Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Efficient password authentication schemes based on a geometric approach for a multi-server environment

Horng-Twu LIAW[†1], Chih-Ta YEN[2], Meng-Yu CHIU[1], Li-Lin HSIAO[1]

([1]*Department of Information Management, Shih Hsin University, Taiwan 116, Taipei*)

([2]*Department of Information Management, National Taiwan University of Science and Technology, Taiwan 106, Taipei*)

[†]E-mail: htliaw@cc.shu.edu.tw

Received Nov. 17, 2009; Revision accepted May 10, 2010; Crosschecked Aug. 14, 2010; Published online Sept. 7, 2010

Abstract: Password authentication schemes based on a geometric approach are proposed. Based on geometrics such as a plane or space, the proposed schemes can be used to fulfill the various requirements for security and achieve more efficient results. They allow a user to register with many servers once without remembering different passwords for each, and also allow a user to change the password freely and update it offline. In addition, the proposed schemes do not need to maintain a verification table and are also computationally efficient. Moreover, we propose a specific access right (AR) to verify the legitimate user who has different authorization levels on a server in a multi-server environment.

Key words: Authentication, Access right, Security, Geometry
doi:10.1631/jzus.C0910712 **Document code:** A **CLC number:** TP309

1 Introduction

In recent years, with the rapid growth of computer networks, more and more computers at different locations are connected to serve users. In such an environment, it becomes very important to authenticate users over insecure networks and manage users' passwords at various servers. Today, a variety of authentication methods based on passwords have been published in the literature to address this problem.

Since Lamport (1981) proposed a remote password authentication scheme over an insecure channel, various schemes (Lin *et al*., 2003; Juang, 2004; Yoon *et al*., 2004; Tsaura *et al*., 2005; Tsai, 2008; Liao *et al*., 2009; Liao and Wang, 2009; Rhee *et al*., 2009; Hwang *et al*., 2010) have been proposed to address this problem, and have achieved more functionality and efficiency. For remote login systems, user authentication is the necessary security requirement. Among the numerous user authentication schemes, the password scheme is the most convenient and has been widely adopted. In a password scheme, the verification table has the risk of being modified because the passwords are stored in a remote system. In this paper, we propose password authentication schemes based on a geometric approach. The schemes can be used to fulfill the various requirements for security and are more efficient by using geometrics such as a plane or space. They allow a user to register with many servers once without needing to remember different passwords for each, and also allow a user to change a password freely and update it offline. In addition, the proposed schemes do not need to maintain a verification table and are computationally efficient. Moreover, we propose a specific access right (AR) to verify the legitimate user who has different authorization levels on a server in a multi-server environment.

2 Our proposed scheme

In this section, we first introduce the requirements of password authentication. Later, we describe

the notations of our schemes and then present two password authentication schemes for different requirements of security. A password authentication scheme is secure and efficient for a multi-server environment if the following requirements are satisfied (Liao and Wang, 2009):

(R1) Single registration: It allows a valid user to register once and then the user can access all the registered servers.

(R2) No verification table: No server needs to maintain a verification table.

(R3) Update password securely and freely: A valid user can change his/her password freely offline.

(R4) Mutual authentication and key management: Valid users and valid servers can authenticate each other and then agree on a session key to protect the message in the transmission process.

(R5) Security: A password authentication scheme must resist all kinds of attack to be realistic. We catalogue the common attacks on a password authentication scheme as follows:

(S1) Eavesdropping attack: An adversary can obtain the authentication information via eavesdropping in the network.

(S2) Stolen verifier attack: An adversary can steal the password table from the server if the server maintains a verification table. After stealing the verification table, the adversary can masquerade as a legitimate user.

(S3) Denial of service (DoS) attack: This attack is to make the service or resource of the server unavailable for valid users.

(S4) Impersonation attack: An adversary sends a message that is changed by the adversary to a legitimate party and claims that the message comes from a valid sender.

(S5) Replay attack: An adversary records the exchanged message and then transmits it to the legitimate party to obtain the authentication information.

(S6) Server spoofing attack: This attack means that an adversary masquerades as a legitimate server and tries to cheat a legitimate user using a valid response message as a fake.

(S7) Security of the session key: This means that even if a session key is disclosed, an adversary could still not obtain any sensitive message from the other sessions.

(S8) Smart card stolen: A password cannot be retrieved even if the smart card is lost or stolen.

(S9) Offline password guessing attack: This attack is to break a password by generating a set of passwords to generate the same parameter that is bounded with a password.

In our schemes, let CM denote the control manager, ID the public user identity of a user, PW a secret weak password of a user, and '$X \rightarrow Y$: Z' that a sender X sends a message Z to a receiver Y. Besides, '||' denotes the conventional string concatenation operator and $H(\cdot)$ a strong one-way hash function. Our schemes are divided into four phases: (1) registration, (2) login, (3) authentication and session key agreement, and (4) change of password. Initially, CM has its public key p_{cm} and chooses a random secret key d_i for each server Server_i, $1 \le i \le m$. Suppose there are m servers with which the new user can register. In the registration phase, a new user chooses his/her own ID and PW, and sends it to CM. According to the rights of the users, CM creates an AR for one user that can access servers in a multi-server environment. In the login phase, when a legitimate user wants to log in the servers, he/she must enter the ID and PW, and generate related arguments for verification. Then the servers verify the legitimacy and ARs of the login user in the authentication phase.

2.1 Password authentication scheme on a plane

Details of the scheme for a multi-server architecture are described as follows.

2.1.1 Registration phase

When a new user would like to log in a multi-server computer system, the user must register with the CM and perform the following steps:

1. User→CM: ID, PW

The new user chooses his/her own ID and PW, and then uses CM's public key p_{cm} to encrypt the register information ID and PW. Then the new user sends the ciphertext to CM.

2. CM→User: ID, P, M_i, N_i ($1 \le i \le m$)

Once CM receives ID and PW, CM computes the values as follows:

$$T_i=\text{AR}_i||H(\text{ID}||\text{AR}_i, d_i)||H(\text{ID}||\text{AR}_i, \text{PW}),$$
$$S_i=H(\text{ID}, d_i),\ U=H(\text{ID}, \text{PW}).$$

CM chooses a plane P randomly. Here P: $Z=F(X, Y)=aX+bY+c$. According to the plane P, S_i, and U, CM computes the M_i value as follows:

$$Z=F(X, Y) \Rightarrow S_i=F(U, M_i)$$
$$\Rightarrow S_i=aU+bM_i+c$$
$$\Rightarrow bM_i=S_i-aU-c$$
$$\Rightarrow M_i=(S_i-aU-c)/b.$$

Similarly, according to the plane P, T_i, CM computes the N_i value as follows:

$$Z=F(X, Y) \Rightarrow T_i=F(U, N_i)$$
$$\Rightarrow T_i=aU+bN_i+c$$
$$\Rightarrow bN_i=T_i-aU-c$$
$$\Rightarrow N_i=(T_i-aU-c)/b.$$

Finally, CM delivers the public parameters ID, P, M_i, and N_i to the user. These parameters can be stored in a smart card or other storage device. The concept of the registration phase is shown in Fig. 1.

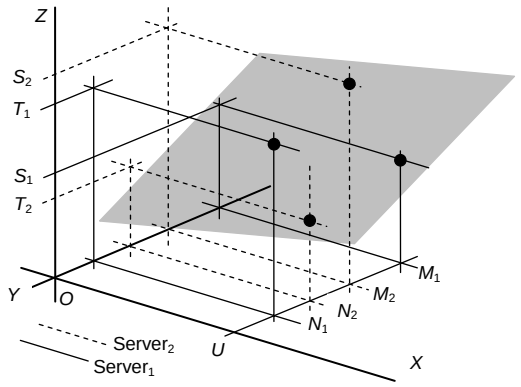

Fig. 1 The registration phase on a plane

2.1.2 Login phase

In this phase, users can be authorized for multiple servers once through the password authentication and then the system will perform the following steps:

1. User→Server$_i$ ($1\le i\le m$): ID, B_i, P, TS

The user first keys in his/her ID and PW, and then computes $U=H(\text{ID, PW})$. According to the plane P, M_i, and U, the system calculates S_i as follows:

$$Z=F(X, Y) \Rightarrow S_i=F(U, M_i)$$
$$\Rightarrow S_i=aU+bM_i+c. \quad (1)$$

Similarly, the system calculates T_i according to the plane P, N_i, and U as follows:

$$Z=F(X, Y) \Rightarrow T_i=F(U, N_i)$$
$$\Rightarrow T_i=aU+bN_i+c. \quad (2)$$

Continuously, the user obtains a time sequence TS, which is similar to a timestamp from the terminal, computes the jth session key $H(T_i, \text{TS})$, $A_i=H(S_i, \text{TS})$, calculates B_i according to the plane P, T_i, and A_i as follows:

$$Z=F(X, Y) \Rightarrow T_i=F(A_i, B_i)$$
$$\Rightarrow T_i=aA_i+bB_i+c$$
$$\Rightarrow bB_i=T_i-aA_i-c$$
$$\Rightarrow B_i=(T_i-aA_i-c)/b,$$

and then sends the public message ID, B_i, P, and TS to all servers.

2.1.3 Authentication and session key agreement phase

In the authentication phase, servers receive the message that was sent in the login phase at TS′.

1. Server$_i$→User ($1\le i\le m$): Accept or Reject, R_i, TS″

Servers check whether the time interval between TS and TS′ is greater than ΔTS, the expected legitimate time interval for the transmission delay between the login terminal and the server Server$_i$. If TS′−TS≥ΔTS, servers reject the login request; otherwise, servers check the correctness of ID. If the format of ID is incorrect, servers will also reject the login request.

Once the time interval of TS′ and the format of ID are valid, the server computes $S_i=H(\text{ID}, d_i)$ and then obtains $A_i=H(S_i, \text{TS})$. According to the plane P, A_i, and B_i, the server calculates T_i as follows:

$$Z=F(X, Y) \Rightarrow T_i=F(A_i, B_i)$$
$$\Rightarrow T_i=aA_i+bB_i+c.$$

After Server$_i$ checks whether the equation

$$H(\text{ID}||\text{AR}_i, d_i)'=H(\text{ID}||\text{AR}_i', d_i)$$

holds, Server$_i$ retrieves AR_i' and $H(\text{ID}||\text{AR}_i, d_i)'$ with T_i. If the above equation does not hold, Server$_i$ rejects the login request; otherwise, the request proceeds to the next step.

Server$_i$ checks whether the equation

$$T_i=\text{AR}_i'||H(\text{ID}||\text{AR}_i, d_i)'||H(\text{ID}||\text{AR}_i', \text{PW})$$

holds. If yes, Server$_i$ accepts the login request and recognizes AR_i' as the access right of the user in Server$_i$.

Continuously, Server$_i$ computes the jth session key $H(T_i, \mathrm{TS})$ as the user, and then computes R_i with $R_i=H(T_i, \mathrm{TS}'')$ as follows:

$$\begin{aligned} Z=F(X, Y) &\Rightarrow T_i=F(U, R_i)\\ &\Rightarrow T_i=aU+bR_i+c\\ &\Rightarrow bR_i=T_i-aU-c\\ &\Rightarrow R_i=(T_i-aU-c)/b, \end{aligned}$$

and then returns R_i and TS″.

2. User→Server$_i$ ($1\le i\le m$): Accept or Reject services

The user receives the above message at TS‴ and then checks whether the time interval between TS″ and TS‴ is greater than ΔTS′, the expected legitimate time interval for the transmission delay between the login terminal and the server Server$_i$. If TS‴−TS″≥ ΔTS′, the user will reject the services from Server$_i$.

Once the time interval of TS‴ is valid, the user checks $R_i'=(T_i-aU-c)/b$. If it does not hold, the user rejects the services from Server$_i$. Hence, mutual authentication is provided between the user and the server.

2.1.4 Changing password phase

In our proposed scheme, users can choose and change their passwords freely offline. When a user wants to change his/her password at server Server$_i$, the steps are as follows:

1. The user should type the old password PW and the new password PW′.

2. The user computes $U=H(\mathrm{ID}, \mathrm{PW})$ from the old password PW.

3. The user calculates S_i according to the plane P, M_i, and U as follows:

$$\begin{aligned} Z=F(X, Y) &\Rightarrow S_i=F(U, M_i)\\ &\Rightarrow S_i=aU+bM_i+c. \end{aligned}$$

4. The user calculates T_i according to the plane P, N_i, and U as follows:

$$\begin{aligned} Z=F(X, Y) &\Rightarrow T_i=F(U, N_i)\\ &\Rightarrow T_i=aU+bN_i+c. \end{aligned}$$

5. The user retrieves AR_i' and $H(\mathrm{ID}||\mathrm{AR}_i, d_i)'$ with T_i. Then the user checks whether the equation

$$T_i=\mathrm{AR}_i'||H(\mathrm{ID}||\mathrm{AR}_i, d_i)'||H(\mathrm{ID}||\mathrm{AR}_i', \mathrm{PW})$$

holds. If the equation does not hold, the system rejects the change of the password request.

6. The user computes $U'=H(\mathrm{ID}, \mathrm{PW}')$ from the new password PW′.

7. The user chooses a new plane P_{new} at random. Here,

$$P_{\mathrm{new}}: Z=F'(X, Y)=a'X+b'Y+c'.$$

8. The user calculates M_i' according to the plane P_{new}, S_i, and U' as follows:

$$\begin{aligned} Z=F(X, Y) &\Rightarrow S_i=F(U', M_i')\\ &\Rightarrow S_i=aU'+bM_i'+c\\ &\Rightarrow bM_i'=S_i-aU'-c\\ &\Rightarrow M_i'=(S_i-aU'-c)/b. \end{aligned}$$

9. The user calculates N_i' according to the plane P_{new}, T_i, and U' as follows:

$$\begin{aligned} Z=F(X, Y) &\Rightarrow T_i=F(U', N_i')\\ &\Rightarrow T_i=aU'+bN_i'+c\\ &\Rightarrow bN_i'=T_i-aU'-c\\ &\Rightarrow N_i'=(T_i-aU'-c)/b. \end{aligned}$$

10. Finally, the system updates the P, M_i, and N_i into P_{new}, M_i', and N_i', respectively.

2.2 Password authentication scheme on a space

In this subsection, we propose another efficient remote user authentication scheme based on space geometry. Details of the scheme for a multi-server architecture are described as follows.

2.2.1 Registration phase

When a new user wants to log in a multi-server computer system, the user must register with the servers.

1. User→CM: ID, PW

The new user chooses his/her own ID and PW, and then uses CM's public key p_{cm} to encrypt the register information ID and PW. Then the new user sends the ciphertext to CM.

2. CM→User: L_i, h_{1i}, h_{2i} ($1\le i\le m$)

Once CM receives the ciphertext, he/she uses his/her private key d_{cm} to decrypt it to obtain ID and PW. Then, CM generates a random number r_i. Suppose the length of r_i is 512 bits. CM computes a point $P_i=(P_{1i}, P_{2i}, P_{3i})$ as follows:

$$P_{1i}=H(\mathrm{ID}, d_i),\ P_{2i}=H(\mathrm{ID}, \mathrm{PW}),\ P_{3i}=r_i||\mathrm{AR}_i.$$

AR_i means that the legitimate user has different authorization levels for the different servers in a multi-server environment. CM constructs a line L_i with the origin (0, 0) and the point P_i. Here, L_i: ($f_i(Z)$, $g_i(Z)$, Z), $f_i(Z)=a_iZ+b_i$, $g_i(Z)=c_iZ+e_i$. Finally, CM computes the values as follows:

$$h_{1i}=H(\mathrm{ID}||\mathrm{AR}_i, d_i),\ \ h_{2i}=H(\mathrm{ID}||\mathrm{AR}_i, \mathrm{PW}),$$

and then sends the public parameters L_i, h_{1i}, and h_{2i} to the user. These parameters can be stored in a smart card or other storage device. The concept of the registration phase is shown in Fig. 2.

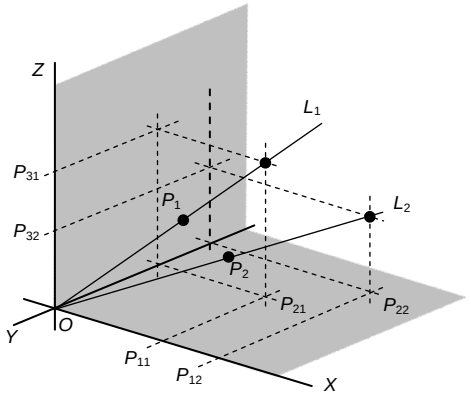

Fig. 2 The concept of the registration phase

2.2.2 Login phase

In this phase, users are authorized to use multiple servers once they have made their way through the password authentication. The following protocol is for the user's jth login.

1. User→Server$_i$ ($1\le i\le m$): ID, P_{vi}, h_{1i}, TS

The user first keys in his/her ID and PW, and then reconstructs a line L_i according to two points, P_i and the origin (0, 0). Here, L_i: ($f_i(Z)$, $g_i(Z)$, Z), $f_i(Z)=a_iZ+b_i$, $g_i(Z)=c_iZ+e_i$. Once the user retrieves L_i, he/she computes $P_{2i}=H(\mathrm{ID}, \mathrm{PW})$ and then obtains P_{3i} and P_{1i} as follows:

$$\begin{aligned} g_i(Z)=c_iZ+e_i &\Rightarrow P_{2i}=g_i(P_{3i})=c_iP_{3i}+e_i \\ &\Rightarrow P_{3i}=(P_{2i}-e_i)/c_i. \qquad (3)\\ f_i(Z)=a_iZ+b_i &\Rightarrow P_{1i}=f(P_{3i})=a_iP_{3i}+b_i \\ &\Rightarrow P_{1i}=a_i(P_{2i}-e_i)/c_i+b_i. \qquad (4)\end{aligned}$$

Continuously, the user obtains a time sequence TS, which is like a timestamp from the terminal, computes the jth session key $H(P_{1i}, P_{2i}, \mathrm{TS})$, $B=H(P_{1i}, \mathrm{TS})$, and selects a new point $P_{vi}=(P_{v1i}, P_{v2i}, P_{v3i})=(f_i(B), g_i(B), B)$ as follows:

$$\begin{aligned} P_{v1i} &\Rightarrow f_i(Z)=a_iZ+b_i=a_iB+b_i=a_iH(P_{1i},\mathrm{TS})+b_i, \qquad (5)\\ P_{v2i} &\Rightarrow g_i(Z)=c_iZ+e_i=c_iB+e_i=c_iH(P_{1i},\mathrm{TS})+e_i, \qquad (6)\\ P_{v3i} &\Rightarrow B=H(P_{1i}, \mathrm{TS}), \qquad (7)\end{aligned}$$

and then sends the public message ID, P_{vi}, h_{1i}, and TS to all servers.

2.2.3 Authentication and session key agreement phase

In the authentication phase, servers receive the message that was sent in the login phase at TS′ and P_{vi}.

1. Server$_i$→User ($1\le i\le m$): Accept or Reject, P_{si}, TS″

Servers check whether the time interval between TS and TS′ is greater than ΔTS, the expected legitimate time interval for the transmission delay between the login terminal and the server Server$_i$. If TS′−TS≥ΔTS, servers will reject the login request. If accepted, servers check the correctness of ID. If the format of ID is incorrect, servers will also reject the login request.

Once the time interval of TS′ and the format of ID are valid, servers retrieve P_{v3i}' with the received P_{vi} and check whether the equation

$$P_{v3i}'=H(H(\mathrm{ID}, d_i), \mathrm{TS})$$

holds. If yes, servers accept the login request.

According to two points, P_{vi} and the origin (0, 0), servers reconstruct a line L_i: ($f_i(Z)$, $g_i(Z)$, Z), where $f_i(Z)=a_iZ+b_i$, $g_i(Z)=c_iZ+d_i$. Once servers retrieve L_i, they compute $P_{1i}=H(\mathrm{ID}, d_i)$ and then obtain P_{3i} and P_{2i} as follows:

$$\begin{aligned} f_i(Z)=a_iZ+b_i &\Rightarrow P_{1i}=f_i(P_{3i})=a_iP_{3i}+b_i \\ &\Rightarrow P_{3i}=(P_{1i}-b_i)/a_i. \\ g_i(Z)=c_iZ+e_i &\Rightarrow P_{2i}=g(P_{3i})=c_iP_{3i}+e_i \\ &\Rightarrow P_{2i}=c_i(P_{1i}-b_i)/a_i+e_i.\end{aligned}$$

Server$_i$ computes whether $(P_{1i}-b_i)/a_i$ is $r_i||\mathrm{AR}_i$. Suppose the length of AR_i is 16 bits. Only if Server$_i$ retrieves AR_i' by doing AND operation with 0xFFFF for retrieving $r_i||\mathrm{AR}_i$, will the access right AR_i' be extracted. Finally, Server$_i$ checks whether the equation

$$h_{1i}=H(\mathrm{ID}||\mathrm{AR}_i', d_i)$$

holds. If yes, Server$_i$ accepts the login request and recognizes AR$_i'$ as the access right of the user in Server$_i$. Then Server$_i$ computes the jth session key $H(P_{1i}, P_{2i}, T_s)$, which is the same as the user's.

Continuously, Server$_i$ obtains a time sequence TS″, which is like a timestamp from the terminal, computes $B=H(P_{1i}, \text{TS}'')$ and a new point $P_{si}=(P_{s1i}, P_{s2i}, P_{s3i})=(f_i(B), g_i(B), B)$ as follows:

$$P_{s1i} \Rightarrow f_i(Z)=a_iZ+b_i=a_iB+b_i=a_iH(P_{2i},\text{TS}'')+b_i, \quad (8)$$
$$P_{s2i} \Rightarrow g_i(Z)=c_iZ+e_i=c_iB+e_i=c_iH(P_{2i},\text{TS}'')+e_i, \quad (9)$$
$$P_{s3i} \Rightarrow B=H(P_{2i}\oplus P_{3i}, \text{TS}''), \quad (10)$$

and then returns the public message P_{si} and TS″ to the user.

2. User→Server$_i$ ($1\le i\le m$): Accept or Reject services

The user receives the above message at TS‴ and then checks whether the time interval between TS″ and TS‴ is greater than ΔTS′, the expected legitimate time interval for the transmission delay between the login terminal and the servers Server$_i$. If TS‴−TS″≥ ΔTS′, the user will reject the services from Server$_i$.

Once the time interval of TS‴ is valid, the user checks $P_{s3i}'=H(P_{2i}\oplus P_{3i}, \text{TS}'')$. If this equation does not hold, the user rejects the services from Server$_i$. Hence, mutual authentication is provided between the user and the server.

2.2.4 Changing password phase

In our proposed scheme, users can choose and change their passwords freely offline. When a user wants to change his/her password at server Server$_i$, the steps are as follows:

1. The user should type the old password PW and the new password PW′.

2. The user reconstructs a line L_i according to two points, P_i and the origin (0, 0). Here, L_i: ($f_i(Z)$, $g_i(Z)$, Z), $f_i(Z)=a_iZ+b_i$, $g_i(Z)=c_iZ+e_i$.

3. The user computes $P_{2i}=H(\text{ID}, \text{PW})$ and then obtains P_{3i} and P_{1i} as follows:

$$g_i(Z)=c_iZ+e_i \Rightarrow P_{2i}=g_i(P_{3i})=c_iP_{3i}+e_i \Rightarrow P_{3i}=(P_{2i}-e_i)/c_i.$$
$$f_i(Z)=a_iZ+b_i \Rightarrow P_{1i}=f_i(P_{3i})=a_iP_{3i}+b_i \Rightarrow P_{1i}=a_i(P_{2i}-e_i)/c_i+b_i.$$

4. The user checks whether the equation

$$h_{2i}=H(\text{ID}||(H(\text{ID}, \text{PW})-e_i)/c_i, \text{PW})$$

holds. If the equation does not hold, the system rejects the change of the password request.

5. The user computes the new point $P_i'=(P_{1i}, P_{2i}', P_{3i})$ from the new password PW′ as follows:

$$P_{2i}'=H(\text{ID}, \text{PW}').$$

6. The user constructs a new line $L_{\text{new}\,i}$ according to two points, P_i' and the origin (0, 0). Here, $L_{\text{new}\,i}$: ($f_i'(Z)$, $g_i'(Z)$, Z), $f_i'(X)=a_i'X+b_i'$, $g_i'(X)=c_i'X+e_i'$.

7. The user chooses a new line P_i' at random in a new line $L_{\text{new}\,i}$ and computes the value as follows:

$$h_{2i}'=H(\text{ID}||P_{3i}, \text{PW}').$$

8. Finally, the system updates the P_i and h_{2i} into P_i' and h_{2i}', respectively. The concept of the process is shown in Fig. 3.

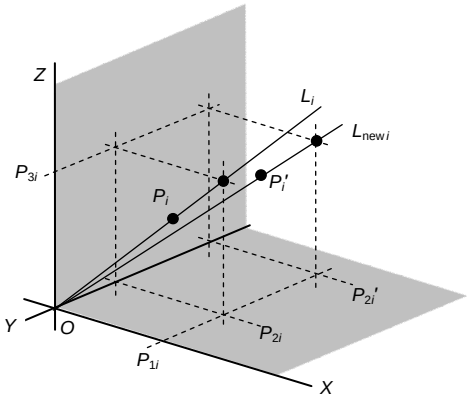

Fig. 3 The concept of changing the password

3 Analysis

In this section, the security of our schemes is analyzed. The strength of our schemes can be demonstrated as follows.

3.1 Security analysis of scheme 1

The plane P: Even if an attacker has a plane P, the attacker cannot retrieve S_i or T_i from Eqs. (1) and (2), because he/she must know both M_i and N_i in the smart card.

S_i: Because of the one-way property of $S_i=H(\text{ID}, d_i)$, no one can derive Server$_i$'s d_i.

A_i: Because of the one-way property of $A_i=H(S_i,$ TS), no one can derive the timestamp TS.

Eavesdropping attack: Assume that all communication is under the adversary's control; that is, the adversary can read or modify the authentic message. The adversary may try to retrieve the secret key d_i from the authentic message (R_i, TS″), but he/she cannot do it successfully because of the one-way hash function.

Stolen verifier attack: Stolen verifier attack means that if an adversary obtains the verification table from the server, he/she can log in the server with any identity. In the proposed scheme, the adversary cannot make the attack successfully, since the server does not store a verification table.

DoS attack: As the remote system has no verification table, it is hard for the attacker to make any modification.

Impersonation attack: When a user loses the portable storage device, the intruder cannot update PW because he/she does not know the owner's PW.

Replay attack: The proposed scheme adopts the timestamp to withstand the replay attack. Assume that an adversary has eavesdropped the past authentic message (ID, B_i, P, TS). Then he/she wants to masquerade as a legitimate user by resending (ID, B_i, P, TS) to log in Server$_i$. When Server$_i$ receives (ID, B_i, P, TS) from the adversary, it will find out that the message has been sent before by checking TS′−TS≥ΔTS. Even if the adversary tries to replace the timestamp TS with the current timestamp, he/she must have $S_i=H$(ID, d_i) to compute B_i and P. He/She will fail since S_i is stored only in the smart card. Hence, the proposed scheme can resist the replay attack.

Server spoofing attack: This attack means that an adversary masquerades as a legitimate server and tries to cheat a legitimate user by responding using a valid response message. The proposed scheme maintains mutual authentication to resist the server spoofing attack. Assume that an adversary tries to plot the server spoofing attack. The adversary must cheat the legal user into sending (ID, B_i, P, TS) to log in the server. When the adversary receives (ID, B_i, P, TS) from the legitimate user, the adversary must generate the response message $R_i=(T_i-aU-c)/b$. Without the legitimate server's secret key d_i, the adversary cannot compute the correct message (R_i, TS″) and then respond with this to the user. Hence, the user will find out that the masqueraded server is illegal.

Security of the session key: A session key $H(T_i,$ TS) is generated from T_i and TS. The session key is different in each session because of the timestamp TS, and the secret parameter $T_i=\mathrm{AR}_i||H(\mathrm{ID}||\mathrm{AR}_i, d_i)||H(\mathrm{ID}||\mathrm{AR}_i,$ PW) can be computed only by the server and the user. Moreover, the session key in each session will not be reused. A new session key will be generated for secret communication between the server and the user when each new session starts. Assume that an adversary has obtained a session key. The adversary will not be able to decrypt the encrypted message in other communication sessions. A known session key cannot be computed without knowing d_i, and the TS is different in each session.

Smart card stolen: Assume that a lost smart card is picked up by an adversary, and he/she may try to retrieve the password from T_i without knowing d_i. The action will fail since PW cannot be retrieved from the one-way hash function.

Off-line password guessing attack: The reasoning is the same as the analysis of smart card stolen. An adversary cannot break the password because it is not only protected with the one-way hash function but also bounded by other parameters.

3.2 Security analysis of scheme 2

P_i: Even if an attacker has P_i to reconstruct a line L_i, the attacker cannot retrieve $P_{\mathrm{v}i}$ from Eqs. (5)–(7) by $B=H(P_{1i},$ TS). Because the attacker must know all of AR_i, Server$_i$'s d_i, and the user's PW, no one can compute $P_{2i}=H$(ID, PW) and then obtain the point $P_i=(P_{1i}, P_{2i}, P_{3i})$ from Eqs. (3) and (4).

h_{1i}, h_{2i}: Due to the one-way property of $h_{1i}=H(\mathrm{ID}||\mathrm{AR}_i, d_i)$ and $h_{2i}=H(\mathrm{ID}||\mathrm{AR}_i,$ PW), no one can derive AR_i, Server$_i$'s d_i, and the user's PW.

$P_{\mathrm{v}i}$: Even if an attacker has P_{vi} to reconstruct a line L_i, the attacker cannot check whether $P_{\mathrm{v}3i}=H(H(\mathrm{ID}, d_i),$ TS). It may fail because TS is not always the same and no one knows Server$_i$'s d_i.

$P_{\mathrm{s}i}$: Even if an attacker has P_{si} to reconstruct a line L_i, the attacker cannot check whether $P_{\mathrm{s}3i}=H(P_{2i}\oplus P_{3i},$ TS″). It may fail because TS″ is not the same every time and no one knows user$_i$'s PW and P_{3i}.

Eavesdropping attack: The reasoning is the same as the security analysis of scheme 1. An adversary cannot retrieve the secret key d_i from the authentic

message (P_{si}, TS″) because of the one-way hash function.

Stolen verifier attack: The reasoning is the same as the security analysis of scheme 1. As the proposed scheme does not store a verification table in the remote system, the attack cannot succeed.

DoS attack: As the remote system has no verification table, it is hard for the attacker to make any user login fail by any modification.

Impersonation attack: When a user loses the portable storage device, the intruder cannot update PW because he/she does not know the owner's PW.

Replay attack: Assume that an adversary has eavesdropped the authentic message (ID, P_{vi}, h_1, TS). Then he/she wants to masquerade as a legitimate user by resending (ID, P_{vi}, h_1, TS) to log in Server$_i$. When Server$_i$ receives (ID, P_{vi}, h_1, TS) from the adversary, it will find out that the message has been sent before by checking TS′−TS≥ΔTS. Even if the adversary tries to replace the timestamp TS with the current timestamp, he/she must have H(ID, d_i) to compute $h_{1i}=H$(ID||AR$_i$, d_i) and $P_{v3i}=H(H$(ID, d_i), TS). The adversary will fail since Server$_i$ is stored only in the smart card. Hence, the proposed scheme can resist the replay attack.

Server spoofing attack: This attack is as described above. The proposed scheme uses mutual authentication to withstand the server spoofing attack. Assume that an adversary tries to plot the server spoofing attack. The adversary must cheat the legal user into sending (ID, P_{vi}, h_1, TS) to log in the server masqueraded by the adversary. When the adversary receives (ID, P_{vi}, h_1, TS) from the legitimate user, he/she must generate the response message (P_{si}, TS″). Without the legitimate server's secret key d_i, the adversary cannot compute the correct message (P_{si}, TS″) according to $P_{1i}=H$(ID, d_i) and then respond with it to the user. Hence, the user will find out that the masqueraded server is illegal.

Security of the session key: A session key $H(P_{1i}$, P_{2i}, TS) is generated from H(ID, d_i), H(ID, PW), and TS. The session key is different in each session because of the timestamp TS. The secret parameters P_{1i} and P_{2i} can be computed only by the server and the user. Moreover, the session key in each session will not be reused. A new session key will be generated for secret communication between the server and the user when each session starts. Assume that an adversary has obtained a session key. The adversary will not be able to decrypt the encrypted message in other communication sessions. A known session key cannot be computed without knowing d_i and PW, and the TS is different in each session.

Smart card stolen: Assume that a lost smart card is picked up by an adversary. The adversary may try to retrieve the password from h_{2i} without knowing AR$_i$. The action will fail since PW cannot be retrieved from the one-way hash function.

Off-line password guessing attack: The reasoning is the same as the analysis of smart card stolen and the analysis of the proposed scheme 1.

3.3 Efficiency

The efficiency of our schemes was examined and compared with other remote authentication schemes (Juang, 2004; Tsaura *et al*., 2005; Tsai, 2008; Liao *et al*., 2009; Liao and Wang, 2009; Rhee *et al*., 2009; Hwang *et al*., 2010). As shown in Table 1, our schemes satisfy all criteria. Table 2 compares the computation cost among various methods in the registration, login, verification, session, and updated password phases. Table 2 also shows that our schemes are the most efficient.

Table 1 Comparison among the different remote schemes

Scheme	R1	R2	R3	R4	R5
Our scheme 1	Y	Y	Y	Y	Y
Our scheme 2	Y	Y	Y	Y	Y
Hwang *et al*., 2010	N	Y	Y	Y	N*
Liao *et al*., 2009	N	Y	Y	Y	Y
Liao and Wang, 2009	Y	Y	Y	Y	Y
Rhee *et al*., 2009	N	Y	Y	Y	N*
Tsai, 2008	Y	Y	N	Y	Y
Tsaura *et al*., 2005	Y	Y	N	N	N*

R1: single registration; R2: no verification table; R3: update password securely and freely; R4: mutual authentication and key management; R5: security. N: no; Y: yes. * No session key

4 Conclusions

In this paper, we present two password authentication schemes on geometrics and propose a specific access right (AR) to authorize the legitimate user on different servers. Compared with previously proposed schemes, our schemes achieve more functionality and efficiency. Furthermore, our proposed schemes allow

Table 2 Comparison of computation cost among the different remote schemes

Scheme	Computation cost				
	Registration	Login	Verification	Session	Updated password
Our scheme 1	2$T(f)$, 4$T(\|\|)$	2$T(f)$	$T(f)$, 2$T(\|\|)$	3$T(f)$	4$T(f)$
Our scheme 2	4$T(f)$, $T(\|\|)$	$T(f)$	3$T(f)$, $T(\|\|)$	3$T(f)$, 2$T(\oplus)$	4$T(f)$, 2$T(\|\|)$
Hwang *et al*., 2010	$T(f)$, 2T(ME)	4$T(f)$, $T(\oplus)$, 2T(ME), $T(r)$	5$T(f)$, $T(\oplus)$, T(ME)	–	–
Liao *et al*., 2009	$T(f)$, 2$T(\oplus)$, T(ME), $T(\|\|)$	$T(f)$, 3$T(\oplus)$, $T(r)$, $T(\|\|)$	$T(f)$, 6$T(\oplus)$, T(ME), $T(r)$, 2$T(\|\|)$	2$T(f)$, $T(\oplus)$, T(ME), 4$T(\|\|)$	3$T(\oplus)$
Liao and Wang, 2009	5(f), 2$T(\oplus)$, 2$T(\|\|)$	6$T(f)$, 3$T(\oplus)$, $T(r)$, 7$T(\|\|)$	6$T(f)$, 3$T(\oplus)$, $T(r)$, 18$T(\|\|)$	2$T(f)$, 8$T(\|\|)$	3$T(f)$, 3$T(\oplus)$, $T(\|\|)$
Rhee *et al*., 2009	2$T(f)$, $T(r)$	2$T(f)$, 2$T(\oplus)$, T(ME), 2$T(r)$	3$T(f)$, 4$T(\oplus)$, 4T(ME)	–	–
Tsai, 2008	2(f), $T(\oplus)$, $T(\|\|)$	$T(f)$, $T(\oplus)$, $T(r)$	6(f), 8$T(\oplus)$, 7$T(\|\|)$, 3$T(r)$	4(f), 5$T(\oplus)$, 2$T(\|\|)$, $T(r)$	–
Tsaura *et al*., 2005	T(ME), $T(r)$	$T(f)$, 3T(ME)	3T(ME)	–	–

$T(f)$: computation cost of a one-way function; $T(\oplus)$: computation cost of an exclusive-OR operation; $T(||)$: computation cost of a string concatenation; $T(r)$: computation cost of random number, nonce and timestamp generation; T(ME): computation cost of modular exponentiation. –: not specified

a user to register with many servers once without having to remember different passwords for each. Therefore, our schemes are suitable for practical implementation.

References

Hwang, M.S., Chong, S.K., Chen, T.Y., 2010. DoS-resistant ID-based password authentication scheme using smart cards. *J. Syst. Softw.*, **83**(1):163-172. [doi:10.1016/j.jss.2009.07.050]

Juang, W.S., 2004. Efficient password authenticated key agreement using smart cards. *Comput. & Secur.*, **23**(2):167-173. [doi:10.1016/j.cose.2003.11.005]

Lamport, L., 1981. Password authentication with insecure communication. *Commun. ACM*, **24**(11):770-772. [doi:10.1145/358790.358797]

Liao, C.H., Chen, H.C., Wang, C.T., 2009. An exquisite mutual authentication scheme with key agreement using smart card. *Informatica*, **33**(2):117-124.

Liao, Y.P., Wang, S.S., 2009. A secure dynamic ID based remote user authentication scheme for multi-server environment. *Comput. Stand. Interf.*, **31**(1):24-29. [doi:10.1016/j.csi.2007.10.007]

Lin, I.C., Hwang, M.S., Li, L.H., 2003. A new remote user authentication scheme for multi-server architecture. *Fut. Gener. Comput. Syst.*, **19**(1):13-22. [doi:10.1016/S0167-739X(02)00093-6]

Rhee, H.S., Kwon, J.O., Lee, D.H., 2009. A remote user authentication scheme without using smart cards. *Comput. Stand. Interf.*, **31**(1):6-13. [doi:10.1016/j.csi.2007.11.017]

Tsai, J.L., 2008. Efficient multi-server authentication scheme based on one-way hash function without verification table. *Comput. & Secur.*, **27**(3-4):115-121. [doi:10.1016/j.cose.2008.04.001]

Tsaur, W.J., Wu, C.C., Lee, W.B., 2005. An enhanced user authentication scheme for multi-server Internet services. *Appl. Math. Comput.*, **170**(1):258-266. [doi:10.1016/j.amc.2004.11.033]

Yoon, E.J., Ryu, E.K., Yoo, K.Y., 2004. Further improvement of an efficient password based remote user authentication scheme using smart cards. *IEEE Trans. Consum. Electron.*, **50**(2):612-614. [doi:10.1109/TCE.2004.1309437]