A forward secure Remote user Authentication Scheme

Manoj Kumar
Department of Mathematics
R. K. College Shamli-Muzaffarnagar U.P.-India- 247776
E-mail: yamu_balyan@yahoo.co.in

Abstract

Remote user authentication schemes allow a valid user to login a remote server. In 2000, Hwang and Li's proposed a new remote user authentication scheme with smart cards. In the recent years,some researchers pointed out the security weaknesses of Hwang and Li's scheme and they also proposed some modified schemes to avoid these weaknesses. This paper analyzes that Hwang and Li's scheme does not satisfy some essential security requirements. Hwang and Li's scheme and all the modified schemes do not support mutual authentication between the remote user and the remote server also there is no session key generation phase for secure communication. In addition, in Hwang and Li's scheme, the remote user is not free to change his password. This paper present an ideal remote user authentication scheme with smart cards that not only resolves all the security problems of Hwang and Li's scheme, but also provides all the essential security requirements and forward secrecy to the remote server.

1. Introduction

A password based remote user authentication scheme allows a authentication server(*AS*) to check the authenticity of a remote user (*U*) over an insecure channel. A typical smart card based remote user authentication scheme comprises three phases: registration phase, login phase and authentication phase. In the registration phase, a user *U* sends a registration request to *AS* and submits some necessary information to the server through a secure channel. The server uses the user's identity and password along with its long-term secret to generate some values and store some of them in a smart card, which then delivered to the user. In the login phase, a user attaches his smart card to a card reader and keys in his identity and password to login the server to gain access right. The smart card then uses the password and the values in the card to construct a login request and then sends it to the server. In the authentication phase, the server uses its long-term secret to check the validity of the login request. If mutual authentication is required, the server also uses its long-term secret to construct a message and sends it back to the user. The user then uses his password and the values in the smart card to check the validity of the message.

Lamport [17] proposed the first well-known remote password authentication scheme using smart cards. In Lamport's scheme, the *AS* stores a password table at the server to check the validity of the login request made by the user. However, high hash overhead and the necessity for password resetting decrease the suitability and practical ability of Lamport's scheme. In addition, the Lamport scheme is vulnerable to a small *n* attack [23]. In 2000, Hwang and Li [13] pointed that Lamport's scheme suffers from the risk of a modified password table and the cost of protecting and maintaining the password table. Further, they proposed a new remote user authentication scheme using smart cards. This scheme does not maintain the password table at the server to check the validity of the login request. Also, it can withstand message-replaying attack [28]. In 2000, Chan and Cheng [5] pointed out the security weakness of Hwang-Li's scheme. In 2003, Shen-Lin- Hwang [25] discussed a different type of attack on the Hwang-Li's scheme and they also proposed a modified scheme to solve the security problem of Hwang-Li's scheme. In the same year, Chang and Hwang [6] explained the practical problems of the Chan - Cheng's attack on the Hwang-Li's scheme and Leung, - Cheng, - Fong and Chen [20]pointed out that the Shen-Lin-Hwang's scheme is still vulnerable to the attack proposed by Chan and Cheng. Although so many modified schemes [22, **?**] have been proposed to solve the security problems of original scheme, but none of them provide complete solution to solve all the possible problems and withstand all possible attacks.

On the other hands,Hwand and Li's scheme also does not support the following three most essential security requirements:

1. Remote user is not free to change his password.

2. This scheme does not support session key generation.

3. The scheme does not support mutual authentication.

Thus, at this stage, We are concerned with mutual authentication and secure session generation. For security point of view, it is better to consider these topics jointly rather than separately. A protocol providing authentication without key exchange is susceptible to an enemy who waits until the authentication is complete and then takes over one end of the communications line. Such an attack is not precluded by a key exchange that is independent of authentication. Key exchange should be linked to mutual authentication so that a party has assurances that an exchanged key (which might be used to facilitate privacy or integrity and thus keep authenticity alive) is in fact shared with the authenticated party, and not an impostor. For these reasons, it is essential to keep key exchange in mind in the design and analysis of authentication protocols.
Keeping in mind all the above requirements,this paper presents an ideal remote user authentication scheme with smart cards that not only resolves all the security problems of Hwang and Li's scheme, but also provides all the essential security requirements and forward secrecy to the remote server.

1.1. Notations

The notations used through out this paper are summarized as follows:

- *U* denotes a remote user.
- *ID* denotes an identity of a remote user *U*.
- *PW* denotes a password corresponding to a registered identity *ID*.
- *AS* denotes an authentication server.
- x_s denotes a permanent secret key of an authentication server.
- *f* (.) denotes a cryptographic one way hash function.
- $\oplus$ denotes the bitwise XOR operation.
- *U*$\Longleftrightarrow$*AS*: *M* User *U* sends *M* to the server *AS* through a secret channel.
- *U* $\Longrightarrow$ *AS*:*M* denotes that user *U* sends *M* to the server *AS* through an open channel.
- *p* denotes a large prime number.
- S_{ID} denotes the redirected identity corresponding to a registered identity *ID*.
- C_{ID} denotes a check digit sum corresponding to a registered identity *ID*.
- *Red* (.) denotes a function to redirect the identity *ID* for every user *U*, which is only possessed with the *AS*.
- $C_K(.)$ denotes a function to generate check digit for the registered identity, which is only possessed with the *AS*.

1.2. Contribution

This paper presents an ideal remote user authentication scheme with smart cards. The proposed scheme not only resolves all the security problems of Hwang and Li's scheme, but also provides essential security requirements for secure communication. The proposed scheme also provides forward secrecy with respect to the long - term secret key of the *AS* , if compromised of the secret key of the *AS* does not result in compromise of the security of the previously registered identities and the corresponding passwords.The proposed scheme enables the remote user to change his password freely and securely without the help of remote server. In addition, our scheme also provides mutual authentication and session key generation for secure communication between *U* and *AS*.

1.3. Organization

The remainder of this paper is organized as follows. Section 2 reviews the Hwang - Li's scheme. Section 3 is about the security of the Hwang - Li's scheme. Section 4 presents an ideal remote user authentication scheme with smart cards. The security of the proposed scheme is analyzed in section 5. Finally, comes to a conclusion in the section 6.

2. Review of Hwang and Li's scheme

There are three phases in the Hwang-Li's scheme [13]: the registration phase, login phase and the authentication phase. In the registration phase, the user *U* sends a request to the *AS* for the registration. The *AS* will issue a smart card and a password to every user legal through a secure channel. In the login Phase, when the user *U* wants to access the *AS*, she/he inserts her/his smart card to the smart card reader and then keys the identity and the password to access services. In the authentication phase, the *AS* checks the validity of the login request.

2.1. Registration Phase

User *U* submits her/his *ID* to the *AS*. *AS* computes the password *PW* for the user *U* as $PW = ID^{x_s}$ mod *p*. *AS*

provides a password *PW* and a smart card to the user *U* through a secure channel. The smart card contains the public parameters (*f, p*), where *f* a one-way hash function

2.2. Login Phase

User *U* attaches her/his smart card to the smart card reader and keys *ID* and *PW*. The smart card will perform the following operations:

1. Generate a random number *r*.
2. Compute $C_1 = ID^r \bmod p$.
3. Compute $t = f(T \bigoplus PW) \bmod p - 1$, where *T* is the current date and time of the smart card reader.
4. Compute $M = ID^t \bmod p$.
5. Compute $C_2 = M(PW)^r \bmod p$.
6. Sends a login request C = (ID, C_1, C_2, T) to the *AS*.

2.3. Authentication Phase

Assume *AS* receives the message C at time T_c, where T_c is the current date and time at *AS*. Then the *AS* takes the following actions:

1. Check the format of *ID*. If the identity format is not correct, then *AS* will rejects this login request.
2. Check, whether $T_c - T \leq \Delta T$, where ΔT is the legal time interval due to transmission delay, if not, then rejects the login request C.
3. Compute $PW = ID^{x_s} \bmod p$ and $t = f(T \bigoplus PW) \bmod (p - 1)$.
4. Check, if $C_2 = C_1^{x_s}(ID)^t \bmod p$, then the *AS* accepts the login request. Otherwise, the login request will be rejected.

3. Cryptanalysis of Hwang and Li's scheme

3.1. Chan and Cheng's Attack

According to Chan and Cheng [5], a legal user Alice can easily generate a valid pair of identity and password without knowledge of the secrete key x_s of *AS*. Alice uses her valid pair (ID_A, PW_A) to generate another valid pair (ID_B, PW_B) as follows:
Alice computes $ID_B = (ID_A \times ID_A) \bmod p$. Then, she can compute the corresponding password $PW_B = ID_B^{x_s} \bmod p = (ID_A \times ID_A)^{x_s} \bmod p = (PW_A \times PW_A) \bmod p$. As a result, Alice can generate a valid pair (ID_B, PW_B) without knowing the secret key x_s of *AS*.

3.2. Shen-Lin-Hwang's Attack: Masquerading Attack

According to Shen, Lin and Hwang [25] masquerading attack is possible on Hwang- Li's scheme. A user Bob can masquerade another user Alice to login a remote server and gain access right. Bob computes an identity $ID_B = (ID_A)^k \bmod p$, where *k* is a random number such that gcd (*k* , *p*) = 1. Then, he submits this identity ID_B to *AS* for registration. *AS* provides a smart card and a password $PW_B = ID_B^{x_s} \bmod p$. With the knowledge of PW_B, Bob can compute $PW_A = ID_A^{x_s} \bmod p = PW_B^{-k} \bmod p$. As a result, Bob can masquerade as Alice to login a remote server and gain access privilege.

3.3. Chang- Hwang's Attack

According to Chang and Hwang [6], there is a mistake in the Chan- Cheng's attack. It is not always possible that the square of a legal identity satisfy the specific identity format. Chang and Hwang generalized the Chan- Cheng's attack. They described two attacks.
Attack-I
Alice computes $ID_B = (ID_A)^k \bmod p$, where *k* is a random number. Then, he can compute the corresponding password $PW_B = (PW_A)^k \bmod p$. As a result, a legal user Alice can impersonate other user Bob with a valid pair of (ID_B, PW_B) to login the *AS*. If ID_A is a primitive root of Z_p, then all the valid identities and their corresponding password can be generated easily.
Attack-II
A group of eavesdroppers (intruders) may cooperate to generate a valid pair of identity (ID_G, PW_G), as follows: $ID_G = \prod ID_{A_j} \bmod p$ and $PW_G = \prod PW_{A_j} \bmod p$. Chang and Hwang pointed out that in Hwang - Li's scheme, it is still difficult to obtain the corresponding password for a known arbitrary valid identity, but once the valid identity is generated, its corresponding password can be obtained easily.

4. An Ideal Remote User Authentication Scheme with Smart Cards

Hwang and Li's scheme [13] has two categories of security attacks. The first category of attacks is attack by a malicious Bob, which is not registered user at the *AS*: Shen-Lin- Hwang's attack and the second category of attacks is attack by a malicious user Alice, which is already registered at the *AS*: Chan- Cheng's attack and Chang- Hwang's Attack. Hwang and Li's scheme also does not have password change phase and there is no mutual authentication and session key generation between the remote user and remote server for secure communication. On the other end,

the secret key of the *AS* is a long-term key. It means the secret key of the server requires further security. Consider the situation, when the secret key of the *AS* is revealed or compromised by an accident or stolen etc, then it is not better to replace/alter the whole system at the *AS*. It is also not efficient to replace/alter the secret key of the *AS* with the previously registered identities and their corresponding passwords. However, the secret key of the *AS* requires further security in term of forward secrecy: the revelation or publication of the secret key of the *AS* does not result in compromise of the security of the previously registered identities and their corresponding passwords.

This section presents an ideal remote user authentication scheme with smart cards. The proposed scheme provides forward secrecy to the *AS*. Forward secrecy ensures that the previously generated identities and their corresponding passwords in the *AS* are secure even if the systems secret key x_s has been revealed or known publicly by an accident or is stolen by any adversary etc. For our requirement, we have modified the Hwang and Li's scheme. This proposed scheme uses two more functions: redirected function *Red* (.) to redirect the registered identity *ID* and a check digit function $C_K(.)$ to generates the corresponding check digit [8, 9, 10] for each registered identity. In this scheme, only the *AS* can redirect the registered identity *ID* and he is able to generate a valid identity and the corresponding check digit. This scheme has four phases: registration phase, login phase and verification phase and password change phase. These phases are described below.

4.1. Registration Phase

This phase is invoked whenever a user *U* wants to register himself at the remote server *AS*. This phase is executed over a secure channel. The following steps are involved in this phase.

Step R_1. *U* $\Longleftrightarrow$ *AS*: J

The string J is the registration request, consists the name of the user *U*, address, identity *ID* and a unique identification number etc, which are unique for the user *U*.

Step R_2.

Upon receiving the registration request, the *AS* computes the followings parameters:

$S_{ID} = Red(ID), C_{ID} = C_K(S_{ID}), PW = (S_{ID})^{x_s}$ mod *p* and $R = S_{ID} \oplus PW$.

Step R_3.

AS $\Longleftrightarrow$ *U*: $(ID \| C_{ID}, PW)$ and a smart card.

In the proposed scheme, the smart card of a user *U* contains the parameters *f, p,*$f(S_{ID})$ and *R*.

4.2. The Login Phase

Whenever, the user wants to gain the access right on the *AS*, *U* attaches her/his smart card to the smart card reader at any time *T* and keys in the *PIN* (Personal Identification Number) to active the smart card. If the PIN code is entered incorrectly multiple times, the smart card may request a *PUK* (Personal Unblocking Key) code. Inputs her/his identity $ID \| C_{ID}$ and the corresponding password *PW*. The smart card of the user *U* conducts the following computations:

Step L_1

- Compute S_{ID} = $R \bigoplus PW$, $f(S_{ID})$ and compare the calculated $f(S_{ID})$ and stored $f(S_{ID})$, if they are equal the smart card accept the password *PW* and proceeds to the next step ,otherwise demands the password again.

Step L_2

- Generate a random number *r* and compute C_1 = $R \bigoplus (S_{ID})^r$ mod *p*.

Step L_3

- Compute $t = f(T \bigoplus PW)$ mod *p* - 1, where *T* is the current date and time of the smart card reader.

Step L_4

- Compute $M = (S_{ID})^t$ mod *p* and compute $C_2 = M(PW)^r$ mod *p*.

Step L_5

- *U* $\Longrightarrow$ *AS*: L_R = $(ID \| C_{ID}, C_1, C_2, R, T)$.

4.3. The Verification phase

Assume that the *AS* receives the login request L_R at time T_c. Then, *AS* does the following computations to check the validity of the login request L_R.

Step V_1

- Check the specific format of the identity *ID*. If the format of the identity is incorrect, then *AS* rejects the login request L_R.

Step V_2

- Computes the value S_{ID} = *Red*(*ID*). Check, whether the condition C_{ID} = C_K (S_{ID}) holds, if not, then *AS* rejects the login request L_R.

Step V_3

- Check, whether $T_c - T \leq \Delta T_S$, where ΔT_S is the legal time interval due to transmission delay, if not, then *AS* rejects the login request L_R.

Step V_4.

- Compute $PW = R \bigoplus S_{ID}$ and $t = f(T \bigoplus PW)$ mod *p* - 1, and check, if $C_2 = (C_1^{x_s})(S_{ID})^t$ mod *p*, then the *AS* accepts the login request and proceeds to the next step, Otherwise the login request will be rejected by *AS*.

Step V_5.

- The *AS* selects a random number r_1 and computes the following values:
  $C_3 = f(C_1^{x_s} \bigoplus T_s)$, where T_s is the current time at *AS*.
  $S_{key} = f(C_1^{x_s}, T_s, r_1)$, $C_4 = C_3 \bigoplus r_1$,$C_5 = C_3 \bigoplus S_{key}$.

Step V_6

- *AS*$\Longrightarrow$*U*: (C_4, C_5, T_s).

Step V_7

- Assume that the *U* receives the message $C_4, C_5, T_s)$ at time T_U, then *U* verifies, whether $T_U - T_S \leq \Delta T_U$, where ΔT_U is the legal time interval due to transmission delay,if not, then *U* interrupts the connection.
- *U* computes $C_3^* = f(C_2 M^{-1} \bigoplus T_s)$.
- Computes $r^* = C_3^* \bigoplus C_4$.
- Computes $S_{key}^* = C_3^* \bigoplus C_5$.
- Computes $S_{key}^{**} = f(C_2 M^{-1}, T_s, r^*)$.
- Compares S_{key}^* and S_{key}^{**} for mutual authentication, if they are equal the user *U* ensures that the responding system is a real *AS* and proceeds to the next step.otherwise *U* interrupts the connection. The number S_{key}^* will be the session key between the user *U* and *AS*,

Step V_8

- *U* Computes $C_6 = f(C_3^*, S_{key}^*)$,
- *U* $\Longrightarrow$ *AS*: (ID, C_6)

Step V_9

- *AS* checks, if $C_6 = f(C_3, S_{key})$, then the *AS* assures that the user *U* also generates the same session key, otherwise rejects the connection.

4.4. The Password change phase

This phase is invoked whenever *U* wants to change his password *PW* with a new password, say *PWnew*. This phase has the following steps.
Step P_1

- *U* inserts her/his smart card to the smart card reader and then keys in the PIN to active the smart card,then inputs her/his identity and the old password *PW* and then requests to change the password.

Step P_2

- Compute $S_{ID} = R \bigoplus PW$, $f(S_{ID})$ and compare the calculated $f(S_{ID})$ and stored $f(S_{ID})$, if they are equal the smart card accept the password change request and proceeds to the next step ,otherwise demands the correct the password again.

Step P_3

- *U*'s smart cards computes *R** = $R \bigoplus PW \bigoplus PWnew$ and then replaces *R* with *R**.

5. Security Discussion

Secure mutual authentication and secret session key generation are two important pillars, which are responsible for the security of a remote user authentication scheme. In other words; a remote user authentication scheme is secure if each user can get an authenticated secret session key after performing the secure mutual authentication protocol and all other users can learn nothing about that session key. To discuss the security of these two protocols: mutual authentication and secret session key generation of the proposed scheme, this section is divided in two subsections. The subsection 5.1 provides some preliminaries and assumptions related to security of authenticated protocols. The subsection 5.2 demonstrates the security of the proposed scheme by random oracle model [2]. Besides, subsection 5.3 prove that the proposed protocol provides explicit key authentication . We demonstrates the proposed protocol resists the replay attack [28], stolen verifier attack [7], Shen-lin Hwang attack, Chan and Cheng's Attack/Chang-Hwang attack in subsections 5.4, 5.5, 5.6 and 5.7 respectively . The subsection 5.8 proves that the proposed scheme is forward secure.

5.1. Prelimnaries

This section presents the definitions, assumption and theorems, which are used to prove the security of the proposed scheme.
Exclusive OR operation- The notation $Z = X \bigoplus Y$ is as *Z* is equal to *X* bitwise Exclusive OR *Y* . Ghanem and Wahab

[11] proved that the Exclusive OR operation is secure, efficient and fast for computation. The Exclusive OR operation has following properties:

- *Z, X* and *Y* are represented in the same bit length.
- All output values are uniformly distributed in the output space.
- If we know any two values out of *X* , *Y* and *Z* ,then the third unknown can be determine easily.
- For any *Z* with *n* bits, there are 2^n different pairs (*X, Y*) satisfy $Z = X \bigoplus Y$.

.

Theorem 5.1.1 *Let x and y are two n bits specific values and* $Z = X \bigoplus Y$. *The probability to get the specific values X and Y from the given Z is negligible when n is large enough.*

Proof. There will be 2^n different possible pairs (x^*, y^*) which satisfy $Z = x^* \bigoplus y^*$.Thus,the probability to get the specific *(X, Y)* from the given *Z* is $\frac{1}{2^n}$, which is negligible, when *n* is large.

5.1.1 The Computational Diffie-Hellman Problem

In *computational Diffie-Hellman (CDH) problem* for given g^{u_1}, g^{u_2} and for random $u_1, u_2 \in Z_q$, compute $g^{u_1 u_2}$.

5.1.2 The Decisional Diffie-Hellman Problem

In *decisional Diffie-Hellman (DDH) problem* for given g^{u_1}, g^{u_2} and for random $u_1, u_2 \in Z_q$, distinguish between $g^{u_1 u_2}$ and a random group element.

5.1.3 Check Digit Scheme

A check digit scheme consists of two algorithms (`CDig`,`Vrfy`). For a given random computation function CPerkey *F*, the algorithm `CDig` computes a standard public value $CDSum = \Omega$ for a given number/ message *M*; this can be written as $\Omega \longleftarrow \texttt{CDig}_F(M)$.The algorithm `Vrfy` checks the validity of the pair(Ω, M) and return 1 if Ω is valid or 0 otherwise. We require that for all function CPerkey *F*, for all *M* and for all $ChDiSum\Omega$ output by $\texttt{CDig}_F(M)$,$\texttt{Vrfy}_F(\Omega, M)$. To defining the security of a `CDig`, we use the standard definition of strong unforgeability under adaptive chosen-message attack. Namely,let *C* is a check digit scheme and *A* be an adversary then consider the following experiment:

- $\underline{\textbf{Exp}^{SUF}_{A,C}(k)}$
- $CPerkeyF \longleftarrow (0,1)^k$
- $(\Omega, M) \longleftarrow A^{\texttt{CDig}_F(.)(1^k)}$
- If $\texttt{Vrfy}_F(M, \Omega) = 1$ and oracle $\texttt{CDig}_F(.)$ never returned Ω on input *M* then return 1 else return 0

The advantage of an adversary *A* is defined as:

$$\textbf{Adv}^{SUF}_{A,C}(k) = \Pr[\ \textbf{Exp}^{SUF}_{A,C}(k) = 1\ ]$$

We say that *C* is strongly unforgeable (SUF-secure) if $\textbf{Adv}^{SUF}_{A,C}(k)$ is negligible for all PPT algorithms *A*. When we are interested in a concrete security analysis, we drop the dependence on *k* and say that C is $(t, q, \epsilon) - SUF - secure$ if$\textbf{Adv}^{SUF}_{A,C} \leq \epsilon$ for all *A* running in time *t* and making at most *q* queries to its `CDig`oracle. (We remark that allowing *N* queries to an oracle $\texttt{Vrfy}_F(.,.)$ cannot increase the advantage of an adversary by more than a factor of *N*.)

5.1.4 Random oracle model

To design a cryptographic protocol, hash function always plays an important role and therefore also has a vital role over the security of such protocols. The implementation of hash function also responsible for the efficiency a cryptographic protocol. But,it is not easy to obtain security arguments (or proofs of security) for such protocols. If a hash function *f* is well designed, then it should be infeasible to compute *f(x)* without evaluating *f* on *x*. This should be the case even if many other hash values $f(x_1); f(x_2)$,... have been computed. Bellare and Rogaway [2] therefore advocated an idealized model for hash functions, which attempts to capture the concept of an ideal hash function. This model is commonly referred to as the random oracle model, and involves modelling hash functions as random functions.Many researcher made extensions to this model. [3, 4, 16]. To discuss the security of the proposed scheme(denoted as protocol Π) in random oracle model, this subsection uses the reduced modified BJM Game [16], which is a variant of random oracle model.We model hash function *f* as a random oracle in our security analysis. In this paper, a technique [16] is used to proving the security of a protocol Π, which works as follows. The first step is to prove that protocol Π has a property that we call strong partnering (which is defined in the next discussion). The second step is to prove that a related protocol π is secure in a highly reduced security model. Finally, we show how the proof of security of π in the reduced model can be translated into a proof of security for Π in the full security model using a Gap assumption. We use the standard notations [16] of security which are defined as follows.

PROTOCOL PARTICIPANTS:
we fix a nonempty set of *ID* of participants. The set *ID* is the union of two nonempty, finite and disjoint subsets User and Server.Each participants is named by a string of fixed length. When the user *U* wants to login the server *AS*, the

user and the server authenticate each other and establish a common session key. The notation Π_A^m denotes the oracle which models the m^{th} instance of participant A.

ORACLE QUERIES

`Send(U, m,M)`: This sends a message M to the oracle Π_A^m, the oracle computes what the protocol says to and sends back the response. The adversary *A* can send a fabricated query Send(U, m, λ) to a user oracle Π_A^m to initiate an execution of protocol Π, of any user *A*.

`Reveal(U,m)`: If Π_A^m has accepted and is holding some session key *sk*, then query returns *sk* to the adversary. This query models the idea that loss of session key should not be damaging to other session.

`Corrupt(U,PW)`: The adversary obtains the password of the user *U*/ long-term private key of the server. This query models the forward secrecy, which means if an adversary gets the long-lived key of a participant, he cannot snatch any previous session key.

`Test(A ,SID)` The only oracle query that does not correspond to any of A's abilities. Depending on a randomly chosen bit *b*, A is given either the actual session key or a session key drawn randomly from the session key distribution. The adversary is limited to only one such query, which can be made at any time during the experiment.

ORACLE STATES:

`Accepted`: If an oracle decides to accept and holds a session key after it received some messages, the oracles state is accepted.

`Rejected`: If an oracle decides not to establish a session key and thus abort the protocol, the oracles state is rejected.

`State` $\star$: If an oracle has not made any decision to accept or reject, the oracle is in state $\star$.

`Revealed`: If an oracle has answered a reveal query, the oracles state is revealed.

`Corrupted`: If an oracle has answered a corrupt query, the oracles state is corrupted.

PARTNERS:

When running a protocol Π, the oracle may hold a partner identity *PID*, a session identity *SID*, and a session key *sk*. The partner identity shows who has exchanged messages and established a session key in the protocol Π. The session identity is the session identifier in the protocol Π. When executing protocol Π, we say that oracles Π_A^m and Π_B^n are partner if both oracles accepted, holding (sk_A, SID_A, PID_A) and (sk_B, SID_B, PID_B), respectively, and the following conditions hold:

1. $sk_A = sk_B, SID_A = SID_B, PID_A = B$ and$PID_B = A$.

2. No oracle besides Π_A^m and Π_B^n accepts with a session identity equal to $SID_A (or SID_B)$.

FRESHNESS:

If an oracle Π_A^m is revealed, or its partner Π_B^n is revealed, orΠ_B^n is corrupted, then the oracle Π_A^m is called unfresh . If an oracle is not unfresh, the oracle is fresh.

STRONG PARTNERING: Suppose Π is a key agreement protocol. If there exists an adversary *A*, which when attacking Π in an mBJM game [16] and with non-negligible probability in the security parameter *l*, can make some two oracles Π_A^m and Π_B^n accept holding the same session key when they are not partners, then we say that the protocol Π has weak partnering. If Π does not have weak partnering, then we say that Π has strong partnering.

RELATED PROTOCOLπ:

We define a related protocol π in order to help us to prove the security of the original proposed protocol Π. The related protocol π is similar to the protocol Π, with exception of the method of computing the session key between the user and server. If the session key in protocol Π is the hash value of the session string, then the session key in protocol π is the session string itself. Therefore the user and server do not use the hash function to compute the session key in protocol π.

THE REDUCED MODIFIED BJM GAME:

The reduced modified BJM game is identical to the BJM game except that the adversary *A* is not allowed to send any reveal and test queries. Instead to win the game, the adversary *A* must select a fresh and accepted oracle as the test oracle and output the session key of this test oracle at the end of this game. Because the adversary *A* in this game must compute the session key of the test oracle, this game also called the computational No-Reveals mBJM (cNR-mBJM) game. In the cNR-mBJM game, we use $Advantage^A(p)$ to denote the adversary *A*s advantage, that is, the probability that *A* outputs a session key *sk* such that *sk* = $sk_{\Pi_A^m}$, where $sk_{\Pi_A^m}$ is the session key held by the test oracle Π_A^m selected by *A*.

Definition 5.1.4 A protocol Π is a cNR-mBJM-secure key agreement protocol if:

1. In the presence of the benign adversary,(a benign adversary is one who simply relays messages between parties without modification) two oracles running the protocol both of the oracles are accepted and holding the same session identity and session key. The session key is uniformly distributed in *GF*(*p*).

2. For any adversary *A*, the advantage $Advantage^A(p)$ in the cNR-mBJM game is negligible.

5.2. Security proof of the proposed scheme

This subsection demonstrates the security of the proposed protocol in the random oracle model.

THEOREM 5.2.1 The proposed scheme has a strong partnering in the random oracle model.
Proof. Observe the steps V_5 to V_9 of the proposed scheme, the partnering information of the user and server is included in the session string. Thus the proposed scheme has strong partnering in the random oracle model.
THEOREM 5.2.2 If protocol Π produces a hashed session key via hash function *f* and is NR-mBJM secure, then the related protocol π is cNR-mBJM secure.
Proof: This theorem shows that if there exists an adversary *A* that can cNR-mBJM-attack π, then we can build an adversary B that can NR-mBJM-attack Π. Suppose that an adversary *A* wins the cNR-mBJM game for protocol π with nonnegligible probability η. Also suppose that B runs an NR-mBJM game with challenger *C*. B in turn acts as a challenger for *A* in a cNR-mBJM game. B passes all queries of *A* to *C* and returns all outputs of *C* to B. Finally B will output the session key $sk\pi_U^i$ of some fresh oracle π_U^i. Recall however that $sk\pi_U^i = ss\Pi_U^i$. B then chooses Π_U^i for the `Testquery` and receives a key *sk*. If *sk* =*f*($sk\pi_U^i$) then B outputs 1, otherwise B outputs 0. It is easy to see that B wins the NR-mBJM game against Π with probability η. We note that in the proof of the above theorem, no assumption is required concerning the properties of *f*.
THEOREM 5.2.3- The cNR-mBJM security of Protocol π is probabilistic polynomial time reducible to the hardness of the *CDH* problem in group*G*.
Proof: Assume that for security parameter *l* there exists an adversary *E* for Protocol π that can win the cNR-mBJM game with non-negligible advantage η and in polynomial time τ . Suppose that the number of participants in the game of *E* are n_P and that the maximum number of sessions each participant may be involved in is n_S, where n_P and n_S are polynomial functions of *l*.

We now construct from *E* an algorithm *F* which solves the *CDH* problem in *G* with non-negligible probability. *F* simulates a challenger in a cNR-mBJM game with *E*. *F* sets up the game with the group *G* and generator $g \epsilon G$. *F* generates a set of participants of size n_P.For each user *Ui*, *F* sets private values: *PWi* and *Ri* of each *Ui*. For the server, *F* sets X_s as server's private key. *F* selects a session number r_f. *F* starts *E* and answers the following queries sent from *E*:
`Send`: *E* may send a special `Send` query to user oracle Π_{Ui}^S which sets $pid_{U_i} = AS$ and instructs *Ui* to initiate a protocol run with *AS* as its partner. *E* can also send a *Send* query with message *M* to any oracle Π_A^m and the oracle outputs the response according to the protocol Π. If E sends an initiate Send query to user oracle Π_A^m, it outputs α.
`Corrupt(U)`: If adversary *E* sends a corrupt query to the user A, then *F* aborts. If *E* sends a Corrupt query to other participants, then *F* gives its private value to *E*. If *F* wants to use *E* to find out the output value of *CDH* problem, then *E* must set oracle Π_A^m as the Test oracle. The probability that *E* sets oracle Π_A^m as the Test oracle is $\frac{1}{n_P \times n_s}$. *E* outputs C_4 and $C_5 \in Zp$. Then *F* determines whether *E* sent an initiate Send query to oracle A . If Π_A^m was an initiator, then *F* outputs C_4 and $C_5 \in Zp$ as its guess for the value C_3; otherwise *F* outputs C_4 and $C_5 \in Zp$ as its guess. Now we can see that if the probability that *E* wins the cNR-mBR game of protocol Π in time, then the probability that *F* solves the *CDH* in GF(p) is negligible.
THEOREM 5.2.4 Protocol Π is secure in the random oracle model, assuming the hardness of the Gap finding Decisional Diffie-Hellman(*DDH*) Problem.
Proof.- In protocol Π, the user and server use a hash function to computes a hashed session key. Protocol Π has strong partnering in the random oracle model by Theorem 5.2.1. By Theorem 5.2.2, the cNR-mBJM security of the related protocol π is probabilistic polynomial time reducible to the hardness of the *CDH* problem. In protocol Π, we assumed that user *U* wants to login server *AS*. They establish a session key in this connection. A decisional finding *DDH* problem oracle can be used to solve the session string decisional problem for protocol Π. Therefore, the session string decisional problem for protocol Π is polynomial time reducible to the decisional finding *DDH* problem. According to previous results and by theorem 5.2.3, we can say that the mBJM security of protocol Π is probabilistic polynomial time reducible to the hardness of the Gap finding *DDH* problem. If the Gap finding *DDH* problem is hard, then the protocol Π is mBJM secure in the random oracle model.
THEOREM 5.2.4 The proposed protocol Π is secure in the random oracle model when *p* is a large prime.
Proof. We know that Gap finding discrete logarithm problem is hard when *p* is a large prime. By Theorem 5.2.3, Protocol Π is secure in the random oracle model assuming the hardness of the Gap finding Decisional Diffie-Hellman(*DDH*) Problem. According to Theorem 5.1.1 and Theorem 5.2.4, the proposed protocol Π is secure in the random oracle model.

5.3 Explicit key authentication

Let *U* and *AS* be two honest terminals who execute the steps of an authentication protocol correctly,then an authentication scheme provides the explicit key authentication, if it should satisfy following two properties [15]:

- **Implicit key authentication**- Informally speaking, an authentication protocol is said to provide implicit key authentication (of *AS* to *U*) if entity *U* is assumed that no other entity from a specifically identified second entity *AS* can possibly learn the value of the particular secret key.

- **Key confirmation** - an authentication protocol is said to provide key confirmation(of *AS* to *U*) if entity *U* is

assumed that second entity *AS* actually possession of a particular secret key

Observe the steps V_5 to V_7 verification section of the proposed scheme. These steps shows that only the specified user and specified server can get correct information which can be used to generate a valid session key. This means that the proposed scheme provides implicit key authentication. In step V_7 the server *AS* assures the user had computed the same session By this result, it is clear that the proposed protocol provides explicit key authentication.

5.4 Replay attack

When the adversary impersonates a legal user to login the specified server by replaying the transmitted messages between the legal user and that server, then we say that this protocol is vulnerable to the replay attack [28]. Suppose that an adversary collects the messages L_R = $(ID\|C_{ID}, C_1, C_2, T)$ from Step L_5, (C_4, C_5, T_s) from Step V_6 and (ID, C_6) from Step V_8 of the proposed protocol when the user *U* logs into the server *AS* . The adversary impersonates the user *U* to login the server *AS* by replying the message L_R = $(ID\|C_{ID}, C_1, C_2, T)$. The Step V_3 of the verification phase does not satisfy, due to the invalid time interval. It is clear that the adversary can not select a valid time *T* to avoid this invalid transmission delay. Thus, the server will detect that he/she is not a valid user *U*.
Also, the adversary can not generate the correct(C_4, C_5, T_s) corresponding to r_1 and returns it to the user *U* because he does not know the secret key of the server *AS*. In this case, the user *U* will detect the fabricated server with the help of Step V_7. In the same way, the Step V_9 will detect the replaying of the message (ID, C_6). Hence, it is very hard for an adversary to masquerade the legal user to login the server by replaying the old message.

5.5 Stolen verifier attack

The proposed scheme is free from the stolen verifier attack [7]. There is such information is stored at the server,by which an adversary can make a fabricated login request to impersonate a legal user to login the server, or can impersonate the server to cheat the legal user.

5.6 Shen- Lin- Hwang's attack

In Shen- Lin- Hwang's attack [25], the attacker Bob is not a registered user at the *AS*. To create some favorable results for a successful attack, he requires the redirected identity S_{ID} of a previously registered user, say Alice. But in our scheme, the redirected identity S_{ID} of every registered user is calculated secretly by the *AS* with the help of *Red* (.) function. The function *Red* (.) redirects a valid identity into a shadow identity S_{ID} on the basis of the information, which is sent by the user at the time of registration request. *AS* computes the password by using the $PW = (S_{ID})^{x_s} mod\ p$, where S_{ID} a redirected secret value corresponding to the registered identity *ID* of the string *J*. Assume that an eavesdropper, Bob intercepts the login request L_R = $(ID\|C_{ID}, C_1, C_2, R, T)$ from a public network, then it is clear that by using the login request L_R neither he can obtain any information to attack the scheme nor he can compute the password *PW* from this login request L_R. In our scheme, there is no way for the attacker to register herself/himself by intercepting the login request L_R. He is not able to produce any fabricated results for a successful attack. Consequently, the functionality of *Red* (.) blocks the masquerade attack via identity: Shen- Lin-Hwang's attack.

5.7 Chan- Cheng's /Chang- Hwang's Attack

In Chan- Cheng's attack and Chang- Hwang's Attack, the attacker Alice is a registered user at the *AS*. To create some fabricated results for a successful attack, only he has the knowledge of a secret redirected identity S_{ID} corresponding to her registered identity *ID*. To perform Chan-Cheng's attack and Chang- Hwang's attack, the attacker Alice computes $S_{ID_B} = (S_{ID_A})^k mod\ p$,where k is a random number. Then, he can compute the corresponding password $PW_B = (PW_A)^k mod\ p$. This result is incomplete; still, it is essential to obtain the corresponding check digit of S_{ID_B} . In our scheme, only the *AS* can generate a valid check digit corresponding to the redirected identity S_{ID_B}. As a result, a legal user Alice cannot compute a valid pair of identity and password to impersonate other user Bob to gain the access login right at the *AS*. Thus, Chang- Hwang's Attack will not work. Since, Chan - Cheng's attack is another form of this attack, so this attack also will not work. Consequently, the functionality of C_K (.) blocks the attacks via password - Cheng's attack/Chang- Hwang's Attack.

5.8 Forward Secrecy

Take a look on the registration phase of our scheme. With a secret key x_s, the *AS* uses two additional functions: *Red* (.) and C_K (.), which are always in possession of *AS*. In this way, only the *AS* is able to compute a redirected/ shadowed identity S_{ID} and a check digit sum C_{ID} corresponding to every valid identity *ID*. Unfortunately, if the secret key x_s of the *AS* is revealed or compromised by an accident or stolen etc, then with the help of revealed secret key x_s any attacker Bob can try to obtain the password *PW* corresponding to the previously registered identity string

J/ID or he can try to generate new password by selecting a newly valid identity string *Jnew*. Thus, he can try to obtain some fake passwords. But, when he tries to obtain the password *PW* corresponding to a previously registered *ID* or the password corresponding to a newly selected valid identity string *Jnew*, he is required to compute a redirected/ shadowed identity S_{ID} and a check digit sum C_{ID} corresponding to every valid identity string *J*, whether it is old or new. Without the knowledge of corresponding shadowed identity S_{ID} and a check digit sum C_{ID} for a identity *ID*, the attacker will not be able to recover a valid pair of proper identity and the proper corresponding password to make a valid login request. The login request does not leak any information for the attacker, while the attacker is in possession of the secret key of the *AS*. Thus, our scheme provides forward secrecy with respect to the long - term secret key x_s of the *AS* if compromised of the secret key of the *AS* does not result in compromise of the security of the previously registered identities and the corresponding passwords.

6 Conclusions

This paper proposes an ideal remote user authentication scheme with smart cards. The proposed scheme not only provides mutual authentication between the user and server, but also establishes a common session key to provide message confidentiality. In addition, the proposed protocol provides the explicit key authentication property for established common session keys. The proposed protocol is provably secure to withstand the replay attack, the stolen verifier attack. We demonstrate the security of the proposed protocol using the random oracle model. In the password change phase of the proposed protocol, each user can change his password without connecting to any server.
Always it is prudent to keep the secret key of any *AS* so that only the authorized person/system can retrieve the secret key, whenever it is required. A possible way is to encrypt the key in a way that it can only be constructed with the help of some sorts of independent servers/machines. To avoid the risk of stealing the secret key of the *AS*, protection of the secret key can be traded off against revealing or stealing. Unfortunately, if the secret key x_s of the *AS* is revealed or compromised by an accident or stolen etc, then with the help of revealed secret key x_s any attacker Bob/Alice can not recover the complete passwords corresponding to the previously registered identities strings *J*. The attacker is also not able to construct or generate new passwords by selecting a newly valid identity string *Jnew*. Due the combined functionality of *Red* (.) and C_K(.) at the *AS*, the malicious user will not be able to make Shen-lin Hwang attack, Chan and Cheng's Attack/Chang-Hwang and all extended attacks on the proposed scheme through identity/password. The proposed scheme also provides forward secrecy with respect to the long - term secret key x_s of the *AS* if compromised of the secret key of the *AS*does not result in compromise of the security of the previously registered identities and the corresponding passwords. Consequently, the proposed scheme provides the forward secrecy to the long term secret x_s of the *AS* and as well as it also overcomes the security flaws of Hwang - Li's scheme.

References

[1] Bellare, M. and Rogaway, P. 1993. Random oracles are practical: A paradigm for designing efficient protocols. *Proceedings of the First ACM Conference on Computer and Communications Security (CCS'93)*, Fairfax, Virginia, USA, 3-5 November, pp. 62-73. ACM Press, New York.

[2] Bellare, M. and Rogaway, P. 1995. Provably secure session key distribution: the three party case. *Proceedings of 27th ACM Symposium on Theory of Computing (STOC 95)*, Las Vegas, Nevada, USA, 29 May-1 June, pp. 57-66. ACM Press, New York.

[3] Bellare, M., Pointcheval, D. and Rogaway, P. 2000. Authenticated key exchange secure against dictionary attacks. *Advances in Cryptology - EUROCRYPT 2000: International Conference on the Theory and Application of Cryptographic Techniques*, LNCS 1807, Bruges, Belgium, 14-18 May, pp. 122-138. Springer- Verlag, Berlin.

[4] Black-Wilson, S., Johnson, D. and Menezes, A. 1997. Key agreement protocols and their security analysis. *Proceedings of 6th IMA International Conference on Cryptography and Coding*, LNCS 1355, Cirencester, UK, 17-19 December, pp. 30-45. Springer-Verlag, Berlin.

[5] Chan C. K. and Cheng L. M. 2000. Cryptanalysis of a remote user authentication scheme using smart cards: *IEEE Trans. Consumer Electronic*, 46, 992-993.

[6] Chang C. C. and Hwang K. F. 2003. Some forgery attack on a remote user authentication scheme using smart cards: *Informatics*, 14, 3: 189 - 294.

[7] Chen, C.M. and Ku, W.C. 2002. Stolen-verifier attack on two new strong-password authentication protocols. *IEICE Transactions on Communications*, E85-B (11), 2519-2521.

[8] Gallian J. A. 1991. Assigning driver's license number: *Mathematics Magazine* - 64, 13-22.

[9] Gallian J. A. 1992. Breaking the Missouri license code: The UMAP Journal -13: 37-42.

[10] Gallian J. A. and Winters S. 1988. Modular arithmetic in the marketplace: *The American Mathematical Monthly*- 95: 548-551.

[11] Ghanem, S.M. and Wahab, H.A. 2000. A simple XOR based technique for distributing group key in secure multicasting. *Proceedings of Fifth IEEE Symposium on Computers and Communications (ISCC 2000)*, Antibes, France, 3-6 July, pp. 166-171. IEEE Computer Society, Los Alamitos, CA.

[12] Gollmann, D. 2003. Authentication by correspondence. *IEEE Journal on Selected Areas in Communications*, 21(1), 88-95.

[13] Hwang M. S. and Li L. H. 2000. A new remote user authentication scheme using smart cards: *IEEE Trans. Consumer Electronic*, 46, 1: 28-30.

[14] Hwang, M.-S., Lee, C.-C. and Tang, Y.-L. 2002. A simple remote user authentication scheme. *Mathematical and Computer Modeling*, 36(1-2), 103-107.

[15] IEEE P1363.2-D13, 2004. Standard Specifications for Password-based Public Key Cryptographic Techniques. *IEEE P1363 working group*, 12 March.

[16] Kudla, C. and Paterson, K.G. 2005. Modular security proofs for key agreement protocols. *Advances in Cryptology - ASIACRYPT 2005: 11th International Conference on the Theory and Application of Cryptology and Information Security*, LNCS 3788, Chennai, India, 4-8 December, pp. 549-565. Springer-Verlag, Berlin.

[17] Lamport L. 1981. Password authentication with insecure communication: *Communication of the ACM*, 24, 11: 770-772.

[18] Lee S. W., Kim H. S. and Yoo K. Y. 2004. Comment on a remote user authentication scheme using smart cards with forward secrecy: *IEEE Trans. Consumer Electronic*, 50, 2: 576-577.

[19] Lennon R. E., Matyas S. M. and C. H. Mayer. 1981. Cryptographic authentication of time-variant quantities: IEEE Trans. on Commun. COM -29, 6: 773 - 777.

[20] Leung K. C., Cheng L. M., Fong A. S. and Chen C. K. 2003. Cryptanalysis of a remote user authentication scheme using smart cards: *IEEE Trans. Consumer Electronic*, 49, 3: 1243-1245.

[21] Leveque W. 1990. *Elementary Theory of Number*. ISBN: 0-486-42539-8. Dover Publication.

[22] Manoj K. 2004. Some remarks on a remote user authentication scheme using smart cards with forward secrecy. *IEEE Trans. Consumer Electronic*, vol. 50, no. 2, pp. 615-618.

[23] Mitchell C. J. and Chen l. 1996. Comments on the S/KEY user authentication scheme: *ACM Operating System Review*, 30, 4:12-16.

[24] NIST FIPS PUB 180. 1994. Secure Hash Standard. *National Institute of Standards and Technology*, U.S. Department of Commerce.

[25] Shen J. J., Lin C. W. and Hwang M. S. 2003. A modified remote user authentication scheme using smart cards: *IEEE Trans. Consumer Electronic*, vol. 49, 2: 414-416.

[26] Smart, N. 2002. *Cryptography*. McGraw-Hill Education, UK.

[27] Udi M. 1996. A simple scheme to make passwords based on the one-way function much harder to crack: *Computer and Security*, 15, 2:171 - 176.

[28] Yen S. M. and Liao K. H. 1997. Shared authentication token secure against replay and weak key attack: *Information Processing Letters*, 78-80.