client C checks whether or not the signature σ on a message $m = C||S||W||Z$ for an identity S is valid. If the signature verification algorithm Verify of the (modified) GG-IBS scheme outputs Reject, the client aborts the protocol. Otherwise, client C computes $Y' \equiv Z \cdot h_2^{-\mathsf{H}_1(pw)}$ and a Diffie-Hellman key $K \equiv (Y')^x$. Finally, the client computes a session key $SK_C = \mathsf{H}_2(sid||X||Y'||K)$ where $sid = m||\sigma$.

謝辞 This work was partly supported by JST, Infrastructure Development for Promoting International S&T Cooperation and JSPS KAKENHI Grant Number JP16H02834.

参考文献

[1] M. Bellare, C. Namprempre, and G. Neven, "Security Proofs for Identity-Based Identification and Signature Schemes," *Journal of Cryptology*, Vol. 22, Issue 1, pp. 1-61, Springer, 2009.

[2] M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," In *Proc. of ACM CCS'93*, pp. 62-73, ACM, 1993.

[3] S. Chatterjee, C. Kamath, and V. Kumar, "Galindo-Garcia Identity-Based Signature Revisted," In *Proc. of ICISC 2012*, LNCS 7839, pp. 456-471, Springer, 2013.

[4] D. Galindo and F. D. Garcia, "A Schnorr-Like Lightweight Identity-Based Signature Scheme," In *Proc. of AFRICACRYPT 2009*, LNCS 5580, pp. 135-148, Springer, 2009.

[5] J. Y. Hwang, S. H. Kim, D. Choi, S. H. Jin, and B. Song, "Robust Authenticated Key Exchange Using Passwords and Identity-Based Signatures," In *Proc. of International Conference on Research in Security Standardisation (SSR) 2015*, LNCS 9497, pp. 43-69, Springer, 2015.

[6] S. M. Bellovin and M. Merritt, "Encrypted Key Exchange: Password-based Protocols Secure against Dictionary Attacks", In *Proc. of IEEE Symposium on Security and Privacy*, pp. 72-84, IEEE Computer Society, 1992.

[7] S. M. Bellovin and M. Merritt, "Augmented Encrypted Key Exchange: A Password-based Protocol Secure against Dictionary Attacks and Password File Compromise", In *Proc. of ACM CCS'93*, pp. 244-250, ACM Press, 1993.

[8] R. Canetti, D. Dachman-Soled, V. Vaikuntanathan, and H. Wee, "Efficient Password Authenticated Key Exchange via Oblivious Transfer", In *Proc. of PKC 2012*, LNCS 7293, pp. 449-466, Springer-Verlag, 2012.

[9] C. Gentry, P. MacKenzie, and Z. Ramzan, "A Method for Making Password-Based Key Excahnge Resilient to Server Compromise", In *Proc. of CRYPTO 2006*, LNCS 4117, pp. 142-159, Springer-Verlag, 2006.

[10] Submissions to IEEE P1363.2. Available at `http://grouper.ieee.org/groups/1363/passwdPK/submissions.html`.

[11] IEEE 1363.2, "IEEE Standard Specifications for Password-Based Public-Key Cryptographic Techniques", IEEE Std 1363.2$^{\mathrm{TM}}$-2008, IEEE Computer Society, January 2009.

[12] ISO/IEC 11770-4, "Information Technology−Security Techniques−Key Management−Part 4: Mechanisms Based on Weak Secrets", International Standard ISO/IEC 11770-4:2006(E), May 2006.

[13] ITU-T Recommendation X.1035, "Password-Authenticated Key Exchange (PAK) Protocol", Series X: Data Networks, Open System Communications and Security, February 2007. Available at `http://www.itu.int/rec/T-REC-X.1035-200702-I/en`.

[14] Research Papers on Password-based Cryptography. Available at `http://www.jablon.org/passwordlinks.html`.

[15] W. Simpson, "PPP Challenge Handshake Authentication Protocol (CHAP)," IETF RFC 1994, August 1996. Available at `http://www.ietf.org/rfc/rfc1994.txt`.

[16] T. Wu, "The SRP Authentication and Key Exchange System", IETF RFC 2945, September 2000. Available at `http://www.ietf.org/rfc/rfc2945.txt`.