Journal of Computer and System Sciences 74 (2008) 1160–1172

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Two-factor mutual authentication based on smart cards and passwords

Guomin Yang [a], Duncan S. Wong [a,*,1], Huaxiong Wang [b], Xiaotie Deng [a]

[a] *Department of Computer Science, City University of Hong Kong, Hong Kong, China*
[b] *School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore*

ARTICLE INFO

Article history:
Received 21 September 2006
Received in revised form 21 April 2008
Available online 15 May 2008

Keywords:
Two-factor authentication
Password
Smart-card
Guessing attack
Dictionary attack

ABSTRACT

One of the most commonly used two-factor user authentication mechanisms nowadays is based on smart-card and password. A scheme of this type is called a *smart-card-based password authentication scheme*. The core feature of such a scheme is to enforce two-factor authentication in the sense that the client must *have* the smart-card and *know* the password in order to gain access to the server. In this paper, we scrutinize the security requirements of this kind of schemes, and propose a new scheme and a generic construction framework for smart-card-based password authentication. We show that a secure password based key exchange protocol can be efficiently transformed to a smart-card-based password authentication scheme provided that there exist pseudorandom functions and target collision resistant hash functions. Our construction appears to be the first one with provable security. In addition, we show that two recently proposed schemes of this kind are insecure.

1. Introduction

Smart-card-based password authentication is one of the most convenient and commonly used *two-factor authentication* mechanisms. This technology has been widely deployed in various kinds of authentication applications which include remote host login, online banking, access control of restricted vaults, activation of security devices, and many more. A smart-card-based password authentication scheme involves a server S and a client A (with identity ID_A). At first, S securely issues a smart-card to A with the smart-card being personalized with respect to ID_A and an initial password. This phase is called the *registration phase* and is carried out only once for each client. Later on, A can access S in the *login-and-authentication phase*, and this phase can be carried out as many times as needed. However, in this phase, there could have various kinds of passive and active adversaries in the communication channel between A and S. They can eavesdrop messages and even modify, remove or insert messages into the channel. The security goal of the scheme in this phase is to ensure mutual authentication between A and S. In particular, the client is required to both *have* the smart-card and *know* the password in order to carry out the smart-card-based password authentication successfully with server S. In other words, the scheme should provide *two-factor authentication*.

There are some other requirements/properties that are desirable in practice. For example, A may want to *change password* from time to time. Conventionally, this requires A to interact with S and S has to maintain a password database for its clients. In this paper, we promote the idea of letting A change the password at will without interacting with or notifying S (while ensuring two-factor authentication), and also eliminating any password database at the server side. Below are the

* Corresponding author.
E-mail addresses: csyanggm@cs.cityu.edu.hk (G. Yang), duncan@cs.cityu.edu.hk (D.S. Wong), hxwang@ntu.edu.sg (H. Wang), deng@cs.cityu.edu.hk (X. Deng).

[1] The author was supported by a grant from CityU (Project No. 7001959).

0022-0000/$ – see front matter
doi:10.1016/j.jcss.2008.04.002

reasons. Most of the current systems require the server to maintain a database for the passwords or derived values of the passwords of its clients. The derived values of the passwords can be obtained by using a password-based KDF (key derivation function) which takes a password and a known random value called salt and apply a hash function or a block cipher for a number of iterations (standardized in PKCS [17] and IETF [8]). However, this approach not only introduces scalability problem to the server but also makes the systems suffer from disastrous loss when the server is compromised and the password database is stolen by adversaries.

Current systems also suffer from other potential security vulnerabilities. One prominent issue is security against *offline guessing attack* (also known as offline dictionary attack). The purpose of offline guessing attack is to compromise a client's password through exhaustive search of all possible password values. In a password-based setting, passwords are considered to be short and human memorizable, and the corresponding password space is so small that an adversary is able to enumerate all possible values in the space within some reasonable amount of time. For example, most of the ATM deployments use PINs (personal identification numbers) of only 4 to 6 digits long, so the password space has no more than one million possible values. Therefore, another security requirement for smart-card-based password authentication is security against offline guessing attack. In particular, compromising a client's smart-card should not allow an adversary to launch offline guessing attack against the client's password. In practice, the adversary may steal the smart-card and extract all the information stored in it through reverse engineering. This notion is reminiscent of password-based authentication protocols [13]. The difference is that for password-based authentication protocols, the focus is on preventing adversaries from getting any useful information about the password from the transcripts of protocol runs under the assumption that the two communicating parties are not compromised; while for smart-card-based password authentication schemes, we further require that the client's password should remain secure even after the client's smart-card is compromised.

1.1. Our results

In this paper, we contribute on the following three areas:

1. We propose a new set of security requirements for two-factor smart-card-based password mutual authentication. The new set is a refinement of some previously proposed requirement set, it not only eliminates the redundancies and ambiguities of the old requirement set, but also facilitates cryptanalysis due to its simplification. The new requirement set also associates with an adversarial model. The separation of requirement set and adversarial capabilities allows us to establish a systematic approach for constructing and proving secure smart-card-based password authentication schemes.
2. We show that two recently proposed schemes are insecure against their claimed security properties which have also been captured in the new requirement set.
3. We propose a new two-factor smart-card-based password mutual authentication scheme and also a generic construction framework. We show that a secure two-factor smart-card-based password mutual authentication scheme can be constructed by *transforming* a proven secure *one-factor* password based mutual authentication protocol (under some appropriate security model) provided that there exist pseudorandom functions and target collision resistant hash functions. The transformation is very efficient. It only adds several a few hash evaluations and one pseudorandom function evaluation.

1.2. Paper organization

In Section 2, we review some related work. In Section 3, we propose a new set of desirable properties and an adversarial model for smart-card-based password authentication and compare them with a recently proposed property set. In Section 4, we review a scheme proposed by Liao et al. [20] and show that the scheme is insecure against their security claims. The security analysis of another scheme proposed by Yoon and Yoo [29] is given in Appendix A. In Sections 5, 6 and 7, we propose a new scheme and also a generic construction framework that can be used to transform a proven secure password-based authentication protocol to a smart-card-based password authentication scheme. We conclude the paper in Section 8.

2. Related work

Since Lamport [18] introduced a remote user authentication scheme in 1981, there have been many smart-card-based password authentication schemes proposed (some recent ones are [7,20,28,29]). These schemes are aimed for different security goals and properties, and noticeably, there is no common set of desirable security properties that has been widely adopted for the construction of this type of schemes. Although the construction and security analysis of this type of schemes have a long history, recently proposed schemes are still having various security weaknesses being overlooked, and we can find many of these schemes broken shortly after they were first proposed [11,12,24,26,29].

In [20], Liao et al. made an attempt to consolidate a large set of desirable properties for smart-card-based password authentication schemes. Although we will see later that many of the properties in their proposed set are redundant, the set indeed comes close to capture the *two-factor authentication* of smart-card-based password authentication schemes. One of the properties requires that offline guessing attack should not be feasible even after a client's smart-card is stolen and compromised. Their attempt is to enforce the two-factor authentication in the sense that the client must have both the

smart-card and the password to gain access to the server. As noted in [20], it does not seem to have any protocol available that satisfies all of their properties. However, we find that Scott has a working paper [23] which has a smart-card-based password "key exchange" protocol that can be extended to provide explicit mutual authentication and satisfy all the properties given in [20]. Scott's construction requires some special non-supersingular elliptic curves to defend against offline guessing attack and may not be suitable for systems where Tate pairing operations are considered to be too expensive or infeasible to implement. In the next section, we will define a refined set of properties that not only captures the set of properties specified in [20], but also removes redundancies and ambiguities from their property set. We also point out that the property set proposed in [20] does not capture the exact two-factor authentication that a smart-card-based password authentication scheme should provide.

Liao et al. also proposed a new scheme in [20] and claimed that the scheme satisfies all the security properties specified in their paper. However, in this paper, we show that the scheme does not satisfy some of their properties. In particular, we describe an offline guessing attack to compromise a client's password once after the client's smart-card is compromised. We also show that their scheme may not provide server authentication even their proposed improvement is in place. Details are given in Section 4. In [29], Yoon and Yoo proposed another scheme. However, we find that their scheme is vulnerable to the offline guessing attack which is similar to the one described in Section 4. Details are given in Appendix A.

Systems using smart-card only. In [25], Shoup and Rubin proposed an extension of Bellare and Rogaway's model [5] for three-party key distribution protocols where smart-cards are used to store the long-term keys. In their system, each user is considered to have a smart-card storing the long-term key of the user. When using the system, the smart-card is attached to a hosting machine and the machine is communicating with another machine which has another user's smart-card attached. The smart-cards are used to prevent adversaries from getting the long-term keys, for example, by compromising the hosting machines. However, it is assumed in their model that once an adversary has compromised the smart-card of a user, the adversary has compromised the user. Therefore, the model still falls in the category of one-factor authentication. That is, once the smart-card is compromised, the adversary can impersonate the corresponding user. In practice, we can do better than this when smart-cards are used. In this paper, we propose a scheme (and also a construction framework) which do not allow an adversary to impersonate a user unless *both* the smart-card and the password of the user are known to the adversary. In addition, compromising the smart-card does not help the adversary get the password and vice versa. This is the core feature of two-factor authentication, namely, the user has to show that he *has* the smart-card and he *knows* the password.

Systems using password only. Because of its practicability, password-based authentication (and key exchange) protocols have been well-studied in the past [1,2,4,10,15]. As discussed above, passwords are of low entropy, therefore, one of the most important security requirements for this kind of protocols is to resist against password guessing (or dictionary) attacks. Although it is possible to thwart offline guessing attack, systems using password only are subject to online guessing attack. In this attack, the adversary simply selects a trial password from the password space and follow the operations honestly according to the protocol. The success probability of the adversary is inversely proportional to the size of the dictionary space. In this paper, we show that by making use of smart-card, we can efficiently "upgrade" the security of any proven secure password based authentication protocol (under some appropriate security model) from password level to cryptographic key level. Moreover, even if the password is compromised, the upgraded protocol remains secure provided that the smart-card is not compromised.

3. Security requirements

As introduced in Section 1, there are two phases and one activity in a smart-card-based password authentication system. The two phases are *registration phase* and *login-and-authentication phase*, and the activity is called *password-changing activity*. In the registration phase, an authenticated and secure environment is assumed to present, and all parties are assumed to be honest and perform exactly according to the scheme specification. After this phase is completed, the client is said to be *registered*. In the login-and-authentication phase, the communication channel between server S and a registered client A is no longer considered to be secure. Both passive and active adversaries are present and their objective is to compromise the scheme's primary security goal, that is, mutual authentication between S and A. During the password-changing activity, a registered client A can change the password and update the smart-card accordingly without any interaction with S. This helps alleviate scalability problem and also facilitate user friendliness. In the following, we describe what we want a secure smart-card-based password authentication scheme to achieve (i.e. security goals and desirable properties) and what the capabilities of the adversary are (i.e. an adversarial model).

3.1. Desirable properties and adversarial model

Below are the five desirable properties for a smart-card-based password authentication scheme.

1. (*Client Authentication*) The server is sure that the communicating party is indeed the registered client that claims to be at the end of the protocol.
2. (*Server Authentication*) The client is sure that the communicating party is indeed the server S at the end of the protocol.
3. (*Server Knows No Password*) S should not get any information of the password of a registered client or anything derived from the password.
4. (*Freedom of Password Change*) A client's password can freely be changed by the client without any interaction with server S. S can be totally unaware of the change of the client's password.
5. (*Short Password*) We consider a human-memorizable password to be a value in the password space.

3.1.1. Adversarial model

Consider an adversary $\mathcal{A}$ who has the full control of the communication channel between the server S and any of the *registered* clients. $\mathcal{A}$ can obtain all the messages transmitted between the server S and a registered client; $\mathcal{A}$ can also modify or block transmitting messages; and even make up then send fake messages to any entity in the system while claiming that the messages are from another entity (i.e. impersonation). To simulate *insider attack*, that is, the adversary *is* a malicious but real client, we also allow $\mathcal{A}$ to know the passwords and the information stored in the smart-cards of all the clients except those of a client who is under attack from $\mathcal{A}$. In addition, we also allow $\mathcal{A}$ to *either* compromise the password *or* the smart-card of the client under attack, but not both. However, $\mathcal{A}$ is not allowed to compromise S.

In Section 5, we formalize the adversarial model above from a model for one-way password-based authentication due to Halevi and Krawczyk [10] and a model for mutual authentication (and key exchange) due to Canetti and Krawczyk [6]. We will also see that in general, the adversarial model can be formalized from any appropriate model for password-based mutual authentication by separating the case when $\mathcal{A}$ has known the password of the client (but not the smart-card) and the case when $\mathcal{A}$ has compromised the smart-card of the client (but not the password) into two disjoint cases.

3.1.2. Discussions regarding the desirable properties and adversarial model

One should note that property (3) does not imply property (4). It is always possible to construct a scheme such that the server does not have any information of a client's password while the client cannot change the password either once after registration.

One remark is that in this paper, we do not make assumption on the existence of any special security features supported by the smart-cards. Instead, we simply consider a smart-card to be a memory card with an embedded micro-processor for performing required operations specified in a scheme. As Kocher et al. [16] and Messerges et al. [21] pointed out, all existing smart-cards cannot prevent the information stored in them from being extracted, for example, by monitoring their power consumption. Some other reverse engineering techniques are also available for extracting information from smart-cards. Hence, we put aside any special security features that could be supported by a smart-card, and simply assume that once a smart-card is stolen by an adversary, all the information stored in it are known to the adversary.

3.2. Comparison with the ten properties proposed by Liao et al.

In [20], Liao et al. proposed a set of requirements for smart-card-based password authentication schemes. In their set, there are ten requirements (**R1**–**R10**) and all of them are claimed to be independent. However, it is easy to see that all of them have been included in our property set in Section 3.1 except **R8**, which says "The scheme must be efficient and practical." For this issue, it does not seem to be measurable without referring to any other comparable schemes, therefore, we choose to not list it in our property set. In [20], the authors define two-factor security in **R10**, which says "The password cannot be broken by guessing attack even if the smart-card is lost." However, this is not the exact definition of two-factor security, it fails to capture another aspect of two-factor security, that is, knowing the password alone should not allow the adversary to compromise the mutual authentication between the client and the server.

4. Offline guessing attack against a smart-card-based password authentication scheme

We now show that the scheme proposed by Liao et al. [20] is insecure with respect to their claimed security properties, this also implies that their scheme is insecure in the new set of properties and adversarial model defined above. In Appendix A, we show that another scheme recently proposed by Yoon and Yoo [29] is also insecure.

Here are the notations that we will use for describing Liao et al.'s scheme. Let p be a 1024-bit prime. Let g be a generator of $\mathbb{Z}_p^*$. The server S chooses a secret key x. In [20], the authors did not specify the length of x, however, in order to prevent brute-force search, we assume x to be a random string of at least 160 bits long. Let h be a hash function (e.g. SHA-256) and $a\|b$ denote the concatenation of a and b.

Registration phase: Server S issues a smart-card to a client A as follows.

1. A arbitrarily chooses a *unique* identity ID_A and password PW_A. PW_A is a short password that is appropriate for memorization. A then calculates $h(PW_A)$ and sends $(ID_A, h(PW_A))$ to S.
2. S calculates $B = g^{h(x\|ID_A)+h(PW_A)} \bmod p$ and issues A a smart-card which has (ID_A, B, p, g) in it.

Login-and-authentication phase:[2] A attaches the smart-card to an input device and keys in ID_A and PW_A. Afterwards, S and A (the smart-card) carry out the following steps.

1. A sends a login request to S.
2. On receiving the login request, S calculates $B'' = g^{h(x\|ID_A)R} \bmod p$ where $R \in \mathbb{Z}_p^*$ is a random number, and sends $h(B'')$ and R to A.
3. Upon receiving the message from S, A calculates $B' = (Bg^{-h(PW_A)})^R \bmod p$ and checks if $h(B'') = h(B')$. If they are not equal, S is rejected. Otherwise, A calculates $C = h(T\|B')$ where T is a timestamp, and sends (ID_A, C, T) to S.
4. Let T' be the time when S receives (ID_A, C, T). S validates A using the following steps.
   (a) S checks if ID_A is in the correct format.[3] If it is incorrect, S rejects.
   (b) Otherwise, S compares T with T'. If $T' - T \geqslant \Delta T$, S rejects, where ΔT is the legal time interval for transmission delay.
   (c) S then computes $C' = h(T\|B'')$ and checks if $C = C'$. If they are not equal, S rejects. Otherwise, S accepts.

We skip the review of their password-changing activity as it is not needed in our attacks below.

Password-changing activity: To change a password, the client A carries out the following steps.

1. Select a new password PW'_A.
2. Compute $Y = g^{h(PW'_A)} \bmod p$.
3. Compute $\beta = Bg^{-h(PW_A)}Y \bmod p$, where PW_A is the original password of A.
4. Replace B with β in the smart-card.

In [20], it is claimed that the scheme above satisfies all of their ten properties reviewed in Section 3.2. However, in the following, we show that the scheme is vulnerable to offline guessing attack once the client's smart-card is compromised.

4.1. Offline guessing attack

4.1.1. Malicious user offline guessing attack

In [20], the scheme above is claimed to be secure against offline guessing attack even if the client's smart-card is compromised. In the following, we show that this is not true. Suppose client A's smart-card is compromised by an adversary $\mathcal{A}$. $\mathcal{A}$ can carry out the offline guessing attack as follows.

1. $\mathcal{A}$ impersonates A and sends a login request to S.
2. S calculates $B'' = g^{h(x\|ID_A)R} \bmod p$ and sends back $(h(B''), R)$.
3. $\mathcal{A}$ then carries out offline guessing attack by checking if

$$h(B'') = h\left(\left(Bg^{-h(PW_A^*)}\right)^R \bmod p\right)$$

   for each trial password PW_A^* (i.e. $\mathcal{A}$'s guess of PW_A).

Note that after $\mathcal{A}$ receives the message from S in step (2), $\mathcal{A}$ does not need to provide any response to S and therefore S does not know whether the communicating party is launching an attack or simply the message sent by S is lost during transmission. This makes the guessing attack described above difficult to detect. Also notice that if $\mathcal{A}$ possesses a past communication transcript Trans between A and S, $\mathcal{A}$ can perform the offline guessing attack directly without interacting with S. In case the current password is not the old one involved in Trans, by performing this attack, $\mathcal{A}$ will retrieve the current password instead of the old one.

4.1.2. Malicious server offline guessing attack

The scheme is also vulnerable to malicious server offline guessing attack. Since in the registration phase, user A will pass the value $h(PW_A)$ to the server. It is obvious that the server can retrieve PW_A easily by performing offline guessing attack. Although the authors of [20] provided some arguments for this issue, we still believe this is undesirable. It also violates property (3) in our property set.

4.2. Impersonation attack

The scheme cannot provide server authentication either. An adversary can impersonate the server S by simply replaying a previously intercepted message $(h(B''), R)$. In [20], the authors have already realized this and suggested to thwart this

[2] In [20], the authors divide the *login-and-authentication phase* into two parts: *login phase* and *authentication phase*. However, we put them together as other protocols do in the literature.

[3] In [20], the format of identity ID_A was not given. We hereby assume that there is some pre-defined format for all the identities used in their system.

replay attack by having S add another timestamp without providing a concrete construction. However, if this timestamp is added improperly, reflection attack will work and an adversary can impersonate a client without knowing the client's password. Consider the additional timestamp T' is added as below in the login-and-authentication phase.

...

2. On receiving the login request, S calculates $B'' = g^{h(x\|ID_A)R} \bmod p$ where $R \in \mathbb{Z}_p^*$ is a random number, and sends $C' = h(T'\|B'')$, T' and R to A.

...

If T' is placed in C' as shown above, an adversary $\mathcal{A}$ can impersonate A by simply sending ID_A, C', T' back to S in step 3.

5. A new two-factor mutual authentication scheme

In this section, we propose a new two-factor smart-card-based password mutual authentication scheme. We prove its security and show that it satisfies all the properties we described in Section 3. This new scheme can also be considered as a generic construction framework for two-factor smart-card-based password mutual authentication scheme. It can be used to transform any password-based mutual authentication protocols to two-factor smart-card-based password mutual authentication protocols. The significance of this work is that we can now design provably secure smart-card-based password authentication schemes in a systematic way by making use of previous results on password-based protocols.

Schemes constructed in this framework may also be chosen to have session keys established, which are generally useful for target applications.

Overview

In the two-factor security, it is assumed that the password and the smart-card cannot be both compromised. This leaves three other cases that need to be considered: (1) neither the password nor the smart-card is compromised; (2) the password is leaked while the smart-card remains secure; (3) the smart-card is compromised but the password remains secure. It is obvious that security under case (1) can be ensured if security under either case (2) or case (3) is guaranteed. And our goal is to achieve security under both case (2) and case (3). In other words, compromising one factor should not affect the other.

(*A Simple But Limited Two-Factor Authentication Protocol.*) If we do not consider properties (3) and (4) defined in Section 3.1, a secure two-factor authentication protocol can readily be constructed in the following way. Each client shares a long-term symmetric key K and a password PW with the server S, the server saves all the symmetric keys and passwords in two different tables, the client uses a smart-card to store the long-term symmetric key and remembers the password in mind. When the client connects to the server, the client first carries out a secure symmetric-key based authentication protocol π_1 with S using K, if π_1 fails, the authentication also fails, otherwise, the client runs a secure password-based authentication protocol π_2 with S using PW. The authentication is successful if both π_1 and π_2 succeed. We assume π_1 and π_2 are independent.

The above protocol is a trivial solution to two-factor authentication, but it fails to achieve properties (3) and (4) defined in Section 3.1. Our goal is to design a secure two-factor authentication protocol which not only satisfies all the properties defined in Section 3.1, but also achieves a better performance than the above trivial solution.

Our solution is built systematically by following the steps below.

1. We first choose a password-based *one-way* authentication protocol. For provable security and simplicity, we have chosen Halevi and Krawczyk's protocol [10] in this paper. But it is important to notice that one may choose to start with some other password-based one-way authentication protocol.
2. We then transform the password-based one-way authentication protocol to a password-based *mutual* authentication (and key exchange) protocol.
3. Finally, we "*upgrade*" the protocol to a two-factor smart-card-based password mutual authentication and key exchange protocol. Notice that the final protocol also establishes session keys for securing/authenticating the subsequent communications of the sessions.

Below are the details.

5.1. A password-based one-way authentication protocol

In [10], Halevi and Krawczyk proposed a password-based one-way authentication protocol and defined a security model for this type of protocols. Informally, the definition of security requires that the "best" possible strategy for the adversary to compromise user authentication is online guessing attack, which can be thwarted in practice by limiting the number of consecutive authentication failures that each user is allowed. The security model is defined by a game in which players are client A, server S and adversary $\mathcal{A}$. The game is parameterized by a security parameter k and a public dictionary $\mathcal{D}$ which contains all possible passwords. The game proceeds as follows.

Halevi–Krawczyk security game for password-based one-way authentication

Set-up phase: S chooses its cryptographic keys and publishes its public keys. A then uniformly[4] picks a password PW from $\mathcal{D}$ and gives it to S while keeping it secret from $\mathcal{A}$. $\mathcal{A}$ can also register clients with S at any time (before, during, or after the set-up phase) by picking any pair of identity A' and password PW' (provided that $A' \neq A$ and $PW' \in \mathcal{D}$) and giving PW' to S.

Game running phase: $\mathcal{A}$ has full control over all the clients it created, as well as the communication between A and S. That is, A and S can only communicate through $\mathcal{A}$, every message that A and S send goes to $\mathcal{A}$, and every message they receive comes from $\mathcal{A}$. $\mathcal{A}$ may choose to forward messages faithfully, but may also insert, drop, or modify messages. $\mathcal{A}$ can send special "prompt" messages to the parties at any time, causing them to start new authentication sessions (in particular, several simultaneous sessions by the same or different parties are possible). Each session will have a unique session identifier. This game is run until $\mathcal{A}$ decides to halt.

Outputs of parties: For capturing the security requirements, A and S will record events related to the security of authentication by giving some special outputs. A outputs a pair (S, sid) whenever it authenticates itself to server S under session identifier sid. S outputs (A, sid) whenever a successful authentication by A is completed during session sid. If an attempt to authenticate (alleged) A in session sid fails, S outputs $(A, sid, \perp)$. This is needed so that the "number of failed authentication attempts" can be counted.

Besides the game above, some terminologies are defined as follows.

- An authentication protocol π is said to be *syntactically correct* if whenever all the messages between A and S in session sid are passed unchanged, then S and A output (A, sid) and (S, sid), respectively.
- An event in which S outputs (A, sid) but A has never output a pair (S, sid) is called a *successful impersonation* (here we assume that the last message is sent by A and A outputs (S, sid) only after the last message is sent, while S outputs (A, sid) only after receiving the last message sent by A). An event in which S outputs $(A, sid, \perp)$ is called an *authentication failure*. An event in which S outputs a pair (A', sid) after already outputting some other pair (A'', sid) in the past is called a *successful replay*. (Here A' and A'' are arbitrary clients, and sid is the same in both pairs.) All the events above are referred to as *active impersonation attempts*.
- An (ℓ, m)-run of the game is a run with at most m active impersonation attempts, and A outputs at most ℓ pairs of (S, sid).[5] The adversary $\mathcal{A}$ achieves an (ℓ, m)-win if in an (ℓ, m)-run of the game, there is at least one successful impersonation or replay event.

Definition 1. Let $\epsilon(\cdot,\cdot,\cdot)$ be a positive real function and π a syntactically correct authentication protocol. We say that π ensures one-way password-based authentication up to ϵ, if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, any finite dictionary $\mathcal{D}$, any sufficiently large security parameter k, any polynomial ℓ, and any integer $m < |\mathcal{D}|$, we have

$$\Pr\big[(\ell, m)\text{-win}\big] \leqslant \frac{m}{|\mathcal{D}|} + \epsilon(k, \ell, m)$$

where the probability is taken over the random coins of S, A and $\mathcal{A}$ in an (ℓ, m)-run of the game.

The security goal is to have $\epsilon(k, \ell, m)$ be a negligible function in k.

Below is the password-based one-way authentication protocol proposed in [10]. $\mathrm{ENC}_{\mathrm{PK}_S}$ denotes a public-key encryption function under S's public key PK_S.

1. A sends a request with (A, sid) to S.
2. Upon receipt of the request, S picks a nonce n and sends (S, sid, n) to A.
3. On receiving the reply, A computes $c = \mathrm{ENC}_{\mathrm{PK}_S}(PW, A, S, sid, n)$. A then sends (A, sid, c) to S.
4. Upon receipt of (A, sid, c), S decrypts c and do the verification.

$$A \to S : A, sid,$$

$$A \leftarrow S : S, sid, n,$$

$$A \to S : A, sid, c = \mathrm{ENC}_{\mathrm{PK}_S}(PW, A, S, sid, n).$$

It is shown in [10] that the winning probability of the adversary to break the above protocol is bounded by $\frac{m}{|\mathcal{D}|} + m \cdot \ell \cdot \epsilon_{pke}$ where ϵ_{pke} is the upper bound of the advantage of any PPT adversary in the IND-CVA game [10] for public key encryption

[4] Security analysis can be extended to the non-uniform case by modifying Definition 1 accordingly.

[5] In the real world, this can be treated as the requirement that A should change the password once every ℓ times of login attempts.

schemes. IND-CVA is a weaker notion than IND-CCA [22], hence, ϵ_{pke} is negligible if the public-key encryption scheme in use is IND-CCA secure.

Next, we transform the protocol above to a password-based *mutual* authentication and key exchange (PWAKE) protocol.

5.2. A password-based mutual authentication and key exchange (PWAKE) protocol

We first describe the PWAKE protocol and then explain how it is transformed from the password-based one-way authentication protocol. Let G be a group of prime order q and g a generator. Let $(\mathrm{PK}_S, \mathrm{SK}_S)$ and $(\mathrm{PK}'_S, \mathrm{SK}'_S)$ denote the encryption and signature key pairs of the server S. User A has a password PW_A which is shared with S. Let SIG be a signing algorithm. The PWAKE protocol is as follows.

$$A \to S : A, sid, g^x,$$

$$A \leftarrow S : S, sid, g^y, \mathrm{SIG}_{\mathrm{SK}'_S}\left(S, A, sid, g^x, g^y\right),$$

$$A \to S : A, sid, c = \mathrm{ENC}_{\mathrm{PK}_S}\left(PW_A, A, S, sid, g^x, g^y\right).$$

The session key is calculated as $\sigma = g^{xy}$. Here g^x and g^y also play the role of nonces. If the session key is not needed and only mutual authentication is required, one can replace g^x and g^y with two random numbers. The final protocol will then be a password-based mutual authentication protocol.

5.2.1. Security analysis and conversion

The security of the protocol above follows the framework due to Canetti and Krawczyk [6], that allows a mutually authenticated key exchange protocol to be constructed from two *authenticators* (as defined in [6]) and one key exchange protocol which only requires to be secure against *passive adversaries* (e.g. the plain Diffie–Hellman key exchange protocol).

In our protocol above, one authenticator is the Halevi–Krawczyk password-based one-way authentication protocol,[6] and the other authenticator is the signature-based one due to Bellare et al. [3]:

$$P_i \to P_j : m, sid,$$

$$P_i \leftarrow P_j : m, sid, N_j,$$

$$P_i \to P_j : m, sid, \mathrm{SIG}_{P_i}(m, sid, N_j, P_j).$$

where $N_j \in_R \{0,1\}^k$ is a nonce, SIG_{P_i} is the signing algorithm of P_i. The signature scheme is assumed to be existentially unforgeable against chosen message attacks [9].

By following the Canetti–Krawczyk model [6], we compile the plain Diffie–Hellman key exchange protocol (which has been proven to provide session key security against passive adversaries [6]) to a PWAKE protocol, using the Halevi–Krawczyk password-based one-way authenticator for user authentication and the signature-based authenticator above for server authentication. In other words, our PWAKE protocol is proven secure under the Canetti–Krawczyk model [6] which has captured simultaneous sessions, interleaving attacks and perfect forward secrecy (for session keys). We refer readers to the full paper of [6] for details.

In the following, we transform this PWAKE protocol to a two-factor smart-card-based password mutual authentication and key exchange protocol.

5.3. A two-factor smart-card-based password mutual authentication and key exchange protocol

Notations: Let G, g, q be the group parameters defined as above. Besides the encryption and signature key pairs $(\mathrm{PK}_S, \mathrm{SK}_S)$ and $(\mathrm{PK}'_S, \mathrm{SK}'_S)$, the server S also maintains a long-term secret x which is a random string of length k. Let $H : \{0,1\}^* \to \{0,1\}^k$ denote a target collision resistant hash function and $PRF_K : \{0,1\}^k \to \{0,1\}^k$ a pseudorandom function keyed by K. Also let $H'\{0,1\}^* \to \{0,1\}^k$ denote a hash function which preserves the entropy of its input. In our scheme, H' takes a password PW as input, and we may simply generate $H'(PW)$ by appending sufficiently many 0's at the end the password.

Registration phase: Server S issues a smart-card to client A as follows.

1. A arbitrarily chooses a unique identity ID_A and sends it to S.
2. S calculates $B = PRF_x(H(ID_A)) \oplus H'(PW_0)$ where PW_0 is the initial password (e.g. a default password such as a string of all '0').
3. S issues A a smart-card which contains $\mathrm{PK}_S, \mathrm{PK}'_S, ID_A, B, p, g, q$. In practice, we can "burn" all these parameters except B in the read-only memory of the smart-card when the smart-card is manufactured.

[6] Canetti and Krawczyk were the first ones to point out (in Section 5.4 of the full paper of [6]) that the Halevi–Krawczyk password-based one-way authentication protocol is a password-based authenticator.

4. On receiving the smart-card, A changes the password immediately by performing the password-changing activity (described below).

Login-and-authentication phase: A attaches the smart-card to an input device, and then keys PW_A. The smart-card retrieves the value $LPW_A = B \oplus H'(PW_A)$. A (actually performed by the client's smart-card) and S then use LPW_A as the password to perform the PWAKE protocol.

$$A \to S : A, sid, g^{\hat{x}},$$

$$A \leftarrow S : S, sid, g^{\hat{y}}, \mathrm{SIG}_{\mathrm{SK}'_S}\left(S, A, sid, g^{\hat{x}}, g^{\hat{y}}\right),$$

$$A \to S : A, sid, c = \mathrm{ENC}_{\mathrm{PK}_S}\left(LPW_A, A, S, sid, g^{\hat{x}}, g^{\hat{y}}\right).$$

Password-changing activity: If A wants to change the password, A carries out the following steps.

1. Select a new password PW'_A.
2. Compute $Z = B \oplus H'(PW_A) \oplus H'(PW'_A)$, where PW_A is the old password.
3. Replace B with Z in the smart-card.

Remark. The "password" used in the login-and-authentication phase is *LPW*, instead of the real password PW_A. Note that S can compute the value of *LPW* once after receiving ID_A. Hence it does not violate property (3) (Server Knows No Password) in Section 3. From the password-changing activity above, it is obvious that the scheme also satisfies property (4) (Freedom of Password Change).

6. Security analysis

The security analysis is done under the framework of Canetti and Krawczyk [6], that is, we should show that our scheme described in Section 5.3 has the authenticator for server authentication and the authenticator for client authentication, and also has a key exchange protocol secure against passive adversaries. As inherited from the original PWAKE protocol, it is obvious that the key exchange protocol is the plain Diffie–Hellman key exchange protocol which has been shown secure against passive adversaries [6]. In addition, the authenticator for server authentication is the signature-based authenticator [3]. Regarding the authenticator for client authentication, however, we need to consider its security under case (2) and case (3) separately, as explained in the overview (Section 5). Note that this is not needed for server authentication because the server side authenticator remains unchanged for both of the cases.

For client side authenticator, if we can show that in each of the cases, if the authenticator is a password-based one-way authentication protocol in the Halevi–Krawczyk security game (reviewed on page 1166), then we can be sure that it is a secure authenticator [6] (please also refer to footnote 6 on page 1167). Therefore, in the following, we only need to show that the authenticator is a password-based one-way authentication protocol under the Halevi–Krawczyk security game for each of the two cases.

Case (2) security. If the smart-card is not compromised (while the password is leaked), our proposed scheme still provides cryptographic key level protection for mutual authentication. We analyze client authentication in the Halevi–Krawczyk security game for password-based one-way authentication (page 1166). To mimic this case, and without loss of generality, we assume that after client A gets the smart-card, she uses the default password without changing it, and the adversary is given the default password of A. The remaining game is the same as in the Halevi–Krawczyk game.

Lemma 1. *If the smart-card is not compromised, $PRF_K(\cdot)$ is replaced by an ideal random function $RAND(\cdot)$, and H is a target collision resistant hash function, then the adversary has only a negligible success probability in the Halevi–Krawczyk security game for password-based one-way authentication.*

Proof. Let Col be the event that $\mathcal{A}$ successfully finds an identity ID_B such that $ID_B \neq ID_A$ but $H(ID_B) = H(ID_A)$. If Col happens, $\mathcal{A}$ can successfully get the value of LPW_A and thus break the client authentication. However, since H is target collision resistant, Col happens with only negligible probability ϵ_{tch}.

For event $\overline{\mathrm{Col}}$, that is, the collision above does not happen, the client authentication mechanism is identical to the Halevi–Krawczyk one-way authentication protocol [10]. Hence for the event $\overline{\mathrm{Col}}$, the game will proceed in the same way as the original Halevi–Krawczyk's one except that LPW_A is randomly chosen (as *RAND* is an ideal random function) from a potentially larger space $\{0,1\}^k$.

Hence by combining the winning probability for $\mathcal{A}$ for both of the events, the probability of an (ℓ, m)-win is bounded by

$$\begin{aligned}\Pr\big[(\ell,m)\text{-win}\big] &= (1-\epsilon_{tch})\left(\frac{m}{2^k}+m\cdot\ell\cdot\epsilon_{pke}\right)+\epsilon_{tch}\cdot 1\\ &= \frac{m}{2^k}+m\cdot\ell\cdot\epsilon_{pke}+\left(1-\left(\frac{m}{2^k}+m\cdot\ell\cdot\epsilon_{pke}\right)\right)\epsilon_{tch}\\ &\leqslant \frac{m}{2^k}+m\cdot\ell\cdot\epsilon_{pke}+\epsilon_{tch}\end{aligned}$$

where the winning bound for event $\overline{\mathsf{Col}}$ is obtained directly from [10]. □

Theorem 1. *If the smart-card is not compromised, and $PRF_K(\cdot)$ is a pseudorandom function, then the adversary has only a negligible success probability in the Halevi–Krawczyk security game.*

Proof. The proof is by contradiction. Suppose there exists a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ which has probability ϵ to win in an (ℓ,m)-run of the Halevi–Krawczyk game. We construct another PPT adversary $\mathcal{D}$ which breaks the pseudorandom function.

Adversary $\mathcal{D}$ is given access to an oracle $\mathcal{O}$ which is either $PRF_K(\cdot)$ (with probability 1/2) or an ideal random function $RAND(\cdot)$ (with probability 1/2). $\mathcal{D}$ can adaptively query an arbitrarily chosen string $x\in\{0,1\}^k$ to $\mathcal{O}$ and get the output which is either $PRF_K(x)$ or a random string uniformly selected from $\{0,1\}^k$ (i.e. $RAND(x)$). After performing polynomially many queries, $\mathcal{D}$ finally makes a decision whether the oracle $\mathcal{O}$ is the function $PRF_K(\cdot)$ or the ideal random function $RAND(\cdot)$. $\mathcal{D}$ wins the game if the decision is correct.

To do this, $\mathcal{D}$ runs a simulation of the Halevi–Krawczyk game and plays the role of the server S. Suppose the encryption and signature key pairs generated by $\mathcal{D}$ is $(\mathrm{PK}_S,\mathrm{SK}_S)$ and $(\mathrm{PK}'_S,\mathrm{SK}'_S)$. $\mathcal{D}$ invokes adversary $\mathcal{A}$ in the game. In the registration phase for client A with identity ID_A, $\mathcal{D}$ calculates $h=H(ID_A)$ and enquiries oracle $\mathcal{O}$ with input h. $\mathcal{D}$ sets the return value from $\mathcal{O}$ to LPW_A and passes it to A (in the form of $LPW\oplus H'(PW_0)$ where PW_0 is a k-bit default value).

$\mathcal{D}$ then runs the game until $\mathcal{A}$ halts, thus $\mathcal{D}$ is in polynomial time. If an (ℓ,m)-win occurs in the game, $\mathcal{D}$ makes a decision that the oracle is $PRF_K(\cdot)$. Otherwise $\mathcal{D}$ chooses the ideal random function $RAND$ as its decision. Then we have

$$\begin{aligned}&\Pr[\mathcal{D}\text{ wins }]\\ &\quad=\Pr[\mathcal{D}\text{ outputs }PRF_K|\mathcal{O}=PRF_K]\Pr[\mathcal{O}=PRF_K]\\ &\qquad+\Pr[\mathcal{D}\text{ outputs }RAND|\mathcal{O}=RAND]\Pr[\mathcal{O}=RAND]\\ &\quad=\frac{1}{2}\Pr[\mathcal{D}\text{ outputs }PRF_K|\mathcal{O}=PRF_K]+\frac{1}{2}\Pr[\mathcal{D}\text{ outputs }RAND|\mathcal{O}=RAND]\\ &\quad=\frac{1}{2}\Pr[\mathcal{D}\text{ outputs }PRF_K|\mathcal{O}=PRF_K]+\frac{1}{2}\big(1-\Pr[\mathcal{D}\text{ outputs }PRF_K|\mathcal{O}=RAND]\big)\\ &\quad=\frac{1}{2}\big(\Pr\big[(\ell,m)\text{-win}\big|\mathcal{O}=PRF_K\big]\big)+\frac{1}{2}\big(1-\Pr\big[(\ell,m)\text{-win}\big|\mathcal{O}=RAND\big]\big)\\ &\quad=\frac{1}{2}+\frac{1}{2}\big(\Pr\big[(\ell,m)\text{-win}\big|\mathcal{O}=PRF_K\big]-\Pr\big[(\ell,m)\text{-win}\big|\mathcal{O}=RAND\big]\big).\end{aligned}$$

According to Lemma 1,

$$\Pr\big[(\ell,m)\text{-win}\big|\mathcal{O}=RAND\big]\leqslant\frac{m}{2^k}+m\cdot\ell\cdot\epsilon_{pke}+\epsilon_{tch}.$$

Then we have

$$\Pr[\mathcal{D}\text{ wins }]\geqslant\frac{1}{2}+\frac{1}{2}\left(\epsilon-\frac{m}{2^k}-m\cdot\ell\cdot\epsilon_{pke}-\epsilon_{tch}\right).\qquad\square$$

Case (3) security. If the smart-card is compromised while the password remains secure, there is no security "upgrade" when compared with the original one-factor password-based mutual authentication protocol (Section 5.2). In the security analysis below we assume that after client A gets the smart-card, she chooses a random password from the dictionary D and performs a password changing activity immediately. After that, the value B inside the smart-card is given to the adversary. The remaining game is the same as in the Halevi–Krawczyk model.

Theorem 2. *If the smart-card is compromised but the password remains secure, the proposed scheme provides the same security level as the original password scheme.*

Proof. The proof is similar to that of Theorem 1, suppose we use $H'(PW)=PW\|0^{k-|PW|}$ where $\|$ denotes concatenation and $|PW|$ denotes the length of PW. Once the smart-card is compromised, then the value B is known to the adversary, hence, by the equation $LPW=B\oplus(PW\|0^{k-|PW|})$, the adversary obtains the last $k-|PW|$ bits of LPW.

Similar to the proof of Lemma 1, if we consider $PRF_K(\cdot)$ to be replaced by an ideal random function $RAND(\cdot)$, then we have

$$\Pr\big[(\ell, m)\text{-win}\big] \leqslant \frac{m}{|D|} + m \cdot \ell \cdot \epsilon_{pke} + \epsilon_{tch}.$$

And similar to the proof of Theorem 1, if we replace the ideal random function $RAND(\cdot)$ by a pseudo-random function $PRF_K(\cdot)$, we have

$$\Pr\big[(\ell, m)\text{-win}\big] \leqslant \frac{m}{|D|} + m \cdot \ell \cdot \epsilon_{pke} + \epsilon_{tch} + 2\epsilon_{prf} - 1,$$

where ϵ_{prf} denotes the upper bound of the winning probability of any PPT adversary in the pseudo-random game. □

7. A generic construction framework

Up to this point, readers may have already realized that a smart-card-based password authentication scheme can readily be built from a proven secure password-based mutual authentication protocol by applying the *upgrading technique* of Section 5.3. The resulting scheme will then be secure under a model similar to the security model for the original password-based protocol, but extended according to the discussions in Section 5.3.

For example, we may choose an efficient password-based mutual authentication (and key exchange) protocol, such as [14,15], then we "upgrade" it to an efficient smart-card-based password authentication scheme using the technique described in Section 5.3. Both of the protocols in [14,15] are proven secure without random oracle, and our upgrading technique does not rely on random oracle either. The "upgraded" smart-card-based scheme will then be secure with security statements similar to that of Theorems 1 and 2 (but now in the corresponding model of the original password-based authentication protocol). We refer readers to [13] for other examples of password-based mutual authentication (and key exchange) protocols.

7.1. Efficiency

The "upgrading" technique proposed in Section 5.3 is very efficient. During the login-and-authentication phase, the smart-card only needs to carry out one hash function and one exclusive-or operation in addition to the operations incurred by the underlying password-based protocol. The generic construction framework allows us to choose a password-based protocol which is efficient enough when implemented on smart-cards.

7.2. A practical issue

In the description above, we consider the server S to maintain one single long-term secret x for communication with all the clients. As a result, the secrecy of x is utmost important because the security of the entire system essentially relies on the security of x. In practice, we can alleviate the damage caused to a system by using multiple values of x to partition the system, and in each partition, a randomly generated x is used by a disjoint set of clients. Each partition is to be handled by a distinct and independent server. Compromising one server will therefore only affect the security of the corresponding partition of clients rather than the entire system. Note that this partitioning method does not affect the fulfillment of any of the desirable properties for a secure smart-card based password authentication scheme proposed in Section 3. Another mechanism which can be used in conjunction with the mechanism above is to set each long-term secret x with a validity period. Usually, smart-cards are used such that they are valid only for a period of time. Hence for a different period of time, a fresh long-term secret x can be used.

8. Conclusion

Smart-card-based password authentication is one of the most convenient ways to provide two-factor authentication for the communication between a client and a server. In this paper, we defined a set of desirable properties for secure smart-card-based password authentication schemes. We provided evidence to support the need of each of the properties. We showed that a recently proposed scheme of this type due to Liao et al. [20] does not satisfy some of the properties and some of their security claims are incorrect. In particular, we showed that their protocol is vulnerable to offline guessing attack once the client's smart-card is compromised. Also, we showed that their protocol may not provide server authentication even the protocol is modified according to the specification of their rectified protocol. Moreover, we showed that another scheme proposed recently by Yoon and Yoo is also vulnerable to our offline guessing attack.

We proposed a new two-factor smart-card-based password mutual authentication scheme and showed that it satisfies all the desirable properties specified in this paper. In addition, we generalize the construction idea of our concrete scheme to a generic construction framework which allows us to efficiently convert a password-based mutual authentication protocol to a smart-card-based password authentication scheme.

Appendix A. Security analysis of Yoon–Yoo smart-card-based password authentication scheme

In [29], Yoon and Yoo presented several attacks against Wu–Chieu [27] and Lee–Lin–Chang [19] smart-card-based password authentication schemes. They also proposed a new scheme and claimed its security against offline guessing attack. In the following, we show that their scheme is actually vulnerable to an offline guessing attack similar to the one described in Section 4. We also show that their scheme does not satisfy another requirement in our property set described in Section 3.

The Yoon–Yoo scheme has the same group setup as that of Liao et al.'s reviewed in Section 4. Let x be the long-term secret key of the server S. In the following, we review the registration phase and login-and-authentication phase of their scheme. The login-and-authentication phase also has a session key generated.

Registration phase: S issues a smart-card to a client A as follows.

1. A arbitrarily chooses a unique identity ID_A and password PW_A. A then sends them to S.
2. S calculates $T = h(ID_A \| x) \bmod p$ and $B = T \oplus PW_A$. S then issues A a smart-card which has (ID_A, B, p, g) in it.

Login-and-authentication phase: A attaches the smart-card to an input device and keys in ID_A and PW_A. Afterwards, S and A (the smart-card) carry out the following steps.

1. A extracts T by computing $T = B \oplus PW_A$, randomly picks $c \in \mathbb{Z}_p^*$, and computes $C_1 = g^c \bmod p$. A then sends (ID_A, C_1) to S.
2. S checks if ID_A is in the correct format.[7] If not, S rejects. Otherwise, S computes $T = h(ID_A \| x)$, randomly pick $s \in \mathbb{Z}_p^*$, and computes $sk = C_1^s \bmod p$, $C_2 = g^s \bmod p$ and $C_3 = h(ID_A \| T \| sk \| C_1)$. Then S sends (C_2, C_3) back.
3. A computes $sk' = C_2^c \bmod p$ and $C_3' = h(ID_A \| T \| sk' \| C_1)$. Then A checks if $C_3' = C_3$. If they are not equal, S is rejected. Otherwise, A computes $C_4 = h(ID_A \| T \| sk' \| C_2)$ and sends it to S.
4. Upon receiving C_4, S computes $C_4' = h(ID_A \| T \| sk \| C_2)$ and checks if $C_4 = C_4'$. If they are equal, S accepts. Otherwise, S rejects.

A.1. Offline guessing attack

In [29], it is claimed that the scheme is secure against offline guessing attack even if A's smart-card is compromised. In the following, we show that this is not true. Suppose an adversary $\mathcal{A}$ has compromised A's smart-card. The following attack can be carried out by $\mathcal{A}$ for finding out A's password PW_A.

1. $\mathcal{A}$ impersonates A by choosing a random value $c \in \mathbb{Z}_p^*$ and then sending (ID_A, C_1) to S, where $C_1 = g^c \bmod p$.
2. On receiving (ID_A, C_1), S chooses a random $s \in \mathbb{Z}_p^*$ and computes $sk = C_1^s \bmod p$, $C_2 = g^s \bmod p$ and $C_3 = h(ID_A \| T \| sk \| C_1)$. Then S sends (C_2, C_3) back.
3. $\mathcal{A}$ then carries out offline guessing attack by checking if $C_3 = h(ID_A \| B \oplus PW_A^* \| C_2^c \bmod p \| C_1)$ for each trial password PW_A^* (i.e. $\mathcal{A}$'s guess of PW_A).

A.2. Other problems of the scheme

Besides the offline guessing attack described above, the Yoon–Yoo scheme has some other problems. When referring to the list the desirable properties in our property set described in Section 3, we notice that the Yoon–Yoo scheme does not achieve property (3) (Server Knows No Password). This is because in the registration phase, the client A sends the chosen password to S directly. Also, the length of passwords is not specified in [29], and the value of B is calculated as $h(ID, x) \oplus PW$, which may be modified to $h(ID, x) \oplus h(PW)$.

References

[1] Michel Abdalla, Pierre-Alain Fouque, David Pointcheval, Password-based authenticated key exchange in the three-party setting, in: Public Key Cryptography, 2005, pp. 65–84.
[2] Michel Abdalla, David Pointcheval, Simple password-based encrypted key exchange protocols, in: CT-RSA, 2005, pp. 191–208.
[3] M. Bellare, R. Canetti, H. Krawczyk, A modular approach to the design and analysis of authentication and key exchange protocols, in: Proc. 30th ACM Symp. on Theory of Computing, ACM, May 1998, pp. 419–428.
[4] M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, in: Proc. EUROCRYPT 2000, in: Lecture Notes in Comput. Sci., vol. 1807, Springer-Verlag, 2000.
[5] M. Bellare, P. Rogaway, Provably secure session key distribution—The three party case, in: Proc. 27th ACM Symp. on Theory of Computing, ACM, Las Vegas, 1995, pp. 57–66.
[6] R. Canetti, H. Krawczyk, Analysis of key-exchange protocols and their use for building secure channels, in: Proc. EUROCRYPT 2001, in: Lecture Notes in Comput. Sci., vol. 2045, Springer-Verlag, 2001, pp. 453–474, full paper available at http://eprint.iacr.org/2001/040/.

[7] The format of identity ID_A was not given in [29]. We hereby assume that there is some pre-defined format for all the identities used in their system.

[7] H.Y. Chien, J.K. Jan, Y.M. Tseng, An efficient and practical solution to remote authentication: Smart card, Comput. Secur. 21 (4) (2002) 372–375.
[8] T. Dierks, C. Allen, The TLS Protocol Version 1.0, IETF RFC 2246, January 1999.
[9] S. Goldwasser, S. Micali, R. Rivest, A digital signature scheme secure against adaptive chosen-message attack, SIAM J. Comput. 17 (2) (April 1988) 281–308.
[10] S. Halevi, H. Krawczyk, Public-key cryptography and password protocols, ACM Trans. Inf. Syst. Secur. 2 (3) (1999) 230–268.
[11] M.-S. Hwang, Cryptanalysis of remote login authentication scheme, Comput. Commun. 22 (8) (1999) 742–744.
[12] M.-S. Hwang, C.-C. Lee, Y.-L. Tang, An improvement of SPLICE/AS in WIDE against guessing attack, Internat. J. Inform. 12 (2) (2001) 297–302.
[13] IEEE, P1363.2/ D23: Standard specifications for password-based public key cryptographic techniques, available at http://grouper.ieee.org/groups/1363/passwdPK/draft.html, March 2006.
[14] S. Jiang, G. Gong, Password based key exchange with mutual authentication, in: 11th International Workshop on Selected Areas in Cryptography, SAC 2004, in: Lecture Notes in Comput. Sci., vol. 3357, Springer-Verlag, 2005, pp. 267–279.
[15] J. Katz, R. Ostrovsky, M. Yung, Efficient and secure authenticated key exchange using weak passwords, J. ACM, in press (pending revisions). A preliminary full version available at http://www.cs.umd.edu/~jkatz/papers/password.pdf.
[16] P. Kocher, J. Jaffe, B. Jun, Differential power analysis, in: Proc. CRYPTO 99, Springer-Verlag, 1999, pp. 388–397.
[17] RSA Laboratories, PKCS #5 v2.0: Password-Based Cryptography Standard, 1999.
[18] L. Lamport, Password authentication with insecure communication, Comm. ACM 24 (11) (November 1981) 770–771.
[19] C.C. Lee, C.H. Lin, C.C. Chang, An improved low computation cost user authentication scheme for mobile communication, in: Proc. 19th Advanced Information Networking and Applications, IEEE AINA '05, 2005, pp. 249–252.
[20] I-En Liao, Cheng-Chi Lee, Min-Shiang Hwang, A password authentication scheme over insecure networks, J. Comput. System Sci. 72 (4) (2006) 727–740.
[21] T.S. Messerges, E.A. Dabbish, R.H. Sloan, Examining smart-card security under the threat of power analysis attacks, IEEE Trans. Comput. 51 (5) (May 2002) 541–552.
[22] C. Rackoff, D.R. Simon, Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, in: Proc. CRYPTO 91, in: Lecture Notes in Comput. Sci., vol. 576, Springer-Verlag, 1992, pp. 433–444.
[23] M. Scott, Authenticated ID-based key exchange and remote log-in with simple token and PIN number, Cryptology ePrint Archive, Report 2002/164 (revised date: 10 December 2004), http://eprint.iacr.org/2002/164, 2002.
[24] M. Scott, Cryptanalysis of an id-based password authentication scheme using smart cards and fingerprints, SIGOPS Oper. Syst. Rev. 38 (2) (2004) 73–75.
[25] V. Shoup, A. Rubin, Session key distribution using smart cards, in: Proc. EUROCRYPT 96, in: Lecture Notes in Comput. Sci., vol. 1070, Springer-Verlag, 1996, pp. 321–331.
[26] B. Wang, J.H. Li, Z.P. Tong, Cryptanalysis of an enhanced timestamp-based password authentication scheme, Comput. Secur. 22 (7) (2003) 643–645.
[27] S.T. Wu, B.C. Chieu, A note on a user friendly remote authentication scheme with smart cards, IEICE Trans. Fund. E87-A (8) (2004) 2180–2181.
[28] E.J. Yoon, E.K. Ryu, K.Y. Yoo, Efficient remote user authentication scheme based on generalized ElGamal signature scheme, IEEE Trans. Consum. Electron. 50 (2) (May 2004) 568–570.
[29] E.J. Yoon, K.Y. Yoo, New authentication scheme based on a one-way hash function and Diffie–Hellman key exchange, in: 4th International Conference of Cryptology and Network Security, CANS 2005, in: Lecture Notes in Comput. Sci., vol. 3810, Springer-Verlag, 2005, pp. 147–160.