An Improved Secure Dynamic ID Based Remote User Authentication Scheme for Multi-Server Environment

Cheng-Chi Lee[1,2], Yan-Ming Lai[1] and Chun-Ta Li[3,*]

[1]Department of Library and Information Science,
Fu Jen Catholic University
[2]Department of Photonics and Communication Engineering,
Asia University
[3]Department of Information Management,
Tainan University of Technology
**Corresponding e-mail: th0040@mail.tut.edu.tw*

Abstract

Recently, Lee et al. proposed a secure dynamic ID based remote user authentication scheme for multi-server environment. They claimed their scheme can remedy the weaknesses of prior schemes and is thus more effective. However, we find Lee et al.'s scheme still fails to achieve the anonymity and has the security weakness of a smart card clone. In this article, we shall propose a new scheme to improve Lee et al.'s scheme. Our scheme not only overcomes the weaknesses of Lee et al.'s scheme, but also maintains a high efficiency standard.

Keywords: *Anonymous, authentication, dynamic ID, multi-server, key agreement*

1. Introduction

Today, network services are a popular issue, because increasingly more services are being provided. When we want to access information or ask something from the server, first we need to log on to the server. Therefore, we need an authentication scheme for network servers to authenticate the legitimate users. However, the Internet now is a public environment, anyone can intercept messages of communication between users and servers from the Internet. In order to protect users' secrets, schemes related to maintaining the security of user authentication is becoming more and more important nowadays [1, 6].

In 2004, Das et al. proposed a dynamic ID-based scheme to solve user tracking problems [2]. However, in 2005, Liao et al. pointed out that Das et al.'s scheme cannot protect against guessing attacks and achieve mutual authentication [5]. In 2009, Liao and Wang proposed a new authentication scheme with anonymity for a multi-server [7]. Their scheme only uses one way hash functions to improve efficiency. But Hsiang and Shih pointed out Liao and Wang's scheme cannot withstand insider attacks, masquerade attacks, server spoofing attacks, and registration center spoofing attacks in the same year [3]. To overcome the weaknesses of Liao and Wang's scheme, Hsiang and Shih proposed their scheme. Nevertheless, in 2011, Lee et al. found Hsiang and Shih's scheme still could not overcome masquerade attacks, and server spoofing attacks [4]. In addition, they proposed an improved scheme to solve the weaknesses of Hsiang and Shih's scheme. Lee et al.'s scheme let registration center share a secret value with a legal server to compute the legal user's secret value. In their paper, they claimed their scheme is practicable in the future. However, we find Lee et al.'s still scheme fails to achieve the anonymity and has security weakness of a smart card clone. To solve these problems of anonymity authentication, we propose an improved scheme in this paper, which is more effective and secure than the others.

2. Review of Lee et al.'s Scheme

In this section, we will review the Lee et al.'s scheme. Three roles participate in this scheme: the registration center (RC), the remote server (S), and the User (U). The RC chooses a master key x and secret value y to compute $h(x||y)$ and $h(y)$. After that, RC shares h(y) to registered server (S). User (U) registers at RC and login S, which registers at the same RC, to access resources. The scheme is divided into four phases: registration phase, login phase, verification phase, and password change phase. To shorten the length of this paper, we omit the review. Please refer to [4].

3. Cryptanalysis of Lee et al.'s Scheme

In this section, we will show the weaknesses of Lee et al.'s scheme that fails to achieve the necessary anonymity, and the situation is a high-risk when the information of the smart card is disclosed. In addition, their mechanism for changing a password is not really friendly. We will assume that the communication between U and S is eavesdropped on by an adversary Z, and explain how the adversary Z can attack Lee et al.'s scheme.

3.1. Fails to Achieve the Anonymity

When a legal user U_i wants to access roaming services from server S_j, he/she must send $\{CID_i, P_{ij}, Q_i, N_i\}$ to S_i. If the adversary Z is another legal user who registers at the same RC with U_i, Z can derive a unique value of U_i and trace U_i by the value at this moment. Because Z registered at the same RC with U_i, he/she also has the value $h(y)$. After intercepting $\{CID_i, P_{ij}, Q_i, N_i\}$, Z can compute $T_i=P_{ij} \oplus h(h(y)||N_i||SID_j)$, $A_i=h(T_i||h(y)||N_i)$, and $h(b \oplus PW_i)=CID_i \oplus h(T_i||A_i||N_i)$.

Because the unique value $h(b \oplus PW_i)$ of U_i doesn't change frequently, Z can trace U_i by this unique value. For this reason, we can know that Lee et al.'s scheme fails to achieve the really anonymity.

3.2. Smart Card Clone

In Lee et al.'s scheme, the security is based on the secure value B_i. In both user authentication or session key establishment, B_i always plays an important role. However, this important value is not stored securely in the smart card. If the adversary Z is another legal user who registers at the same RC with U_i and if he/she temporarily possesses the smart card of U_i, Z can easily obtain B_i and uses it to attack the system, such as by re-establishing the session key SK, which is established by B_i, N_i, N_j, A_i, and SID_j to decrypt those cipher-text. Because $\{N_i, N_j, SID_j\}$ can be intercepted from the public network environment, and as A_i can be derived as section 3.1 mentioned, Z can implement a man-in-the-middle attack and masquerade an attack as follows.

3.3. Man-in-the-middle Attack

If Z obtains B_i, Z can execute a man-in-the-middle attack with processes as follows.

Step ZV1: Z intercepts $\{CID_i, P_{ij}, Q_i, N_i\}$ and computes $\{T_i, A_i, h(b \oplus PW_i)\}$ as $T_i=P_{ij} \oplus h(h(y)||N_i||SID_j)$, $A_i=h(T_i||h(y)||N_i)$, and $h(b \oplus PW_i)=CID_i \oplus h(T_i||A_i||N_i)$.

Step ZV2: After Z obtains $\{T_i, A_i, h(b \oplus PW_i)\}$, Z chooses a random number Nz_i and computes $\{CIDz_i, Pz_{ij}, Qz_i, Nz_i\}$ as $Az_i=h(T_i||h(y)||Nz_i)$, $CIDz_i=h(b \oplus PW_i) \oplus h(T_i||Az_i||Nz_i)$, $Pz_{ij}=T_i \oplus h(h(y)||Nz_i||SID_j)$, and $Qz_i=h(B_i||Az_i||Nz_i)$. And then, Z sends $\{CIDz_i, Pz_{ij}, Qz_i, Nz_i\}$ to S_j.

Step ZV3: After S_j returns $\{M'_{ij}, N_j\}$, Z also intercepts the data and computes both $M''z_j=h(B_i||N_j||Az_i||SID_j)$ and the session key $SKz_j=h(B_i||Nz_i||N_j||Az_i|| SID_j)$.

Step ZV4: Z randomly chooses a number Nz_j and intercepts the data and computes both $M'_{zj} =h(B_i||Nz_j||A_i||SID_j)$ and the session key $SKz_i=h(B_i||N_i|| Nz_j||A_i||SID_j)$. Finally, Z sends $M''z_j$ to S_j and sends $M'z_j$ to U_i individually.

Because Z has the secret value B_i, he/she can establish session $\{SKz_j, SKz_i\}$ with S_j and U_i individually, and then Z can eavesdrop or alter the communication between U_i and S_j.

4. Our Scheme

In this section, we will show a new scheme, which satisfies Liao et al.'s ten requirements, to avoid those weaknesses of Lee et al.'s scheme and improve the computation efficiency. Our scheme also includes three roles as user (U_i), registration center (RC), and server (S_j). There are five phases in this scheme: registration phase, login phase, verification phase, update session key phase, and password change phase. RC chooses a secret value x and computes $h(x)$, and then shares $h(x)$ with each legal servers via a secure channel. The notations of our scheme are described as follows. U_i denotes the i-th user. S_j denotes the j-th server. RC denotes the registration center. PW_i denotes the password of U_i. ID_i denotes the identity of $U_{i.}$ SID_j denotes the identity of S_j. CID_i denotes the dynamic ID of U_i. $h(.)$ denotes a one-way hash function. N_X denotes number used only once (Nonce) generated by an entity X. $||$ denotes concatenation operation. $\oplus$ denotes XOR operation. SK denotes session key between entity U_i and S_j. x denotes the secret value of RC. b, b_{new} denotes random number generated by U_i.

4.1. Registration Phase

When a new user U_i wants to access the services S_j, he/she needs to register at the same RC that S_j registered. The communication between U_i and RC is via a secure channel.

Step 1: When the registration starts, U_i generates a random number b, and chooses his/her password PW_i. After that, U_i computes CID as $CID= h(ID_i \oplus PW_i) \oplus b$. And then, U_i sends CID to RC.

Step 2: After RC receives CID from U_i, RC computes B_i as $B_i=h(CID||h(x))$, And then, RC returns $\{B_i, h(.)\}$ to U_i.

Step 3: When U_i receives $\{B_i, h(.)\}$, U_i computes BPW as $BPW=B_i \oplus h(PW_i)$. Finally, U_i stores $\{BPW, b, h(.)\}$ in smart card.

4.2. Login and Verification Phase

After the registration phase, U_i can access the resources of S_j. For security, U_i and S_j have to agree a session key to communicate each other, but U_i and S_j must mutual authenticate before that.

Step 1: U_i inputs his/her identity ID_i and password PW_i, and then the device computes $\{CID, B_i\}$ as $CID = h(ID_i \oplus PW_i) \oplus b$ and $B_i=BPW \oplus h(PW_i)$. After that, U_i generates two random numbers, b_{new} and N_i. Next, U_i computes $\{V_i, CID_{new}, Q_i\}$ as $CID_{new}= h(ID_i \oplus PW_i) \oplus b_{new}$, $V_i = CID_{new} \oplus h(B_i||N_i)$, and $Q_i = h(CID_{new}||B_i||N_i)$. Finally, U_i submits $\{CID, V_i, Q_i, N_i\}$ to S_j.

Step 2: When S_j receives $\{CID, V_i, Q_i, N_i\}$ from U_i, S_j computes $\{B_i, CID_{new}\}$ and verifies Q_i.

$$B_i=h(CID||h(x))$$

$$CID_{new}= V_i \oplus h(B_i||N_i)$$

$$Q_i=?h(CID_{new}||B_i||N_i)$$

If Q_i equals $h(CID_{new}||B_i||N_i)$, S_j can authenticate U_i, and the authenticating phase of U_i is completed. After the authenticating phase of U_i, S_j generates a random number N_j, and computes $\{B_{new}, V_j, Q_j\}$ as $B_{new}=h(CID_{new}||h(x))$, $V_j= B_{new} \oplus h(B_i||N_j)$, and $Q_j= h(CID|| B_{new}||N_j)$. Finally, S_j returns $\{V_j, Q_j, N_j\}$ to U_i.

Step 3: When U_i obtains $\{V_j, Q_j, N_j\}$ from S_j, U_i computes $B_{new}=V_j \oplus h(B_i||N_j)$ and verifies Q_j as $Q_j=?h(CID||B_{new}||N_j)$. If Q_j equals $h(CID||B_{new}||N_j)$, U_i can authenticate the S_j and the authenticating phase of S_j is completed. Next, U_i computes BPW_{new} as $BPW_{new} =B_{new} \oplus h(PW_i)$. And then, U_i stores $\{BPW_{new}, b_{new}\}$ for the next login. After that, U_i computes SK and Q_{ij} as $SK=h(N_i||N_j||B_i)$ and $Q_{ij}=h(N_i||B_i||N_j||B_{new})$. SK is the session key of U_i which is used for the communication with S_j this time. And then, U_i submits Q_{ij} to S_j for double check.

Step 4: When S_j receives Q_{ij}, S_j verifies Q_{ij} as $Q_{ij}=?h(N_i||B_i||N_j||B_{new})$. If they are equal, S_j computes the session $SK=h(N_i||N_j||B_i)$. After that, the login and verification phase is completed.

4.3. Update Session Key Phase

In order to improve the overall security, U_i and S_j can update their session key SK when they have held the communication for a long time in our scheme. In this phase, $\{N_i, N_j\}$ denotes the existing random values, and $\{N^*_i, N^*_j\}$ denotes two random values. After this phase completed, U_i and S_j will replace $\{N_i, N_j\}$ with $\{N^*_i, N^*_j\}$.

Step 1: U_i chooses a random number N^*_i. After that, U_i computes $\{V^*_i, Q^*_i\}$ as $V^*_i= N^*_i \oplus h(B_i \oplus h(N_i||N_j))$ and $Q^*_i= h(N^*_i \oplus B_i)$. Last, U_i submits V^*_i and Q^*_i to S_j.

Step 2: When S_j receives V^*_i and Q^*_i from U_i, S_j computes N^*_i and verifies Q^*_i as $N^*_i=V^*_i \oplus h(B_i \oplus h(N_i||N_j))$ and $Q^*_i =? h(N^*_i \oplus B_i)$. If Q^*_i is equal to $h(N^*_i \oplus B_i)$, S_j generates a random number N^*_j. Next, S_j computes $\{V^*_j, Q^*_j\}$ as $V^*_j= N^*_j \oplus h(B_i \oplus h(N_j||N_i))$ and $Q^*_j= h(N^*_j \oplus B_i)$. Finally, S_j returns $\{V^*_j, Q^*_j\}$ to U_i.

Step 3: When U_i obtains V^*_j and Q^*_j from S_j, U_i computes N^*_j and verifies Q^*_j as $N^*_j = V^*_j \oplus h(B_i \oplus h(N_j||N_i))$ and $Q^*_j =?h(N^*_j \oplus B_i)$. If Q^*_j and $h(N^*_j \oplus B_i)$ are equal, U_i updates the session key SK to SK^* as $SK^* = h(N^*_i||N^*_j||B_i)$. Next, U_i computes Q^*_{ij} as $Q^*_{ij} = h(N^*_i \oplus N^*_j \oplus B_i)$. Finally, U_i sends Q^*_{ij} to S_j.

Step 4: When S_j obtains Q^*_{ij} from U_i, S_j verifies Q^*_{ij} first as $Q^*_{ij} =?h(N^*_i \oplus N^*_j \oplus B_i)$. If Q^*_{ij} and $h(N^*_i \oplus N^*_j \oplus B_i)$ are equal, S_j also updates the session key SK to SK^* as $SK^* = h(N^*_i||N^*_j||B_i)$. And then, the session key update is finished.

4.4. Password Change Phase

A user-friendly system should provide a mechanism with respect to the password chosen so that a password may be changed. In our scheme, the user can change his/her password freely at will by the owner. If a user wants to change his/her password, he/she just needs to input the existing password PW_i and chooses a new password PW_{new}. Our scheme will implement the password-changed phase, which is similar to both the authentication and establishment of session key phase.

Step 1: U_i inputs his/her password PW_i, and chooses a new password PW_{new} then the device computes $\{CID, B_i\}$ as $CID = h(ID_i \oplus PW_i) \oplus b$ and $B_i=BPW \oplus h(PW_i)$. After that, the device generates two random numbers, b_{new} and N_i. Next, U_i computes $\{V_i, CID_{new}, Q_i\}$ as $CID_{new}= h(ID_i \oplus PW_{new}) \oplus b_{new}$, $V_i =CID_{new} \oplus h(B_i||N_i)$, and $Q_i = h(CID_{new}||B_i||N_i)$. Finally, U_i submits $\{CID, V_i, Q_i, N_i\}$ to S_j.

Step 2: When S_j receives $\{CID, V_i, Q_i, N_i\}$ from U_i, S_j computes $\{B_i, CID_{new}\}$ and verifies Q_i as $B_i=h(CID||h(x))$, $CID_{new}= V_i \oplus h(B_i||N_i)$, *and* $Q_i=?h(CID_{new}||B_i||N_i)$. If Q_i equals $h(CID_{new}||B_i||N_i)$, S_j can authenticate U_i, and the authenticating phase of U_i is completed. After the authenticating phase of U_i, S_j generates a random number N_j, and computes $\{B_{new}, V_j, Q_j\}$ as $B_{new}=h(CID_{new}||h(x))$, $V_j= B_{new} \oplus h(B_i||N_j)$, and $Q_j= h(CID|| B_{new}||N_j)$. Finally, S_j returns $\{V_j, Q_j, N_j\}$ to U_i.

Step 3: When U_i obtains $\{V_j, Q_j, N_j\}$ from S_j, U_i computes B_{new} and verifies Q_j as $B_{new}=V_j \oplus h(B_i||N_j)$ and $Q_j=?h(CID||B_{new}||N_j)$. If Q_j equals $h(CID||B_{new}||N_j)$, U_i can authenticate the S_j and the authenticating phase of S_j is completed. Next, U_i computes BPW_{new} as $BPW_{new} =B_{new} \oplus h(PW_{new})$. And then, U_i stores $\{BPW_{new}, b_{new}\}$ for the next login. After that, the password changed phase is completed.

5. Analysis of our Scheme

We analyze the security of our scheme in this section. Our scheme not only withstands the various leaks of previous studies but also has more efficacies.

First, we assume an adversary Z in following scenarios, and explain some hurdles that the adversary Z will come across if the adversary Z wants to break the security of our scheme. ***Anonymity***: If the adversary Z attempts to use the information, which can be intercepted from the public communicating environment, to trace the user U_i, he/she needs to discern the relation between each communication. However, we use a random number b, which is changed for each authorization, to compose the values as $\{CID, V_i, Q_i\}$, and b has never been disclosed. Namely the values are different for each communication, and there is no any relation between each change. In addition, $\{CID, V_i\}$ are not directly composed with the parameters from RC. Therefore, those values have no relation with other legal users, and Z can't derive constant values by the public information even if he/she is also a legal user. Moreover, as there are no verification tables stored in servers, Z unable to obtain any information about the user even if Z invades any server in this system successfully. For the above reasons, we claim our scheme achieves the necessary anonymity.

Perfect forward secrecy and backward secrecy: In this scheme, the session key is established by $\{N_i, N_j, B_i\}$, where $\{N_i, N_j\}$ are two random numbers generated each time and B_i is a secure value, which is also changed for each authentication phase. Although Z can intercept N_i and N_j when the authentication phase proceeds, but he/she cannot obtain B_i which is always computed by one-way hash function. In addition, the new $\{N^*_i, N^*_j\}$ are always encrypted by $h(B_i \oplus h(N_i||N_j))$ and $h(B_i \oplus h(N_j||N_i))$ respectively except for first time. Therefore, Z cannot use one of the session keys to derive other key. Hence, we claim our scheme achieves perfect forward secrecy and backward secrecy.

Masquerade attack: We suppose the adversary Z intercepts the legal user U_i's information and he/she attempts to masquerade the legal user U_i to communicate with the legal server S_j or masquerade the legal server S_j to communicate with the legal user U_i in our scheme. However, he/she will come across some difficulties in following scenarios.

Z intercepts {CID, V_i, Q_i, N_i} when U_i submit them to S_j and replays them for disguising the user U_i after U_i is offline: This action will pass the first authentication of S_j but will not be approved in the second authentication. As a result, Z cannot obtain {B_i, B_{new}} to compute Q_{ij}. For this reason, Z cannot disguise the user U_i and communicate with S_j.

Z intercepts {V_j, Q_j, N_j} when S_j returns them to U_i and replays them for disguising the S_j to communicate with U_i: This action cannot pass the authentication of U_i. When Z replays {V_i, Q_i, N_i}, U_i verifies that Q_j is equal to $h(CID||V_j \oplus h(B_i||N_j)||N_j)$, and detects they are not equal, because the secure value B_i of U_i will be changed after the authentication phase between U_i and S_j each time. Hence, Z cannot disguise S_j and communicate with the user U_i.

Z intercepts {CID, V_i, Q_i, N_i} and {Q_{ij}} when U_i submit them to S_j and replays them for disguising the user U_i after U_i is offline: The action cannot pass the second authentication of S_j. As a result, S_j verifies Q_{ij} when it receives Q_{ij} by N_j which is a random and temporary number generated by itself. Because the S_j doesn't store N_j after each communication, it cannot compute $h(N_i||B_i||N_j||B_{new})$ again. Therefore, Z cannot disguise the user U_i and communicate with S_j.
Because of the above reasons, we claim our scheme can withstand the masquerade attack and replay attack.

Device is lost: If a user U_i lost his/her device and the adversary Z picks it up, Z can obtain the information which is stored in the device as {BPW, b}. But Z still cannot use that information he/she obtains to disguise U_i and communicate with the server S_j. Because Z cannot compute $B_i = BPW \oplus h(PW_i)$ and cannot compute {V_i, Q_i}, hence, Z cannot pass the authentication of S_j and disguise U_i. For the reason, we claim the information of a user is secure even if the user lost his/her device in our scheme.

Password be disclosed: Suppose a user U_i discloses his/her password carelessly and the adversary Z obtains it, Z also cannot use the password to break the security of this scheme. In our scheme, the parameters, which are transmitted between U_i and S_j aren't constructed from password of U_i. Because of that, Z cannot use the password to disguise the user U_i or decrypt the information he/she intercepts. Our scheme not only withstands the attack from the adversary Z when user's password lost, but also provides a friendly way to users about password changed that is described in section 4.4 of this article. For the above reasons, we claim our scheme is secure even if a user lost his/her password.

In addition, we provide a mechanism for updating the session key SK between U_i and S_j when the communication has been hold a long time. This mechanism can strengthen the security of our scheme. Because it can allow the cipher-text encrypted by different SK in the same communication, the adversary Z cannot decrypt all of them even if Z obtains one of SK. For the above reasons, we claim the security of our scheme is better than previous schemes.

6. Conclusion

In this article, we propose an improved scheme of anonymous authentication to improve the security and efficiency of a multi-server environment. The scheme not only overcomes the secure leaks of Lee et al.'s scheme but also maintains a high level of computational efficiency. Our analysis shows that our scheme can achieve the real anonymity and better efficiency by means of easy computations as one-way hash functions and XOR operations. Moreover, our scheme has other advantages, such as a changeable password, no verification table, and mutual authentication, etc. From the above advantages, we can claim that our scheme is very suitable for a multi-server environment.

Acknowledgements

This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC 100-2221-E-030-015 and NSC 100-2221-E-165-002.

References

[1] C. Chang and J. S. Lee, Proceeding of *International Conference on Cyberworlds*, **(2004)**, Nov. 18-20, Japan.
[2] M. L. Das, A. Saxena, and V. P. Gulati, *IEEE Transactions on Consumer Electronics 50(2)*, pp. 629 - 631, **(2004)**.
[3] Han-Cheng Hsiang and Wei-Kuan Shih, *Computer Standards & Interfaces 31(6)*, pp. 1118-1123, **(2009)**.
[4] Cheng-Chi Lee, Tsung-Hung Lin, and Rui-Xiang Chang, *Expert Systems with Applications 38(11)*, pp. 13863-13870, **(2011)**.
[5] I-En Liao, Cheng-Chi Lee, and Min-Shiang Hwang, Proceeding of *International Conference on Next Generation Web Services Practices* (NWeSP 2005), **(2005)**, August 22-26, Seoul, Korea.
[6] I. E. Liao, C. C. Lee, and M. S. Hwang, *Journal of Computer and System Sciences 72(4)*, pp. 727-740, **(2006)**.
[7] Yi-Pin Liao and Shuenn-Shyang Wang, *Computer Standards & Interfaces 31(1)*, pp. 24-29, **(2009)**.