available at www.sciencedirect.com
ScienceDirect
journal homepage: www.elsevier.com/locate/cose
Computers & Security

Digest

Efficient multi-server authentication scheme based on one-way hash function without verification table

Jia-Lun Tsai

Department of E-Learning, National Chiao Tung University, No. 1001 Ta Hsueh Road, Hsinchu 300, Taiwan, ROC

ARTICLE INFO

Article history:
Received 19 April 2007
Received in revised form
25 November 2007
Accepted 25 April 2008

Keywords:
Authentication scheme
Multi-server
Smart card
One-way hash function
Verification table

ABSTRACT

Following advances in network technologies, an increasing number of systems have been provided to help network users via the Internet. In order to authenticate the remote users, password-based security mechanisms have been widely used. They are easily implemented, but these mechanisms must store a verification table in the server. If an attacker steals the verification table from the server, the attacker may masquerade as a legal user. To solve the verification table stolen problem, numerous single server authentication schemes without verification tables have been proposed. These single authentication schemes suffer from a shortcoming. If a remote user wishes to use numerous network services, they must register their identity and password in these servers. In response to this problem, numerous related studies recently have been proposed. These authentication schemes enable remote users to obtain service from multiple servers without separately registering with each server. This study proposes an alternative multi-server authentication scheme using smart cards. The proposed scheme is based on the nonce, uses one-way hash function, and does not need to store any verification table in the server and registration center. The proposed scheme can withstand seven well known network security attacks.

© 2008 Elsevier Ltd. All rights reserved.

1. Introduction

With the rapid growth of computer networks, increasing numbers of systems provide services via the Internet. Primarily via the convenience of the Internet, distant users can share information with each other. The Internet has facilitated information sharing, but typically access to valuable information is restricted to authorized users. Therefore, it is important to authenticate the identity of remote users. In 1981, Lamport proposed an authentication scheme for use in insecure networks. Numerous authentication schemes have been proposed. However, these proposed schemes require the server to contain a verification table; making it easy for hackers to steal information. In 1990, Hwang et al. proposed a single server authentication scheme without verification table. Increasing numbers of single server authentication schemes without verification table have been proposed, but these schemes use difficulty level of scattered logarithm, or asymmetrical encryption methods. Consequently the communication and computational costs of these schemes are very high, making them unsuitable for use with limited computing equipment. In 2000, Sun proposed a single server authentication scheme using a one-way hash function without a verification table. This single server authentication scheme only uses timestamp and one-way hash function to authenticate remote users. This approach reduces the communication

E-mail address: crousekimo@yahoo.com.tw
0167-4048/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2008.04.001

and computation costs, and thus numerous single server authentication schemes using one-way hash function without verification table recently have been proposed.

These single authentication schemes suffer a significant shortcoming. If a remote user wishes to use numerous network services, they must register their identity and password at these servers. It is extremely tedious for users to register numerous servers. Therefore, Li et al. (2001) proposed a multi-server authentication scheme using neural networks without any verification table in 2001. Any remote user can obtain service from multiple servers without needing to register individually with every server in these authentication schemes, and individuals can only register with the registration center once. However, the communication and computation costs in this authentication scheme based on neural networks are extremely high. More recently, numerous studies have applied more complex methods to help users access multiple servers. In 2003, Lin et al. proposed a multi-server authentication scheme without verification table based on the discrete logarithm problem. Moreover, in 2004, Tsaur et al. proposed a multi-server authentication scheme based on the RSA cryptosystem and Lagrange interpolating polynomial. These schemes are not efficient, because they involve high communication and computation costs. In 2004, Juang proposed a multi-server authentication scheme using symmetrical encryption methods without verification table. This approach not only solved the problem of repeat registration, but also that of computation efficiency. In 2004, Chang and Lee proposed a multi-server authentication scheme using symmetrical encryption methods without verification table, which was more efficient than the multi-server authentication scheme of Juang. The registration center and all system servers are assumed to be trustworthy, and the registration center sends each server the secret key Kx via a secure channel following they are authorized. This authentication scheme involves numerous security problems.

In this study, we propose a multi-server authentication scheme using smart card. This multi-server authentication scheme is also based on the random nonce, and thus it does not suffer from the time synchronization problem. The proposed multi-server authentication scheme is extremely suitable for use in distributed network environments. Following mutual verification, a session key is produced for encrypting all the messages of the subsequent communication in the same session.

2. Proposed multi-server authentication scheme

This section introduces the proposed multi-server authentication scheme based on one-way hash function. This proposed multi-server authentication scheme comprises four phases as follows. Figs. 1 and 2 show diagram of our proposed authentication scheme.

1. User registration phase
   When a user wants to become a legal user, the user must submit his/her identity ID_u and password PW_u to the server via secure channel. Then, the registration center computes some secure parameters stored in the smart card, and sends this smart card to this user via secure channel.
2. Login phase
   When a legal user wants to login the server S_j, this user must insert his/her legal smart card and enter his/her identity ID_u and password PW_u. Then, the user sends authentication messages computed by the smart card to the server S_j.
3. Authentication server and registration center phase
   When the server S_j receives the authentication messages from the user, this server S_j asks for the registration center provide the user authentication key to help the server S_j verify the user after the server and the registration center verify each other. To prevent the server S_j from knowing the $h(ID_u\|x)$ of a user, the value of this user authentication key $h(h(ID_u\|x)\|N_c)$ is changed with random nonce N_C. To avoid the cost of confirming the server S_j and registration center, the registration center and the server S_j could store the secret key $h(h(SID_j \| y) \| N_S+1 \| N'_{RC}+2)$ for the user temporarily in the system after successfully authenticating each other.
4. Authentication server and user phase
   When the server S_j has received authentication key from the registration center, this server S_j uses this authentication key to verify the user. After authentication is complete, a session key is generated to encrypt/decrypt all communication messages between the server and the user.

Let x denote the user secret key maintained by the registration center, and let y denote the server secret key maintained by the registration center. When the registration center permits the entry of a host computer, the registration center uses SID_j to compute the shared secret key $R_s = h(SID_j\|y)$, and sends $R_s = h(SID_j\|y)$ to server S_j via the secure channel. This shared key is used to confirm host legality. Before reviewing this authentication scheme, the following symbols are defined in Table 1.

2.1. User registration phase

The entire user registration phase is mainly executed on the server. When a user U_u wants to register and become a new legal user, the user U_u must submit his identity ID_u and password PW_u to the server via secure channel. The following steps are then performed during the registration phase.

Step 1: $U_u \rightarrow RC{:}ID_u, PW_u$
The user U_u sends their ID_u and PW_u to the registration center via a secure channel, and the registration center uses ID_u to compute $R_u = h(ID_u\|x)$, where x denotes the user secret key maintained by the registration center. Then uses R_u and PW_u to compute $C_0 = R_u \oplus h(PW_u)$.

Step 2: The registration center stores $h(\,)$ and C_0 into a smart card and issues this smart card to the user U_u via secure channel.

2.2. Login phase

When a legal user U_u wants to login to a server, the user U_u must insert smart card into a card reader, and enter identity ID_u and password PW_u. The reader generates a random nonce

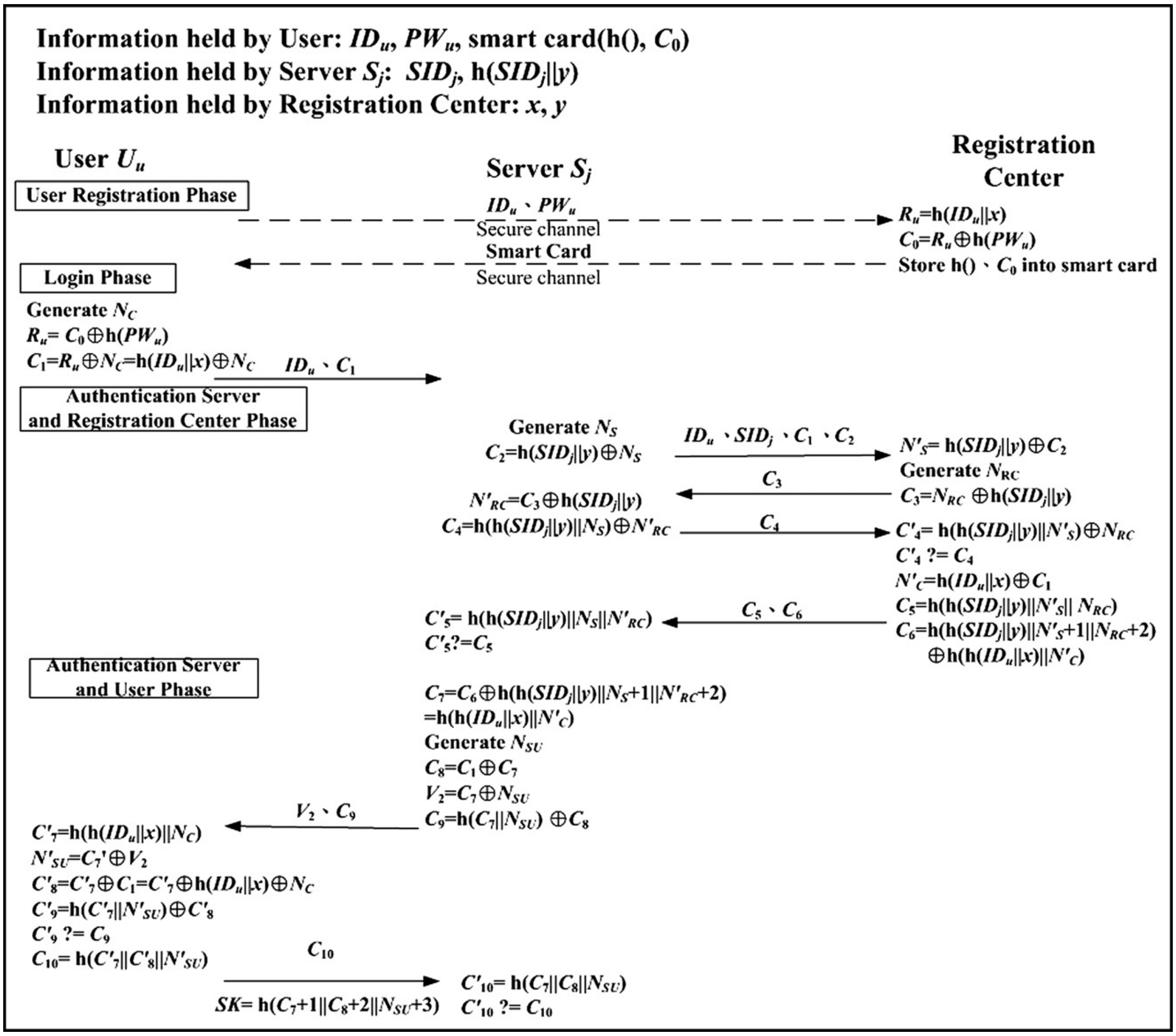
Information held by User: ID_u, PW_u, smart card(h(), C_0)
Information held by Server S_j: SID_j, h(SID_j||y)
Information held by Registration Center: x, y
User U_u
Server S_j
Registration Center
User Registration Phase
ID_u 、PW_u
Secure channel
R_u=h(ID_u||x)
$C_0=R_u \oplus$h(PW_u)
Smart Card
Secure channel
Store h() 、C_0 into smart card
Login Phase
Generate N_C
$R_u = C_0 \oplus$h(PW_u)
$C_1=R_u \oplus N_C$=h(ID_u||x)$\oplus N_C$
ID_u 、C_1
Authentication Server and Registration Center Phase
Generate N_S
C_2=h(SID_j||y)$\oplus N_S$
ID_u 、SID_j 、C_1 、C_2
N'_S= h(SID_j||y)$\oplus C_2$
Generate N_{RC}
C_3
$N'_{RC}=C_3 \oplus$h(SID_j||y)
$C_3=N_{RC} \oplus$h(SID_j||y)
C_4=h(h(SID_j||y)||N_S)$\oplus N'_{RC}$
C_4
C'_4= h(h(SID_j||y)||N'_S)$\oplus N_{RC}$
C'_4 ?= C_4
N'_C=h(ID_u||x)$\oplus C_1$
C_5=h(h(SID_j||y)||N'_S|| N_{RC})
C_6=h(h(SID_j||y)||N'_S+1||N_{RC}+2) $\oplus$h(h(ID_u||x)||N'_C)
C'_5= h(h(SID_j||y)||N_S||N'_{RC})
C_5 、C_6
C'_5?=C_5
Authentication Server and User Phase
$C_7=C_6 \oplus$h(h(SID_j||y)||N_S+1||N'_{RC}+2)
=h(h(ID_u||x)||N'_C)
Generate N_{SU}
$C_8=C_1 \oplus C_7$
$V_2=C_7 \oplus N_{SU}$
C_9=h(C_7||N_{SU}) $\oplus C_8$
V_2 、C_9
C'_7=h(h(ID_u||x)||N_C)
$N'_{SU}=C_7' \oplus V_2$
$C'_8=C'_7 \oplus C_1=C'_7 \oplus$h($ID_u$||x)$\oplus N_C$
C'_9=h(C'_7||N'_{SU})$\oplus C'_8$
C'_9 ?= C_9
C_{10}= h(C'_7||C'_8||N'_{SU})
C_{10}
C'_{10}= h(C_7||C_8||N_{SU})
SK= h(C_7+1||C_8+2||N_{SU}+3)
C'_{10} ?= C_{10}

Fig. 1 – Our proposed authentication scheme (the key $h(h(\mathrm{SID}_j \parallel y) \parallel N'_S + 1 \parallel N_{RC} + 2)$ do not generate).

N_C to change the transmitting data. The following steps are then performed during the login phase.

Step 1: $R_u = C_0 \oplus h(PW_u)$
When the user U_u enters identity ID_u and password PW_u, the reader extracts $h(ID_u \| x)$ from the smart card by computing $C_0 \oplus h(PW_u)$.

Step 2: $C_1 = h(ID_u \| x) \oplus N_C$
The reader generates a random nonce N_C, and uses N_C to compute $C_1 = h(ID_u \| x) \oplus N_C$.

Step 3: $U_u \rightarrow S_j$:ID_u, C_1
The reader sends ID_u and C_1 to server S_j.

2.3. *Authentication server and registration center phase*

To prevent the server S_j from knowing the $h(ID_u \| x)$ of a user, this study applied a method for protecting $h(ID_u \| x)$. The server S_j sends an alternative key $h(h(ID_u \| x) \| N_C)$ to help the server S_j verify the user U_u. The value of this key is changed with random nonce N_C. If an attacker steals a key in a session, this attacker could not use it in next session.

To avoid the cost of confirming the server S_j and registration center, the registration center and the server S_j could store the secret key $h(h(\mathrm{SID}_j \parallel y) \parallel N_S + 1 \parallel N'_{RC} + 2)$ for the user U_u temporarily in the system after successfully authenticating each other. If the user U_u wants to login the server S_j again, the server S_j just sends ID_u, SID_j, C_1 to the registration center in step 1, and the registration center retrieves N'_C and generates C_6 to transmit the user authentication key to server S_j in step 5.

Step 1: $S_j \rightarrow RC$:ID_u, SID_j, C_1, C_2
When the server S_j receives messages from the user U_u, it generates a random nonce N_S, and uses $h(SID_j \| y)$ to compute $C_2 = h(SID_j \| y) \oplus N_S$. The server S_j then sends ID_u, SID_j, C_1, and C_2 to the registration center.

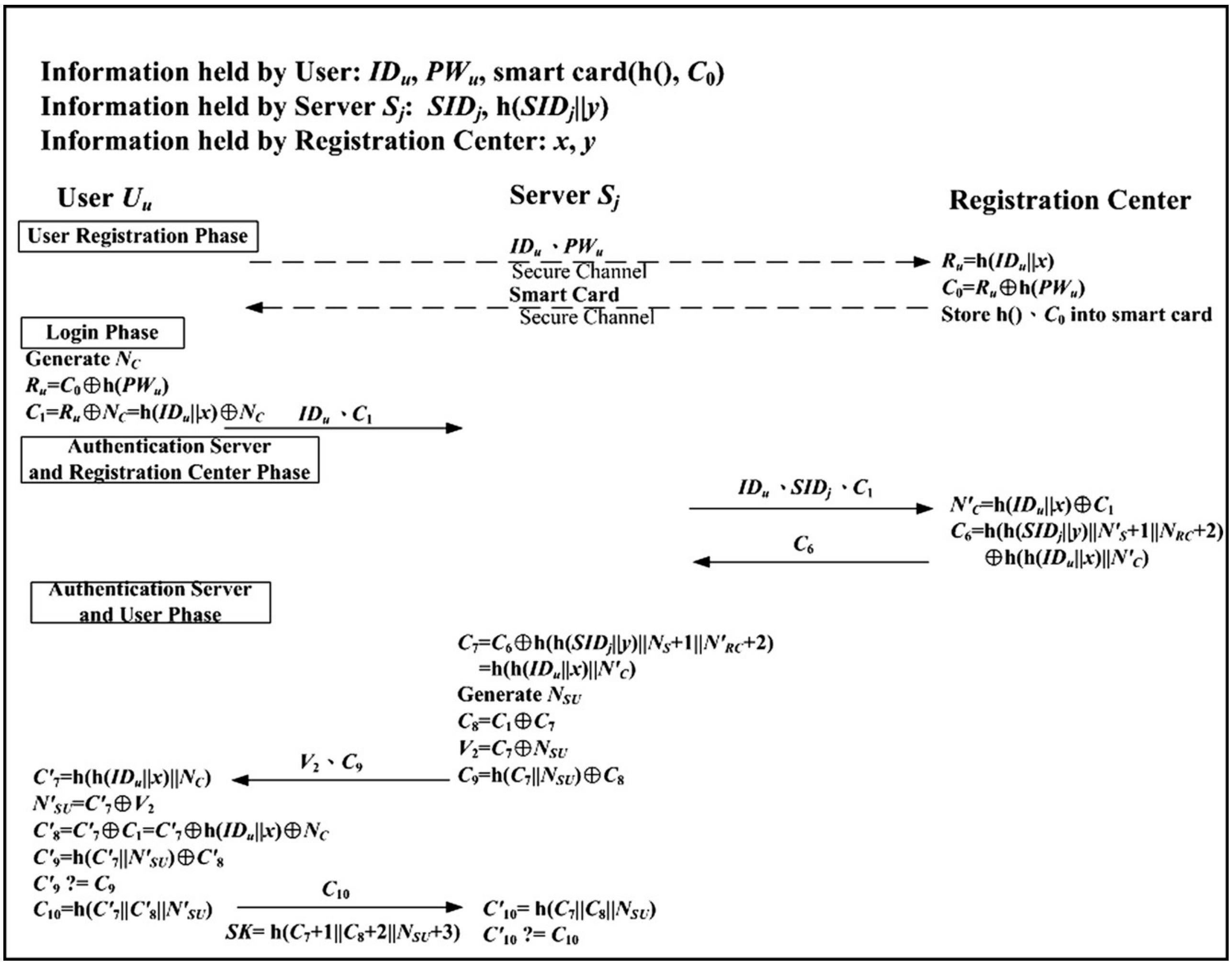
Information held by User: ID_u, PW_u, smart card(h(), C_0)
Information held by Server S_j: SID_j, h($SID_j||y$)
Information held by Registration Center: x, y
User U_u
Server S_j
Registration Center
User Registration Phase
ID_u 、PW_u
Secure Channel
R_u=h($ID_u||x$)
$C_0=R_u\oplus$h(PW_u)
Smart Card
Secure Channel
Store h() 、C_0 into smart card
Login Phase
Generate N_C
$R_u=C_0\oplus$h(PW_u)
$C_1=R_u\oplus N_C$=h($ID_u||x$)$\oplus N_C$
ID_u 、C_1
Authentication Server and Registration Center Phase
ID_u 、SID_j 、C_1
N'_C=h($ID_u||x$)$\oplus C_1$
C_6=h(h($SID_j||y$)$||N'_S+1||N_{RC}+2$)$\oplus$h(h($ID_u||x$)$||N'_C$)
C_6
Authentication Server and User Phase
$C_7=C_6\oplus$h(h($SID_j||y$)$||N_S+1||N'_{RC}+2$)
=h(h($ID_u||x$)$||N'_C$)
Generate N_{SU}
$C_8=C_1\oplus C_7$
$V_2=C_7\oplus N_{SU}$
C_9=h($C_7||N_{SU}$)$\oplus C_8$
V_2 、C_9
C'_7=h(h($ID_u||x$)$||N_C$)
$N'_{SU}=C'_7\oplus V_2$
$C'_8=C'_7\oplus C_1=C'_7\oplus$h($ID_u||x$)$\oplus N_C$
C'_9=h($C'_7||N'_{SU}$)$\oplus C'_8$
C'_9 ?= C_9
C_{10}=h($C'_7||C'_8||N'_{SU}$)
C_{10}
C'_{10}= h($C_7||C_8||N_{SU}$)
C'_{10} ?= C_{10}
SK= h($C_7+1||C_8+2||N_{SU}+3$)

Fig. 2 – Our proposed authentication scheme (the key $h(h(\mathrm{SID}_j \parallel y) \parallel N'_S+1 \parallel N_{RC}+2)$ has generated).

Step 2: RC → S_j:C_3
Upon receiving messages from the server S_j, the registration center uses C_2, SID_j and y to retrieve N'_s by computing $C_2 \oplus h(\mathrm{SID}_j||y)$. Then the registration center generates a random nonce N_{RC} to compute $C_3 = N_{RC} \oplus h(\mathrm{SID}_j||y)$, and sends C_3 to the server S_j.

Step 3: S_j → RC:C_4
When the server S_j receives C_3 from the registration center, the server S_j uses C_3 and $h(\mathrm{SID}_j||y)$ to retrieve N'_{RC} by computing $C_3 \oplus h(\mathrm{SID}_j||y)$. Meanwhile, the server S_j uses N'_{RC}, N_S, and $h(\mathrm{SID}_j||y)$ to compute $C_4 = h(h(\mathrm{SID}_j \parallel y) \parallel N_S) \oplus N'_{RC}$ and sends C_4 to the registration center.

Step 4: $C'_4 ?= C_4$
When the registration center receives C_4 from the server S_j, it uses N_{RC}, N'_S, $h(\mathrm{SID}_j||y)$ to compute $C'_4 = h(h(\mathrm{SID}_j \parallel y) \parallel N'_S) \oplus N_{RC}$ and checks whether $C'_4 ?= C_4$. If they are equal, the server S_j is correct, and the following steps are performed. If they are not equal, the server S_j is not correct and the communication is rejected.

Step 5: RC → S_j:C_5, C_6
The registration center uses $h(\mathrm{ID_u}||x)$ and C_1 to retrieve N'_C by computing $h(\mathrm{ID_u}||x) \oplus C_1$. The registration center then uses $h(\mathrm{SID}_j||y)$, N'_S, N_{RC} and N'_S to compute $C_5 = h(h(\mathrm{SID}_j \parallel y) \parallel N'_S \parallel N_{RC})$ and $C_6 = h(h(\mathrm{SID}_j \parallel y) \parallel N'_S + 1 \parallel N_{RC} + 2) \oplus h(h(\mathrm{ID_u} \parallel x) \parallel N'_C)$, and sends C_5 and C_6 to server S_j. C_5 is used to verify the identity of registration center. C_6 is used to transmit the user authentication key to the server S_j. $h(h(\mathrm{SID}_j \parallel y) \parallel N'_S + 1 \parallel N_{RC} + 2)$ is the secret key between the server S_j and the registration center.

Table 1 – Symbol explanation

Symbol	Explanation
$h()$	One-way hash function
$\oplus$	Exclusive or
ID, PW	The remote user identity and password
SID	The server's identity
RC	The registration center
$\|\|$	Concatenation
N	The random nonce
X → Y:M	X sends a message M to Y
U_i, S_j	Each represents as ith user and jth server

The key is used to protect the user authentication key $h(h(\mathrm{ID_u} \| x) \| N'_C)$.

Step 6: When the server S_j receives C_5 and C_6 from the server S_j, it computes $C'_5 = h(h(\mathrm{SID}_j \| y) \| N_S \| N'_{RC})$ to verify whether $C'_5 ? = C_5$. If these two variables are equal, the registration center is legal. If they are not equal, the registration center is not legal, and the process ends.

2.4. *Authentication server and user phase*

When the server S_j receives C_6 from registration center, the server S_j uses $h(h(\mathrm{SID}_j \| y) \| N_S + 1 \| N'_{RC} + 2)$ and C_6 to compute the user authentication key $h(h(\mathrm{ID_u} \| x) \| N'_C)$. Then the server S_j could use the user authentication key to authenticate the remote user.

Step 1: $S_j \rightarrow U_u{:}V_2, C_9$

After successfully verifying the registration center, the server S_j must get the user authentication key. The server S_j computes $C_7 = C_6 \oplus h(h(\mathrm{SID}_j \| y) \| N_S + 1 \| N'_{RC} + 2) = h(h(\mathrm{ID_u} \| x) \| N'_C)$ and generates a random nonce N_{SU}. Here C_7 represents the user authentication key. This study uses C_7 to compute $C_8 = C_7 \oplus C_1 = C_7 \oplus h(\mathrm{ID_u} \| x) \oplus N_C$, and the server S_j uses N_{SU}, C_7, C_8 to compute $V_2 = C_7 \oplus N_{SU}$ and $C_9 = h(C_7 \| N_{SU}) \oplus C_8$. Send V_2 and C_9 to the user U_u.

Step 2: $C'_9 ? = C_9$

After receiving V_2 and C_9 from the server S_j, the user U_u first uses $h(\mathrm{ID_u} \| x)$ and N_C to compute the user authentication key $C'_7 = h(h(\mathrm{ID_u} \| x) \| N_C)$, as well as $N'_{SU} = C'_7 \oplus V_2$ and $C'_8 = C'_7 \oplus C_1 = C'_7 \oplus h(\mathrm{ID_u} \| x) \oplus N_C$. The user U_u then uses N'_{SU}, C'_7 and C'_8 to compute $C'_9 = h(C'_7 \| N'_{SU}) \oplus C'_8$, and compares C_9 and C'_9. If they are equal, the server S_j is legal, in which case the following steps should be executed. If they are not equal, the user U_u will reject the following steps.

Step 3: $U_u \rightarrow S_j{:}C_{10}$

The user U_u computes $C_{10} = h(C'_7 \| C'_8 \| N'_{SU})$ and sends it to server S_j.

Step 4: $C'_{10} ? = C_{10}$

When the server S_j receives C_{10} from the user U_u, the server S_j computes $C'_{10} = h(C_7 \| C_8 \| N_{SU})$, and checks whether $C'_{10} ? = C_{10}$. If the two values are not equal, the user U_u is not a legal user. If they are equal, the user U_u is a legal user. Then they will define a session key $SK = h(C_7 + 1 \| C_8 + 2 \| N_{SU} + 3)$. The session key is applied to encrypt all following communications between the server S_j and the user U_u.

3. Security analysis

The proposed authentication scheme is efficient, because it only uses one-way hash function. The servers and the registration center do not store any verification table. The following examines whether the authentication scheme is safe, and considers its ability to resist various known attacks.

3.1. *Replay attack*

The proposed authentication scheme uses random nonce to withstand replay attack. Random nonces N_C, N_S, N_{SU}, and N_{RC} are generated independently, and their values differ among sessions. Thus attackers cannot enter the system by resending messages previously transmitted by legal users.

3.2. *Stolen-verifier attack*

Because the servers and the registration center do not store any verification table, the proposed authentication scheme is secure against stolen-verifier attack.

3.3. *Server spoofing*

It is impossible for an attacker to masquerade as the server to cheat a remote user or the registration center. Because none of the servers store any user authentication key in it, none of the servers can authenticate the remote user. If the server wants to authenticate, the server must be authenticated by the registration center first, and then obtain the user authentication key from the registration center. If an attacker wishes to cheat the registration center, they must have $h(\mathrm{SID}_j \| y)$. The proposed authentication scheme can prevent server spoofing.

3.4. *Security of session key*

A session key is generated from $h(h(\mathrm{ID_u} \| x) \| N_C)$, N_{SU}, and N_C. These parameter values are different in each session, and each is only known by the server and the user. Whenever the communication ends between the user and the server, the key will immediately self-destruct and will not be reused. When the user re-enters the system, a new session key will be generated for encrypting all the messages between the server and the registration center. Therefore assuming the attacker has obtained a session key, the user will be unable to use this session key to decode the information in other communication processes. Because the random nonce N_{SU} and N_C are both generated randomly, a known session key cannot be used to calculate the value of the next session key. Additionally, since the value of the nonces is very large, attackers cannot directly guess the value of the nonces to generate session key.

3.5. *Registration center spoofing*

It is impossible for an attacker to masquerade as the registration center, because every server S_j has a $h(\mathrm{SID}_j \| y)$. The server can use $h(\mathrm{SID}_j \| y)$ to verify the identity of the registration center, and thus the proposed scheme can prevent registration center spoofing.

3.6. *Security of the user authentication key*

The user authentication key $h(h(\mathrm{ID_u} \| x) \| N_C)$ is used to help the server authenticate the remote user. When the user re-enters the system, the user authentication key is regenerated during the authentication process. Therefore assuming the attacker has obtained a user authentication key by cracking the server, an attacker will be unable to use this key to authenticate the user in other authentication processes. Because the random nonce N_C is generated randomly and very large, it is impossible to use a known user authentication key to calculate the value of the next user authentication key.

3.7. *Impersonation attack*

The proposed authentication scheme makes it impossible for an attacker to masquerade as a legal user. To successfully perform an impersonation attack, the attacker requires $h(ID_u \| x)$ to generate authentication messages correctly. All authentication messages between the server and the user are protected by $h(ID_u \| x)$. To avoid the server get $h(ID_u \| x)$, the registration center sends the user authentication key $h(h(ID_u \| x) \| N_C)$ to the server. The user authentication key $h(h(ID_u \| x) \| N_C)$ is generated by the random nonce N_C, so this key is different in each authentication process. When an attacker obtains a user authentication key in an authentication process, this attacker can use this authentication key to masquerade as a legal user.

4. Comparison with other two authentication schemes

This section compares the proposed multi-server authentication scheme with two other multi-server authentication schemes. The two comparison schemes were the multi-server authentication scheme using symmetrical encryption methods proposed by Juang (2004) and Chang and Lee (2004), respectively. Efficiency comparisons between our proposed scheme and other schemes are summarized in Table 2.

Even though the communication costs of the registration center of Chang and Lee multi-server authentication scheme are lower than our proposed scheme and authentication scheme of Juang, but the multi-server authentication scheme of Chang and Lee assumes that the registration center and all system servers are reliable, and the registration center sends each server share the secret key Kx via a secure channel. Sharing the same secret key enables each server to masquerade as other servers or as the registration center in the real network environments. This multi-server authentication scheme thus is vulnerable to security breaches arising from server spoofing and registration center spoofing. The proposed authentication scheme and authentication scheme of Juang do not suffer from server spoofing problem and registration spoofing problem.

These two multi-server authentication schemes and the proposed multi-server authentication scheme do not store any verification table in the server and only use nonce, preventing the problems of stolen-verifier attack and time synchronization. Notably, in the multi-server authentication scheme of Juang, the entire $h(x, ID_u)$ uses passwords to keep secrets by exclusive-or operation, and stores into memory of a smart card. PW must be memorized by the remote user, but it is impossible to request users to memorize the long and meaningless password. The authentication scheme of Juang is not user friendly. The authentication scheme of Chang and Lee and the proposed authentication scheme use one-way hash function to encrypt the user password PW_u, and then uses $h(PW_u)$ to keep secrets by exclusive-or operation, enabling the remote user to memorize it. To prevent the server S_j from knowing the $h(ID_u \| x)$ of a user, this study applied a method for protecting $h(ID_u \| x)$. The server S_j sends an alternative key $h(h(ID_u \| x) \| N_C)$ to help the server S_j verify the user U_u, but the two other schemes do not apply such mechanism. All comparisons between our proposed scheme and two other schemes are described in Table 3.

Table 2 – Efficiency comparisons between our proposed scheme and related schemes

		Our		Juang	Chang et al.
		No	Yes		
Communication cost of user registration		2H	2H	1H	2H
Communication cost of server registration		1H	1H	1H	0
Communication costs of authentication	User	5H	5H	3H, 3Sym	3H, 3Sym
	Server	5H	3H	3H, 4Sym	3H, 3Sym
	RC	6H	2H	1H, 2Sym	0

H: one-way hash function; Sym: symmetric cryptosystems; No: the key $h(h(SID_j \| y) \| N'_S + 1 \| N_{RC} + 2)$ do not generate; and Yes: the key $h(h(SID_j \| y) \| N'_S + 1 \| N_{RC} + 2)$ has generated.

5. Conclusions

Identifying remote users as legal or illegal is a key issue in network security. This study proposes a multi-server authentication scheme using one-way hash function. The proposed authentication scheme is different from other such schemes. The proposed scheme is based on the nonce and one-way hash function, and thus can be used in the distributed network environment. Because this study only uses the one-way hash function in the proposed authentication scheme, problems associated with the cost of computation can be avoided. This future research will soon apply the proposed scheme to equipment with low computational capabilities. However, the safety of using one-way hash function as a base for authentication depends on the characteristics of the one-way hash function. Thus it is necessary to select a safer one-way hash function, since the MD5 one-way hash function was cracked

Table 3 – Comparisons between our proposed scheme and other schemes

	Our	Juang	Chang et al.
C1	Yes	Yes	Yes
C2	Yes	Yes	Yes
C3	Yes	Yes	No
C4	Yes	Yes	No
C5	Yes	No	Yes
C6	Yes	No	No

C1: no verification table; C2: single registration; C3: prevention of server spoofing; C4: prevention of registration center spoofing; C5: user friendly; and C6: the user authentication key is dynamic.

by Wang and Yu (2000) from China Shan Dong University. The more secure one-way hash function could be used for system protection.

REFERENCES

Chang CC, Lee JS. An efficient and secure multi-server password authentication scheme using smart cards. In: International conference on cyberworlds; November 2004. p. 417–22.

Hwang T, Chen Y, Laih CS. Non-interactive password authentication without password tables. In: IEEE region 10 conference on computer and communication system, vol. 1; September 1990. p. 429–31.

Juang WS. Efficient multi-server password authenticated key agreement using smart cards. IEEE Transactions on Consumer Electronics November 2004;50(1):251–5.

Lamport L. Password authentication with insecure communication. Communications of the ACM November 1981;24(11):770–2.

Li LH, Lin IC, Hwang MS. A remote password authentication scheme for multi-server architecture using neural networks. IEEE Transactions on Neural Network November 2001;12(6): 1498–504.

Lin IC, Hwang MS, Li LH. A new remote user authentication scheme for multi-server architecture. Future Generation Computer System January 2003;19:13–22.

Sun HM. An efficient remote use authentication scheme using smart cards. IEEE Transactions on Consumer Electronics November 2000;46(4):958–61.

Tsaur WJ, Wu CC, Lee WB. A smart card-based remote scheme for password authentication in multi-server Internet services. Computer Standard & Interfaces 2004;27:39–51.

Wang XY, Yu HG. How to break MD5 and other hash functions. Eurocrypt 2000:19–35.