Security Analysis of Password-Authenticated Key Retrieval

SeongHan Shin and Kazukuni Kobara

***Abstract*—A PAKR (Password-Authenticated Key Retrieval) protocol and its multi-server system allow one party (say, client), who has a rememberable password, to retrieve a long-term static key in an exchange of messages with at least one other party (say, server) that has a private key associated with the password. In this paper, we analyze the only one PAKR (named as PKRS-1) standardized in IEEE 1363.2 [9] and its multi-server system (also, [11]) by showing that any passive/active attacker can find out the client's password and the static key with off-line dictionary attacks. This result is contrary to the security statement of PKRS-1 (see Chapter 10.2 of IEEE 1363.2 [9]).**

***Index Terms*—Password authentication, key retrieval, on-line/off-line dictionary attacks, IEEE 1363.2.**

I. Introduction

THE problem of safely storing client's long-term static keys (e.g., symmetric keys, signature keys for digital signatures or decryption keys for public-key encryptions) can be addressed with credential services (including cloud services or SSO (Single Sign-On)), which also solve many usability constraints for clients. Consider a roaming client who accesses a network from different locations in order to retrieve his/her static keys (e.g., for temporal use of PKI (Public-Key Infrastructures)). This kind of roaming model can be supported by a credentials server that authenticates the client and then assists in downloading static keys for the client.

For authentication in the roaming model, several works [14], [12], [15], [16], [6] utilized PAKE (Password-Authenticated Key Exchange) protocols that provide password-only authentication and establishment of temporal session keys to protect subsequent communications. The concept of PAKE protocols was introduced by Bellovin and Merritt [3], [4] where a client remembers a short password only (without any storage devices) and the corresponding server authenticates the client with the password or its verification data for verifying the client's knowledge of the password (see [8], [10] and references therein). However, one should be very careful about two major attacks on passwords: on-line and off-line dictionary attacks. The on-line dictionary attacks are performed by an attacker who impersonates one party so that the attacker 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 attacker exhaustively enumerates all possible password candidates, in an attempt to determine the correct one. The latter attacks are possible since passwords are chosen from a relatively-small dictionary that allows the exhaustive searches. While on-line dictionary attacks can be prevented by taking appropriate countermeasures (e.g., lock-up accounts after some consecutive failures), off-line dictionary attacks can not be avoided by such countermeasures.

By utilizing PAKE protocols in the roaming model, Perlman and Kaufman [14] showed that simple modifications of the underlying PAKE protocols are sufficient for secure roaming access to credentials. Such an example for MyProxy was proposed in [6]. Also, two different approaches (called, virtual soft token and virtual smartcard) have been proposed with the name of password-enabled PKI [14], [12], [15], [16] in order to integrate convenience of password authentication into the conventional PKI.

As another approach for the roaming model, Ford and Kaliski [5] proposed some protocols (later, named as PAKR (Password-Authenticated Key Retrieval)) using multiple n servers, each of which holds a share of static keys, in order to provide security of passwords/static keys against server compromises. That is, even if an attacker takes full control of up to $n-1$ servers, the attacker will not be able to verify a single guess for the password and get any information about the static key. In order to prevent off-line dictionary attacks, PAKR protocols in [5] rely on a prior server-authenticated secure channel such as SSL/TLS which means it may be vulnerable to web-spoofing/phishing attacks. In [11], Jablon proposed a PAKR protocol using multiple servers which does not need a prior server-authenticated secure channel. Also, see [2] for another PAKR based on the unique blind signature [1]. Differently from password-enabled PKI, the retrieved static key in PAKR [5], [11], [2] is derived from both the client's password and the server's private key.[1]

A. Our Contributions

Based on [5], [11], PKRS-1 (Password-authenticated Key Retrieval Scheme, version 1) has been standardized and was included in IEEE 1363.2 standard [9]. In this paper, we revisit PKRS-1 and its multi-server system (also, [11]) to show that any passive/active attacker can find out the client's password and the (long-term) static key with off-line dictionary attacks. This result is contrary to the security statement of PKRS-1 (see Chapter 10.2 of IEEE 1363.2 [9]). Note that PKRS-1 is the only one PAKR (Password-Authenticated Key Retrieval) in IEEE 1363.2 standard [9].

S. H. Shin and K. Kobara are with the Research Institute for Secure Systems (RISEC), National Institute of Advanced Industrial Science and Technology (AIST), Ibaraki, 305-8568 Japan (e-mail: seonghan.shin@aist.go.jp).

[1] In addition, the retrieved static key can be used to encrypt the client's signature/decryption keys for a later use.

B. Notation

Here, we explain some notation to be used throughout this paper. Let $\mathbb{G}$ be a finite, cyclic subgroup of prime order q of the multiplicative group $\mathbb{Z}_p^\star$ where $p = aq+1$ is a prime and a is an integer. Let g_a and g_b be two distinct generators of $\mathbb{G}$ in which the group operation is denoted multiplicatively. Note that an exponential relationship between g_a and g_b should be unknown. The (p, q, g_a, g_b) are public to everyone and (p, q) are called domain parameters. In the aftermath, all the subsequent arithmetic operations are performed in modulo p unless otherwise stated.

Let k be the security parameter for hash functions. Let $\{0,1\}^\star$ denote the set of finite binary strings and $\{0,1\}^k$ the set of binary strings of length k. We use two different hash functions H and $\mathcal{H}$ where $H : \{0,1\}^\star \rightarrow \mathbb{Z}_q^\star$ and $\mathcal{H} : \{0,1\}^\star \rightarrow \{0,1\}^k$. Let KDF be the key derivation function, which can be instantiated with a secure one-way hash function $\mathcal{H}$ (see Section 12.3 and 11 of [9]), and P be the key derivation parameter for KDF. Also, let A, B be the concatenation of bit strings of A and B in $\{0,1\}^\star$. Let C and S be the identities of client and server, respectively, with each identity ID $\in \{0,1\}^\star$.

II. PKRS-1

In this section, we describe PKRS-1 (Chapter 10.2 and Annex D.2.2.3.4 of IEEE 1363.2 [9]) in detail where PKRS is an acronym for Password-authenticated Key Retrieval Scheme. The PKRS-1 is actually based on [5], [11], and it consists of key establishment operation phase and key confirmation operation phase for an invoking application. Note that this scheme is the only one PAKR included in IEEE 1363.2 standard [9].

A. Key Establishment Operation

Whenever client C needs to retrieve a static key with the assistance of server S, they execute the below key establishment operation over insecure networks. Before this operation, client C just remembers his/her password pw and server S has its private key $u \in [1, q-1]$ that corresponds to the password pw.[2] During the key establishment operation, client C and server S exchange values, and then client C retrieves a static key K derived from both the client's password pw and the server's associated private key u.

Step 1: First, client C computes a generator value $R \equiv g_a \cdot g_b^{H(pw)}$ (by REDP-2 (Random Element Derivation Primitive, version 2) in Chapter 8.2.18 of IEEE 1363.2 [9]) from his/her password pw. Also, the client selects a random secret s from the range $[1, q-1]$ and computes a blinded password value $W_C \equiv R^s$. Then, client C sends the first message (C, W_C) to server S.

$$C \rightarrow S : (C, W_C)$$

Step 2: After receiving (C, W_C), server S checks whether the client's blinded password value W_C is in the parent group or not. If $W_C \notin [1, p-1]$, it outputs ”invalid” and stops (Server validation 1). The server also checks whether W_C is a valid value or not (i.e., the order check of W_C). If $(W_C)^q \not\equiv 1$, it outputs ”invalid” and stops (Server validation 2).[3] Otherwise, server S computes a permuted blinded password value $W_S \equiv (W_C)^u$ with its private key u. Then, server S sends the second message (S, W_S) to client C.

$$S \rightarrow C : (S, W_S)$$

Step 3: After receiving (S, W_S), client C checks whether the server's permuted blinded password value W_S is in the parent group or not. If $W_S \notin [1, p-1]$, it outputs ”invalid” and stops (Client validation 1). The client also checks whether W_S is a valid value or not (i.e., the order check of W_S). If $(W_S)^q \not\equiv 1$, it outputs ”invalid” and stops (Client validation 2). Otherwise, client C computes a permuted password value $Z \equiv (W_S)^{1/s}$ with the secret s, and then derives a static key $K = \mathsf{KDF}(Z, P)$ from Z where P is the key derivation parameter.

B. Key Confirmation Operation

In general, any invoking application that uses the key establishment operation of Section II-A is required to execute the following key confirmation operation (Annex D.2.2.3.4.2 of IEEE 1363.2 [9]). This operation is important in order to prevent disclosure of the client's password pw to other parties (particularly, an attacker who impersonates server S). Before this operation, server S has a key confirmation value $V_S = \mathcal{H}(Z)$. During the key confirmation operation, client C confirms that the permuted password value Z (or equivalently, the static key K derived from Z) is correct before revealing any information about Z to other parties.

Step 4: The server S sends the key confirmation value V_S to client C.

$$S \rightarrow C : V_S$$

Step 5: After receiving V_S, client C checks correctness of the permuted password value Z by verifying if V_S is equal to $\mathcal{H}(Z)$.

Note that $Z \equiv R^u \equiv \left(g_a \cdot g_b^{H(pw)}\right)^u$.

III. Security Analysis of PKRS-1

In this section, we analyze PKRS-1 of Section II in terms of security of the client's password as well as the static key. According to Chapter 10.2 of IEEE 1363.2 [9], it is clearly stated that PKRS-1 prevents both an active attacker (who impersonates the client) and a passive attacker (who eavesdrops the values exchanged between the legitimate client

[2] Note that the server's private key does not, in itself, contain sufficient information to allow the server to determine the client's password or the retrieved static key.

[3] If the order of W_C has too many small factors, a Pohlig-Hellman decomposition attack [13] on the server's private key u is possible. This attack can be prevented by 1) the order check of W_C (see Annex A.16.6 of IEEE 1363-2000 [7]) or 2) the selection of domain parameters such as a safe prime $p = 2q+1$ or a secure prime $p = aq+1$ s.t. $(a/2)$ is also a prime.

and server) from being able to derive the client's password pw, the server's private key u, or the retrieved static key K, even with off-line dictionary attacks on the password.

A. An Attack on PKRS-1

Here, we show an attack on PKRS-1 where a passive/active attacker can find out the client's password pw and the retrieved static key K with off-line dictionary attacks. The success probability of this attack is 1.

Consider an attacker $\mathcal{A}$ who impersonates client C without knowing the password pw. The attacker executes the following key establishment operation of Section II-A with server S.

Step 1': First, attacker $\mathcal{A}$ computes a generator value $R' \equiv g_a \cdot g_b^{H(pw')}$ where pw' is a guessed password. Also, the attacker selects a random secret s from the range $[1, q-1]$ and computes a blinded password value $W'_C \equiv (R')^s$. Then, attacker $\mathcal{A}$ sends the first message (C, W'_C) to server S.

$$\mathcal{A} \rightarrow S : (C, W'_C)$$

Step 2': Same as **Step 2** of Section II-A

Step 3': After receiving $(S, W'_S \equiv (W'_C)^u)$, attacker $\mathcal{A}$ computes a permuted password value $Z' \equiv (W'_S)^{1/s}$ with the secret s.

Next, the attacker repeats the above operation with another guessed password $pw''(\neq pw')$ so as to get $Z'' \equiv (W''_S)^{1/s}$ where $W''_S \equiv (W''_C)^u$, $W''_C \equiv (R'')^s$ and $R'' \equiv g_a \cdot g_b^{H(pw'')}$.

Let $w \equiv (H(pw') - H(pw'')) \bmod q$. From Z' and Z'', attacker $\mathcal{A}$ obtains g_b^u and g_a^u as follows:

$$\left(\frac{Z'}{Z''}\right)^{1/w} \equiv \left(\frac{(R')^u}{(R'')^u}\right)^{1/w} \equiv \left(\frac{g_a \cdot g_b^{H(pw')}}{g_a \cdot g_b^{H(pw'')}}\right)^{u/w} \equiv g_b^u \quad (1)$$

and

$$\frac{Z'}{g_b^{u \cdot H(pw')}} \equiv \frac{\left(g_a \cdot g_b^{H(pw')}\right)^u}{g_b^{u \cdot H(pw')}} \equiv g_a^u \ . \quad (2)$$

After eavesdropping the key confirmation value V_S between client C and server S, the attacker tests if V_S is equal to $\mathcal{H}(g_a^u \cdot g_b^{u \cdot H(\widetilde{pw})})$ for all possible password candidates $\widetilde{pw}$. With this test, attacker $\mathcal{A}$ can find out the client's password $pw(=\widetilde{pw})$ from which the retrieved static key K is also easily derived because $K = \mathsf{KDF}(Z(\equiv g_a^u \cdot g_b^{u \cdot H(pw)}), P)$.

B. Multi-Server System of PKRS-1

From Annex D.2.2.3.4.1 of IEEE 1363.2 [9], PKRS-1 is extended for a multi-server system [5], [11] where the key shares established between client C and each server S_i are combined into the static key. However, the above attack of Section III-A can be readily applied to the multi-server system of PKRS-1. For more concreteness, we show that PAKR using multiple servers [11] is insecure in Section V.

IV. PAKR using Multiple Servers

In this section, we describe PAKR using multiple servers (for short, PAKR-M) [11] in detail where client C uses his/her password pw to retrieve the static key K from key shares that have been distributed and stored with n servers S_i $(1 \leq i \leq n)$. By using multiple servers, PAKR-M and [5] can reduce vulnerabilities such as exposure of passwords and/or static keys due to (up to $n-1$) server compromises. Note that PAKR-M mainly differs from [5] in that the former does not require a prior server-authenticated secure channel (e.g., SSL/TLS) between client C and each server S_i. The PAKR-M consists of enrollment phase and authenticated retrieval (i.e., key establishment plus key confirmation) phase.

A. Notation for PAKR-M

Here, we explain additional notation for PAKR-M. The PAKR-M uses specific domain parameters (p, q) where $p = aq + 1$ is a secure prime such that $(a/2)$ is also a prime. Let $(SigK, VerK)$ be the signature and verification key pair for digital signatures, and $\mathsf{Sign}_{SigK}(msg)$ be the signature of message msg with signature key $SigK$. Also, let $\mathsf{E}_K(\cdot)$ and $\mathsf{D}_K(\cdot)$ be the encryption/decryption with symmetric key K satisfying $msg = \mathsf{D}_K(\mathsf{E}_K(msg))$.

B. Enrollment

In the enrollment phase, client C registers a set of credentials to each server S_i $(1 \leq i \leq n)$. This phase should be done securely and is performed only once between client C and server S_i.

First, client C who remembers his/her password pw computes a generator value $R \equiv g_a \cdot g_b^{H(pw)}$ (see Section 4.3 of [11]). For server S_i, the client selects a random private key u_i from the range $[1, q-1]$ and computes a permuted password value (i.e., key share) $Z_i \equiv R^{u_i}$. By combining all Z_i, client C derives a static key $K = \mathcal{H}(Z_1, Z_2, \cdots, Z_n)$ and a key confirmation value $V_S = \mathcal{H}(K, R)$. Second, the client generates a signature/verification key pair $(SigK, VerK)$ for digital signatures, and an encryption $\mathsf{E}_K(SigK)$ of $SigK$ with the static key K. Finally, client C registers a set of credentials $(C, u_i, VerK, \mathsf{E}_K(SigK), V_S)$ to each server S_i.

After the enrollment phase, client C just remembers the password pw and server S_i has the set of credentials $(C, u_i, VerK, \mathsf{E}_K(SigK), V_S)$.

C. Authenticated Retrieval

In the authenticated retrieval phase, client C retrieves the static key K (from key shares Z_i) with the assistance of server S_i $(1 \leq i \leq n)$ over insecure networks whenever K is needed.

Step 1: First, client C computes the generator value $R \equiv g_a \cdot g_b^{H(pw)}$ from his/her password pw. Also, the client selects a random secret s from the range $[1, q-1]$ and computes a blinded password value $W_C \equiv R^s$. Then, client C sends the first message (C, W_C) to all servers S_i.

$$C \rightarrow S_i : (C, W_C)$$

Step 2: After receiving (C, W_C), server S_i adds $(C, W_C, VerK)$ to the list $List_i$ that is used to distinguish honest client's behaviors from on-line dictionary attacks on the password. The server computes a permuted blinded password value $W_{S_i} \equiv (W_C)^{u_i}$ with its private key u_i. Then, server S_i sends the second message $(S_i, W_{S_i}, \mathsf{E}_K(SigK), V_S)$ to client C.

$$S_i \rightarrow C : (S_i, W_{S_i}, \mathsf{E}_K(SigK), V_S)$$

Step 3: After receiving $(S_i, W_{S_i}, \mathsf{E}_K(SigK), V_S)$, client C computes the permuted password value $Z_i \equiv (W_{S_i})^{1/s}$ with the secret s for each i $(1 \leq i \leq n)$. From all Z_i, the client derives the static key $K = \mathcal{H}(Z_1, Z_2, \cdots, Z_n)$ and checks whether the received key confirmation value V_S is valid or not. If $V_S \neq \mathcal{H}(K, R)$, it outputs "invalid" and stops. Otherwise, client C recovers the signature key $SigK = \mathsf{D}_K(\mathsf{E}_K(SigK))$ and generates a signature $\sigma = \mathsf{Sign}_{SigK}(W_C)$ with $SigK$. Then, the client sends the third message σ to all servers S_i.

$$C \rightarrow S_i : \sigma$$

Step 4: After receiving σ, server S_i checks whether the signature σ is valid or not with the verification key $VerK$. If σ is valid, server S_i removes $(C, W_C, VerK)$ from the list $List_i$. Otherwise, the W_C is counted as an on-line dictionary attack on the password pw.

V. Security Analysis of PAKR-M

Similarly to Section III-A, we show an attack on PAKR-M where an active attacker can find out the client's password pw and the retrieved static key K with off-line dictionary attacks. Of course, the success probability of this attack is 1.

Consider an attacker $\mathcal{A}$ who impersonates client C without knowing the password pw. The attacker executes the following authenticated retrieval of Section IV-C with server S_i $(1 \leq i \leq n)$.

Step 1': First, attacker $\mathcal{A}$ computes a generator value $R' \equiv g_a \cdot g_b^{H(pw')}$ where pw' is a guessed password. Also, the attacker selects a random secret s from the range $[1, q-1]$ and computes a blinded password value $W'_C \equiv (R')^s$. Then, attacker $\mathcal{A}$ sends the first message (C, W'_C) to all servers S_i.

$$\mathcal{A} \rightarrow S_i : (C, W'_C)$$

Step 2': Same as **Step 2** of Section IV-C

Step 3': After receiving $(S_i, W'_{S_i} \equiv (W'_C)^{u_i}, \mathsf{E}_K(SigK), V_S)$, attacker $\mathcal{A}$ computes a permuted password value $Z'_i \equiv (W'_{S_i})^{1/s}$ with the secret s for each i $(1 \leq i \leq n)$, and then stops.

Step 1": Again, attacker $\mathcal{A}$ computes a generator value $R'' \equiv g_a \cdot g_b^{H(pw'')}$ with another guessed password $pw''(\neq pw')$. Also, the attacker selects a random secret $s \in [1, q-1]$ and computes $W''_C \equiv (R'')^s$. Then, attacker $\mathcal{A}$ sends (C, W''_C) to all servers S_i.

$$\mathcal{A} \rightarrow S_i : (C, W''_C)$$

Step 2": Same as **Step 2** of Section IV-C

Step 3": After receiving $(S_i, W''_{S_i} \equiv (W''_C)^{u_i}, \mathsf{E}_K(SigK), V_S)$, attacker $\mathcal{A}$ computes $Z''_i \equiv (W''_{S_i})^{1/s}$ with the secret s for each i $(1 \leq i \leq n)$. Let $w \equiv (H(pw') - H(pw'')) \bmod q$. From Z'_i and Z''_i, attacker $\mathcal{A}$ obtains $g_b^{u_i}$ and $g_a^{u_i}$ for each i as follows:

$$\left(\frac{Z'_i}{Z''_i}\right)^{1/w} \equiv g_b^{u_i} \text{ and } \frac{Z'_i}{g_b^{u_i \cdot H(pw')}} \equiv g_a^{u_i} \ . \qquad (3)$$

Now, the attacker tests if the key confirmation value V_S is equal to $\mathcal{H}(\widetilde{K}, g_a \cdot g_b^{H(\widetilde{pw})})$, where $\widetilde{K} = \mathcal{H}(g_a^{u_1} \cdot g_b^{u_1 \cdot H(\widetilde{pw})}, g_a^{u_2} \cdot g_b^{u_2 \cdot H(\widetilde{pw})}, \cdots, g_a^{u_n} \cdot g_b^{u_n \cdot H(\widetilde{pw})})$, for all possible password candidates $\widetilde{pw}$. With this test, attacker $\mathcal{A}$ can find out the client's password $pw(= \widetilde{pw})$ and static key $K(= \widetilde{K})$. Also, the attacker recovers the signature key $SigK = \mathsf{D}_K(\mathsf{E}_K(SigK))$ and generates a valid signature $\sigma = \mathsf{Sign}_{SigK}(W'_C, W''_C)$. Then, attacker $\mathcal{A}$ sends the third message σ to all servers S_i.

$$\mathcal{A} \rightarrow S_i : \sigma$$

As above, executing the authenticated retrieval phase of PAKR-M two times is enough for attacker $\mathcal{A}$ to get the client's password pw, the static key K and the signature key $SigK$. However, this attack is not applicable to [5] because the latter uses a safe prime $p = 2q + 1$ and computes a generator value $R = \mathcal{G}(pw)^2$ where $\mathcal{G}$ is a full-domain hash mapping from $\{0, 1\}^\star$ to $\mathbb{Z}_p^\star$.

VI. Concluding Remarks

In this paper, we have analyzed PKRS-1 (standardized in IEEE 1363.2 [9]) and its multi-server system (also, [11]) by showing that any passive/active attacker can find out the client's password pw and the (long-term) static key K with off-line dictionary attacks. Notice that these attacks are always possible regardless of the selection of domain parameters (p, q) for PKRS-1 and its multi-server system [11].

This result is imperative for practitioners/implementers because PKRS-1 is the only one PAKR (Password-Authenticated Key Retrieval) included in IEEE 1363.2 standard [9] and deployment of PKRS-1 for real-world applications can result in a total compromise of security. Consequently, it is inevitable to revise IEEE 1363.2 standard [9] NOT to use the REDP function (i.e., REDP-2) for the generator value in PKRS-1 and its multi-server system.[4]

References

[1] A. Boldyreva, "Threshold Signatures, Multisignatures and Blind Signatures based on the Gap-Diffie-Hellman-Group Signature Scheme", In *Proc. of PKC 2003*, LNCS 2567, pp. 31-346, Springer-Verlag, 2003.

[2] X. Boyen, "Hidden Credential Retrieval from a Reusable Password", In *Proc. of ASIACCS 2009*, pp. 228-238, ACM Press, 2009.

[3] 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.

[4] Though REDP-2 is also used in the PAK/SPEKE family [9], these attacks shown in this paper are not applicable.

[4] 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.

[5] W. Ford and B. S. Kaliski, ”Server-Assisted Generation of a Strong Secret from a Password”, In *Proc. of the 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE 2000)*, pp. 176-180, IEEE Press, 2000.

[6] L. Fang, S. Meder, O. Chevassut, and F. Siebenlist, ”Secure Password-based Authenticated Key Exchange for Web Services”, In *Proc. of the ACM Workshop on Secure Web Services (SWS)*, ACM, 2004.

[7] IEEE 1363, ”IEEE Standard Specifications for Public-Key Cryptography”, IEEE Std 1363$^{\text{TM}}$-2000, IEEE Computer Society, 2000.

[8] Submissions to IEEE P1363.2. http://grouper.ieee.org/groups/1363/passwdPK/submissions.html.

[9] IEEE 1363.2, ”IEEE Standard Specifications for Password-based Public-Key Cryptographic Techniques”, IEEE Std 1363.2$^{\text{TM}}$-2008, IEEE Computer Society, January 2009.

[10] Research Papers on Password-based Cryptography. http://www.jablon.org/passwordlinks.html.

[11] D. P. Jablon, ”Password Authentication Using Multiple Servers”, In *Proc. of CT-RSA 2001*, LNCS 2020, pp. 344-360, Springer-Verlag, 2001.

[12] T. Kwon, ”Virtual Software Tokens - A Practical Way to Secure PKI Roaming”, In *Proc. of the Infrastructure Security (InfraSec)*, LNCS 2437, pp. 288-302. Springer-Verlag, 2002.

[13] P. C. van Oorschot and M. J. Wiener, ”On Diffie-Hellman Key Agreement with Short Exponents”, In *Proc. of EUROCRYPT’96*, LNCS 1070, pp. 332-343, Springer-Verlag, 1996.

[14] R. Perlman and C. Kaufman, ”Secure Password-based Protocol for Downloading a Private Key”, In *Proc. of Network and Distributed System Security Symposium*, Internet Security, 1999.

[15] R. Sandhu, M. Bellare, and R. Ganesan, ”Password Enabled PKI: Virtual Smartcards vs. Virtual Soft Tokens”, In *Proc. of the 1st Annual PKI Research Workshop*, pp. 89-96, 2002.

[16] X. Wang, ”Intrusion-Tolerant Password-Enabled PKI”, In *Proc. of the 2nd Annual PKI Research Workshop*, pp. 44-53, 2003.