Security Enhancement for Protecting Password Transmission

Chou-Chen Yang, Ting-Yi Chang, Jian-Wei Li, Min-Shiang Hwang

Graduate Institute of Networking and Communication Engineering
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.
Email: mshwang@cyut.edu.tw
Fax: 886-4-23742337

Department of Information Management
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.

Department of Information and Communication Engineering
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.

September 19, 2012

*This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC91-2213-E-324-003.

†Responsible for correspondence: Prof. Min-Shiang Hwang.
Abstract

In 2002, Hwang and Yeh proposed some improved schemes to mend several security flaws in the Peyravian-Zunic password transmission scheme and password change scheme. However, this article will point out that there still exist some security flaws in the Hwang-Yeh schemes; at the same time, we shall also propose some improved versions of their schemes.

*Keywords:* Authentication, cryptography, password, security.

1 Introduction

The rapid growth of networks in both number and size encourages more and more computers to link together to share various resources and exchange huge amounts of information. With more and more participants joining in, network security has become a more and more important issue. Various resources distributed among the hosts are shared across the network in the form of network services provided by servers. Before users request for services, they should pass the identity authentication by the servers. The password authentication schemes [1, 5, 7, 9, 11, 12, 13] are simple and practical solutions to authenticating users’ identities by servers.

In 2000, Peyravian and Zunic [14] offered a pair of schemes that allowed users to transmit passwords over public networks and arbitrarily change their passwords, and the security was based on the one-way hash function. Unfortunately, Tseng et al. [17] have pointed out that the Peyravian-Zunic schemes are vulnerable to the guessing attack and can be frustrated by server spoofing, and they have used the Diffie-Hellman key agreement scheme [3] to help
mend the above security flaws. On the other hand, Hwang and Yeh [6] have also pointed out that the Peyravian-Zunic schemes suffer from not only the guessing attack and server spoofing but also server data eavesdropping. They have also proposed their improved schemes by using the server’s public key cryptosystem such as RSA [2, 10, 15] and ElGamal [4, 8] cryptosystems to overcome the security flaws in the Peyravian-Zunic schemes.

However, there are still some security flaws in Hwang and Yeh’s password change scheme. Any adversary can intercept the request for changing passwords sent by a legal user and modify it with a wrong password. As a result, the user will not be able to successfully login the server next time. Furthermore, their schemes are vulnerable to server data eavesdropping. If anyone obtains the secret data stored in the server, he/she can easily impersonate a legal user to login the server. In this paper, we will show the security flaws and enhance the security in the Hwang-Yeh schemes.

The organization of the article is as follows. In the next section, we will briefly review the Hwang-Yeh schemes. In Section 3, the security flaws will be shown. In Section 4, we will propose our improvement to enhance the security of their schemes and analyze the security. Finally, we shall conclude this article in Section 5.

2 Review of the Hwang-Yeh Schemes

The main difference between Hwang-Yeh password transmission scheme and password change scheme is that in the latter the client sends a password change request to the server. In the system, the client (or the user) has an identity id and a corresponding password pw. The password is known only to both of the client and the server. The server stores \( v \cdot pw = H(pw) \) instead of pw for each client in the database, where \( H(\cdot) \) is a secure one-way hash function. Here, we only present the password transmission scheme.
Step 1. Client $\rightarrow$ Server: $id, C_{cipher}$

The client encrypts the random number $rc$ and along with $pw$ with the server’s public key $PK_S$ denoted as $C_{cipher} = (rc, pw)PK_S$ and send it with $id$ as a login request to the server.

Step 2. Server $\rightarrow$ Client: $S_{auth\_token}, S_{rs\_digest}$

The server decrypts $C_{cipher}$ to obtain $rc$ and $pw$ by using its private key. Then, the server computes the hash value $H(pw)$ and checks whether $H(pw) = v_{pw}$ holds or not. If it holds, the server chooses a random number $rs$ and computes $S_{auth\_token} = rs \oplus rc$ and $S_{rs\_digest} = H(rs)$, where $\oplus$ is the exclusive-or (XOR) operation. Then, the server sends $\{S_{auth\_token}, S_{rs\_digest}\}$ to the client.

Step 3. Client $\rightarrow$ Server: $id, C_{auth\_token}, C_{digest\_new\_pw}$

The client retrieves $rs$ by computing $S_{auth\_token} \oplus rc$ and then verifies the consistency between the retrieved $rs$ and the received $S_{rs\_digest}$. If the result is positive, the client chooses a new password $new\_pw$ and computes $C_{auth\_token} = H(rc, rs)$ and $C_{digest\_new\_pw} = H(new\_pw) \oplus H(rc + 1, rs)$. Finally, the client sends $\{id, C_{auth\_token}, C_{digest\_new\_pw}\}$ to the server.

Step 4. Server $\rightarrow$ Client: access granted or access denied

The server computes the hash value $H(rc, rs)$ and checks whether $H(rc, rs) = C_{auth\_token}$ holds or not. If it holds, the server can obtain $H(new\_pw)$ by computing $C_{digest\_new\_pw} \oplus H(rc + 1, rs)$ and then store $v_{pw} = H(new\_pw)$ in the database.

Obviously, by employing the public key cryptosystem on the server’s side to protect the transmitted password, Hwang and Yeh have effectively avoided the guessing attack and server spoofing that treated the Peyravian-Zunic schemes.
3 Security Flaws of the Hwang-Yeh Schemes

In this section, we will show Hwang and Yeh’s password change scheme is vulnerable to the modification attack, and both the password transmission scheme and the password change scheme are vulnerable to server data eavesdropping.

Modification attack:
Upon seeing \{id, C_{auth}token, C_{digest}new\_pw\} sent by the client in Step 3, the adversary replaces \(C_{digest}new\_pw\) with a random number \(ra\). After receiving \{id, C_{auth}token, ra\}, the server first computes the hash value \(H(rc, rs)\) and checks whether \(H(rc, rs) = C_{auth}token\) holds or not. Since \(C_{auth}token\) is computed by the client, the equation \(H(rc, rs) = C_{auth}token\) checked by the server will turn out positive. Then, the server computes \(ra \oplus H(rc + 1, rs)\) and stores \(v_{pw} = ra \oplus H(rc + 1, rs)\) in place of \(H(pw)\) in the database.

However, the client is under the impression that it has successfully changed from an old password \(pw\) to a new password \(new\_pw\). Once the client logsins to the server the next time, it sends \{id, C_cipher = (rc, new\_pw)PKS\} to the server in Step 1. In Step 2, the server decrypts the message to obtain \(rc\) and \(new\_pw\) with its private key. Then, the server computes the hash value \(H(new\_pw)\) and check whether \(H(new\_pw) = v_{pw}\) holds or not. However, \(H(new\_pw)\) is not equal to \(v_{pw}\) because \(v_{pw} = ra \oplus H(rc + 1, rs)\). The server will reject the client’s login request.

Server data eavesdropping:
Hwang and Yeh claimed that their schemes could avoid the security flaw where the secret data \(v_{pw} = H(pw)\) gets eavesdropped by the attacker who desires forge the login request to pass the authentication. In practice, people would find the long random string password difficult to use and remember. It would
be much more user-friendly if the password is a meaningful string that people can recognize easily such as a natural language phrase. On the other hand, however, natural language phrases narrow down the possibilities for attackers. When an attacker somehow acquires the secret data \( v_{pw} = H(pw) \) stored in the server, she/he can verify the correctness of the guessed password \( guess_{pw} \) by checking whether \( H(guess_{pw}) = v_{pw} \) holds or not. If the passwords are guessed right, she/he can easily forge the login request to pass the authentication.

4 The Improvement on the Hwang-Yeh Schemes

In this section, we will propose an improved scheme to overcome the previously shown security flaws in the Hwang-Yeh schemes and analyze the security of our improved schemes.

4.1 The Improved Schemes

To overcome the problem of server data eavesdropping in the Hwang-Yeh schemes, the server stores \( v_{pw} = MAC_K(pw) \) instead of \( H(pw) \) for each client in the database, where \( MAC_K(\cdot) \) is a MAC generation function and the keying materials as its key \( K \) [16]. The password change protection scheme has the following steps.

Step 1. Client \( \rightarrow \) Server: \( id, C_{cipher} \)

The client encrypts the random number \( rc \) along with \( pw \) and \( new_{pw} \) with the server public key \( PK_S \) as \( C_{cipher} = (rc, pw, new_{pw})PK_S \) and its id as a login request to the server.

Step 2. Server \( \rightarrow \) Client: \( S_{auth\_token}, S_{rs\_digest} \)

The server decrypts \( C_{cipher} \) to obtain \( rc, pw \) and \( new_{pw} \) with its private key. Then, the server computes \( MAC_K(pw) \) and check whether \( MAC_K(pw) = v_{pw} \) holds or not. If it holds, the server chooses a random
number rs and computes \( S_{\text{auth\_token}} = rs \oplus rc \) and \( S_{\text{rs\_digest}} = H(rs) \). Then, the server sends \( \{S_{\text{auth\_token}}, S_{\text{rs\_digest}}\} \) to the client.

Step 3. Client \( \rightarrow \) Server: \( id, C_{\text{auth\_token}} \)

The client retrieves \( rs \) by computing \( S_{\text{auth\_token}} \oplus rc \) and then verifies the consistency between the retrieved \( rs \) and the received \( S_{\text{rs\_digest}} \).

If the result is positive, the client computes \( C_{\text{auth\_token}} = H(rc, rs) \).

Finally, the client sends \( \{id, C_{\text{auth\_token}}\} \).

Step 4. Server \( \rightarrow \) Client: access granted or access denied

The server first computes the hash value \( H(rc, rs) \) and then checks whether \( H(rc, rs) = C_{\text{auth\_token}} \) holds or not. If it holds, the server can ensure the client is legal. Then, the new \( pw \) decrypted in Step 1 will be stored as \( v_{pw} = (H(pw))K_S \) in the server’s database.

The difference between our password transmission scheme and password change scheme is that in the latter one the client encrypts \( new\_pw \) in Step 1 to the server.

4.2 Security Analysis

In this section, several attacks will be analyzed to demonstrate the security of our improved schemes.

Replay attack:

The attacker intercepts \( \{id, C_{\text{cipher}}\} \) sent by the client in Step 1 and uses it to masquerade as the client to send the login request the next time. However, the random number \( rs \) generated by the server is different every time, and the random number \( rc \) is protected under the server’s public key. Without knowing \( rc \) and \( rs \), the attacker cannot make a correct response \( C_{\text{auth\_token}} \) to the server in Step 3.
Guessing attack:
The random numbers $rc$ and $rs$ separately generated by the client and the server are protected as $C_{\text{cipher}} = (rc, pw)PK_S$ and $S_{\text{auth\_token}} = rs \oplus rc$. No one can reveal $rc$ from $C_{\text{cipher}}$ without knowing the server’s private key. Hence, the attacker cannot verify the correctness of the guessed password by checking $(rc, guess\_pw)PK_S = C_{\text{cipher}}$. For the same reason, in the password change scheme, the guessing attack will also fail.

Server spoofing:
Just like Hwang-Yeh schemes, our improved schemes use the server’s public key to ensure that only the real server can decrypt $C_{\text{cipher}}$. Only a real server can obtain $rc$ from $C_{\text{cipher}}$ and use it to compute the response in Step 2. After verifying the identity of the server, the client sends $C_{\text{auth\_token}}$ to the server to achieve mutual authentication.

Modification attack:
The messages $C_{\text{auth\_token}}$ and $C_{\text{digest\_new\_pw}}$ in the Hwang-Yeh password change scheme are used by the server to authenticate the client and obtain the new password, respectively. Only when the adversary modifies $C_{\text{digest\_new\_pw}}$ to a random number and uses the original $C_{\text{auth\_toke}}$ generated by the client can he/she pass the authentication. In our improved scheme, the new password $new\_pw$ is also encrypted by using the server’s public key in Step 1. Therefore, the adversary is unable to choose a random number to replace $new\_pw$.

Server data eavesdropping:
We know that the servers are always the targets of attack. The attacker may
acquire $v_{pw} = MAC_K(pw)$ stored in the server. However, without knowing the secret key $K$, the attacker cannot forge the login request to pass the authentication because $pw$ is encrypted by the server’s secret key $K_S$ and therefore the correctness of the guessed password cannot be verified by checking $MAC_K(\text{guess } pw) = v_{pw}$.

5 Conclusion

In this article, we have shown that the Hwang-Yeh password change scheme is vulnerable to the modification attack, and both their password transmission scheme and password change scheme are weak against server data eavesdropping. In addition, we have proposed our improved schemes to effectively avoid the security flaws in the Hwang-Yeh schemes.

References


