Secure Password-based Remote User Authentication Scheme with Non-tamper Resistant Smart Cards

Ding Wang[1,2⋆], Chun-guang Ma[1,⋆⋆], and Peng Wu[1]

[1] Harbin Engineering University, Harbin City 150001, China
[2] Automobile Management Institute of PLA, Bengbu City 233011, China
wangdingg@mail.nankai.edu.cn, chunguangma@hrbeu.edu.cn

Abstract. In DBSec'11, Li et al. showed that Kim and Chung's password-based remote user authentication scheme is vulnerable to various attacks if the smart card is non-tamper resistant. Consequently, an improved version was proposed and claimed that it is secure against smart card security breach attacks. In this paper, however, we will show that Li et al.'s scheme still cannot withstand offline password guessing attack under the non-tamper resistance assumption of the smart card. In addition, their scheme is also prone to denial of service attack and fails to provide user anonymity and forward secrecy. Therefore, a robust scheme with a brief analysis is presented to overcome the identified drawbacks.

Keywords: Cryptanalysis; Network security; Authentication protocol; Smart card; Non-tamper resistant; User anonymity

1 Introduction

Password-based authentication is widely used for systems that control remote access to computer networks. In order to address some of the security and management problems that occur in traditional password authentication protocols, research in recent decades has focused on smart card based password authentication. Since Chang and Wu [1] introduced the first remote user authentication scheme using smart cards in 1993, there have been many smart card based authentication schemes proposed. In most of the previous authentication schemes, the smart card is assumed to be tamper-resistant, i.e., the secret information stored in the smart card cannot be revealed. However, recent research results have shown that the secret data stored in the smart card could be extracted by some means, such as monitoring the power consumption [2, 3] or analyzing the leaked information [4]. Therefore, such schemes based on the tamper resistance assumption of the smart card are vulnerable to some types of attacks, such as impersonation attacks, offline password guessing attacks, etc., once an adversary

⋆ This is the extended abstract and a full version [7] of this paper is available at http://machunguang.hrbeu.edu.cn/Research/

⋆⋆ Corresponding author.

has obtained the secret information stored in a user's smart card and/or just some intermediate computational results in the smart card.

In DBSec'11, Li et al. [5] identified that Kim and Chung's scheme [6] cannot withstand various attacks and further proposed an enhanced remote authentication scheme. They claimed their scheme is secure and can overcome all the identified security flaws of Kim and Chung's scheme even if the smart card is non-tamper resistant. In this work, however, we will demonstrate that Li et al.'s scheme cannot withstand denial of service attack, and it is still vulnerable to offline password guessing attack under their assumption. In addition, their scheme does not provide forward secrecy and user anonymity. To conquer the identified weaknesses, a robust authentication scheme based on the secure one-way hash function and the well-known discrete logarithm problem is presented.

2 Review of Li et al.'s scheme

In this section, we briefly illustrate the remote user authentication scheme proposed by Li et al. [5] in DBSec 2011. Their scheme consists of four phases: registration, login, verification and password update. For ease of presentation, we employ some intuitive abbreviations and notations listed in Table 1.

Table 1. Notations

Symbol	Description	Symbol	Description
U_i	i^{th} user	x	the secret key of remote server S
S	remote server	$\|\|$	the string concatenation operation
ID_i	identity of user U_i	$h(\cdot)$	collision free one-way hash function
P_i	password of user U_i	$\rightarrow$	a common channel
$\oplus$	the bitwise XOR operation	$\Rightarrow$	a secure channel

2.1 Registration phase

The registration phase involves the following operations:

1) User U_i chooses his/her identity ID_i, password P_i, and then generates a random number RN_1.

2) $U_i \Rightarrow S : \{ID_i, h(h(P_i \oplus RN_1))\}$.

3) On receiving the registration message from U_i, the server S creates an entry $\{ID_i, N, h(h(P_i \oplus RN_1))\}$ in the verification table,where $N = 0$ if it is U_i's initial registration, otherwise S set $N = N + 1$. Then, server S computes $C_1 = h(ID_i \parallel x \parallel N) \oplus h(h(P_i \oplus RN_1))$

4) $S \Rightarrow U_i$: A smart card containing security parameters $\{ID_i, C_1, h(\cdot)\}$.

5) Upon receiving the smart card,user U_i stores RN_1 into his/her smart card.

2.2 Login phase

When U_i wants to login to S, the following operations will be performed:

1) U_i inserts his/her smart card into the card reader, and inputs ID_i,P_i and a random number RN_2.

2) The smart card generates a random number RC and then computes $C_2 = h(P_i \oplus RN_1)$,$C_3 = C_1 \oplus h(C_2)$,$C_4 = C_3 \oplus C_2$,$C_5 = h(h(P_i \oplus RN_2))$ and $C_6 = E_{KU_i}(C_5, RC)$,where $K_{U_i} = h(C_2 \parallel C_3)$.

3) $U_i \rightarrow S : \{ID_i, C_4, C_6\}$.

2.3 Verification phase

After receiving the login request from U_i, S performs the following operations:

1) The server S first checks the validity of identity ID_i and then computes $C_7 = h(ID_i \parallel x \parallel N)$, $C_8 = C_4 \oplus C_7$, $C_9 = h(C_8)$, and compares C_9 with the third field of the entry corresponding to ID_i in its verification table. If it equals, S successfully authenticates U_i and computes symmetric key $K'_{U_i} = h(C_8 \parallel C_7)$, and obtains ($C_5$,$RC$) by decrypting C_6. Then, S replaces the third field $h(h(P_i \oplus RN_i))$ of the entry corresponding to ID_i with $C_5 = h((P_i \oplus RN_2))$, generates a random RS and computes $K_5 = h(C_7 \parallel C_8)$.

2) $S \rightarrow U_i : \{E_{K_5}(RC, RS, C_5)\}$.

3) On receiving the response from server S, the smart card computes the symmetric key $K'_s = h(C_3 \parallel C_2)$ and obtains (RC', C'_5) by decrypting the received message using K_s. Then, the smart card checks whether (RC', C'_5) equals to (RC, C_5) generated in the login phase. This equivalency authenticates the legitimacy of the server S, and smart card replaces original RN_1 and C_1 with new RN_2 and $C_3 \oplus C_5$, respectively.

4) $U_i \rightarrow S : \{h(RS)\}$

5) On receiving $h(RS)'$, the serve S compares the computed $h(RS)$ with the received value of $h(RS)'$. If they are not equal, the connection is terminated.

6) The user U_i and the server S agree on the session key $SK = h(RC \oplus RS)$ for securing future data communications.

2.4 Password change phase

The password change phase is provided to allow users to change their passwords freely. Since the password change phase has little to do with our discussion, we omit it here and detailed information is referred to Ref. [5].

3 Cryptanalysis of Li et al.'s scheme

In this section we will show that Li et al.'s scheme is vulnerable to offline password guessing attack and denial of service attack. In addition, their scheme fails to preserve user anonymity and forward secrecy. Although tamper resistant smart card is widely assumed in the literature, such an assumption is difficult in practice. Many researchers have shown that the secret information stored in a smartcard can be breached [2–4]. Be aware of this threat, Li et al. intentionally based their scheme on the assumption of non-tamper resistance of the smart card. However, Li et al.'s scheme fails to serve its purposes.

3.1 Offline password guessing attack

Let us consider the following scenarios. In case a legitimate user U_i's smart card is stolen by an adversary $\mathcal{A}$ just before U_i's jth login, and the stored secret values such as C_1 and RN_j can be revealed. Then, $\mathcal{A}$ returns the smart card to U_i and eavesdrops on the insecure channel. Because U_i's identity is transmitted in plaintext within the login request, it is not difficult for $\mathcal{A}$ to identify the login request message from U_i. Once the jth login request message $\{ID_i, C_i^j = h(ID_i \parallel x \parallel N) \oplus h(P_i \oplus RN_i), C_6^j\}$ is intercepted by $\mathcal{A}$, an offline password guessing attack can be launched in the following steps:

Step 1. Guesses the value of P_i to be P_i^* from the password dictionary.
Step 2. Computes $T = h(h(P_i^* \oplus RN_j)) \oplus h(P_i^* \oplus RN_j)$, as RN_j is known. .
Step 3. Computes $T' = C_1 \oplus C_4^j$, as C_1 has been extracted and C_4^j has been intercepted, where $C_1 = h(ID_i \parallel x \parallel N) \oplus h(h(P_i \oplus RN_j)), C_4^j = h(ID_i \parallel x \parallel N) \oplus h(P_i \oplus RN_j)$.
Step 4. Verifies the correctness of P_i^* by checking if T is equal to T'.
Step 5. Repeats Steps 1, 2, 3, and 4 until the correct value of P_i is found.

After guessing the correct value of P_i, the adversary $\mathcal{A}$ can compute $C_3^j = C_1 \oplus h(h(P_i \oplus RN_j)), C_2^j = h(P_i \oplus RN_j)$ and $K_{U_i}^j = h(C_2^j \parallel C_3^j)$. Then the adversary can obtain RC_j by decrypting C_6^j using $K_{U_i}^j$, and gets RS_j in a similar way. Hence the malicious user can successfully compute the session key $SK_j = h(RC_j \oplus RS_j)$ and renders the jth session between U_i and S completely insecure.

3.2 Denial of service attack

A denial of service attack is an offensive action whereby the adversary could use some methods to work upon the server so that the login requests issued by the legitimate user will be denied by the server. In Li et at.'s scheme, an adversary can easily launch a denial of service attack in the following steps:

Step 1. Eavesdrops over the channel, intercepts a login request $\{ID_i, C_4^j, C_6^j\}$ from U_i and blocks it, supposing it is U_i's jth login.
Step 2. Replaces C_6^j with an equal-sized random number R, while ID_i and C_4^j are left unchanged.
Step 3. Sends $\{ID_i, C_4^j, R\}$ instead of $\{ID_i, C_4^j, C_6^j\}$ to the remote server S.

After receiving this modified message, S will perform Step $V1$ and $V2$ of the verification phase without observing any abnormality, as a result, the verifier corresponding to ID_i in the verification table will be updated and the response $E_{K_S}(RC_j^*, RS_j, C_5^{j*})$ will be sent to U_i. On receiving the response from S, U_i decrypts $E_{K_S}(RC_j^*, RS_j, C_5^{j*})$ and will find (RC_j^*, C_5^{j*}) unequal to (RC, C_5) , thus the session will be terminated. Thereafter, U_i's succeeding login requests will be denied unless he/she re-registers to S again. That is, the adversary can easily lock the account of any legitimate user without using any cryptographic techniques. Thus, Li et al.'s protocol is vulnerable to denial of service attack.

3.3 Failure to achieve forward secrecy

Let us consider the following scenarios. Supposing the server S's long time private key x is leaked out by accident or intentionally stolen by an adversary $\mathcal{A}$. Once the value of x is obtained, with previously intercepted C_4^j, C_6^j and $E_{K_S}(RC, RS, C_5)$ transmitted in the legitimate user U_i's jth authentication process, $\mathcal{A}$ can compute the session key of S and U_i's jth encrypted communication through the following method:

Step 1. Assumes $N = 0$.
Step 2. Computes $C_7^* = h(ID_i \parallel x \parallel N)$ and $C_8^* = C_7^* \oplus C_4^j$, where ID_i is previously obtained by eavesdropping on the insecure channel.
Step 3. Computes $K_{U_i}^* = h(C_8^* \parallel C_7^*)$ and $K_S^* = h(C_7^* \parallel C_8^*)$.
Step 4. Decrypts C_6^j with $K_{U_i}^*$ to obtain RC_i^*.
Step 5. Decrypts $E_{K_S}(RC, RS, C_5)$ with K_S^* to obtain RC_i^{**}.
Step 6. Verifies the correctness of N by checking if RC_i^* is equal to RC_i^{**}. If they are unequal, sets $N = N + 1$ and goes back to Setp 2.
Step 7. Decrypts $E_{K_S}(RC, RS, C_5)$ to obtain RS_i using K_S^*.
Step 8. Computes $SK_i = h(RC_i \oplus RS_i)$.

Note that the value of N should not be very big, since the re-registration phase is not performed frequently in practice, and thus the above procedure can be completed in polynomial time, which results in the breach of forward secrecy.

3.4 Failure to preserve user anonymity

In Li et al.'s scheme, user's identity ID is static and in plaintext form in all the transaction sessions, an adversary can easily obtain the plaintext identity of this communicating client once the login messages were eavesdropped. Hence, different login request messages belonging to the same user can be traced out and may be interlinked to derive some secret information related to the user [8]. Consequently, user anonymity is not preserved in their scheme.

4 Our proposed scheme

According to our analysis, three principles for designing a sound password-based remote user authentication scheme are presented. First, user anonymity, especially in some application scenarios, (e.g., e-commence), should be preserved, because from the identity ID_i, some personal secret information may be leaked about the user. Second, a nonce based mechanism is often a better choice than the timestamp based design to resist replay attacks, since clock synchronization is difficult and expensive in existing network environment, especially in wide area networks, and these schemes employing timestamp may still suffer from replay attacks as the transmission delay is unpredictable in real networks. Finally, the password change process should be performed locally without the hassle of interaction with the remote authentication server for the sake of security, user friendliness and efficiency. In this section, we present an improved remote user authentication scheme against smart card security breach.

4.1 Registration phase

Let $(x, y = g^x \bmod n)$ denote the server S's private key and its corresponding public key, where x is kept secret by the server and y is stored inside each user's smart card. The registration phase involves the following operations:

Step R1. U_i chooses his/her identity ID_i, password P_i and a random number b.
Step R2. $U_i \Rightarrow S : \{ID_i, h(b \parallel P_i)\}$.
Step R3. On receiving the registration message from U_i, the server S computes $N_i = h(b \parallel P_i) \oplus h(x \parallel ID_i)$ and $A_i = h(ID_i \parallel h(b \parallel P_i))$.
Step R4. $S \Rightarrow U_i$: A smart card containing security parameters $\{N_i, A_i, n, g, y, h(\cdot)\}$.
Step R5. Upon receiving the smart card, U_i enters b into his smart card.

4.2 Login phase

When U_i wants to login the system, the following operations will be performed:

Step L1. U_i inserts his/her smart card into the card reader and inputs ID_i^*, P_i^*.
Step L2. The smart card computes $A_i^* = h(ID_i^* \parallel h(b \parallel P_i^*))$ and verifies the validity of A_i^* by checking whether A_i^* equals to the stored A_i. If the verification holds, it implies $ID_i^* = ID_i$ and $P_i^* = P_i$. Otherwise, the session is terminated.
Step L3. The smart card chose a random number u and computes $C_1 = g^u \bmod n$, $Y_1 = y^u \bmod n$, $h(x \parallel ID_i) = N_i \oplus h(b \parallel P_i)$, $CID_i = ID_i \oplus h(C_1 \parallel Y_1)$ and $M_i = h(CID_i \parallel C_1 \parallel h(x \parallel ID_i))$.
Step L4. $U_i \rightarrow S : \{C_1, CID_i, M_i\}$.

4.3 Verification phase

After receiving the login request, the server S performs the following operations:

Step V1. The server S computes $Y_2 = (C_1)^x \bmod n$ using its private key x, and derives $ID_i = CID_i \oplus h(C_1 \parallel Y_2)$ and $M_i^* = h(CID_i \parallel C_1 \parallel h(x \parallel ID_i))$. S compares M_i^* with the received value of M_i. If they are not equal, the request is rejected. Otherwise, server S generates a random number v and computes the session key $SK = (C_1)^v \bmod n$, $C_2 = g^v \bmod n$ and $C_3 = h(SK \parallel C_2 \parallel h(x \parallel ID_i))$.
Step V2. $S \rightarrow U_i : \{C_2, C_3\}$.
Step V3. On receiving the reply message from the server S, U_i computes $SK = (C_2)^u \bmod n$, $C_3^* = h(SK \parallel C_2 \parallel h(x \parallel ID_i))$, and compares C_3^* with the received C_3. This equivalency authenticates the legitimacy of the server S, and U_i goes on to compute $C_4 = h(C_3 \parallel h(x \parallel ID_i) \parallel SK)$.
Step V4. $U_i \rightarrow S : \{C_4\}$
Step V5. Upon receiving $\{C_4\}$ from U_i, the server S first computes $C_4^* = h(C_3 \parallel h(x \parallel ID_i) \parallel SK)$ and then checks if C_4^* is equal to the received value of C_4 . If this verification holds, the server S authenticates the user U_i and the login request is accepted else the connection is terminated.
Step V6. The user U_i and the server S agree on the common session key SK for securing future data communications.

4.4 Password change phase

In this phase, we argue that the user's smart card must have the ability to detect the failure times. Once the number of login failure exceeds a predefined system value, the smart card must be locked immediately to prevent the exhaustive password guessing behavior. This phase involves the following local operations:

Step P1. U_i inserts his/her smart card into the card reader and inputs the identity ID_i and the original password P_i. The smart card computes $A_i^* = h(ID_i \parallel h(b \parallel P_i))$ and verifies the validity of A_i^* by checking whether A_i^* equals to the stored A_i. If the verification holds, it implies the input ID_i and P_i are valid. Otherwise, the smart card rejects.

Step P2. The smart card asks the cardholder to resubmit a new password P_i^{new} and computes $N_i^{new} = N_i \oplus h(b \parallel P_i) \oplus h(b \parallel P_i^{new})$, $A_i^{new} = h(ID_i \parallel h(b \parallel P_i^{new}))$. Thereafter, smart card updates the values of N_i and A_i stored in its memory with N_i^{new} and A_i^{new}.

5 Security analysis

In the following, we briefly analyze the enhanced security of the proposed scheme under the assumption that the secret information stored in the smart card can be revealed, i.e., the security parameters N_i, A_i and y can be obtained by a malicious privileged user. A comprehensive analysis is available in [7].

(1) User anonymity: Suppose that the attacker has intercepted U_i's authentication messages $\{CID_i, M_i, C_1, C_2, C_3, C_4\}$. Then, the adversary may try to retrieve any static parameter from these messages, but these messages are all session-variant and indeed random strings due to the randomness of u and/or v. Accordingly, without knowing the random number u, the adversary will face to solve the discrete logarithm problem to retrieve the correct value of ID_i from CID_i,, while ID_i is the only static element corresponding to U_i in the transmitted messages. Hence, the proposed scheme can preserve user anonymity.

(2) Offline password guessing attack: Suppose that a malicious privileged user U_i has got U_k's smart card, and the secret information b, N_k, A_k and y can also be revealed under our assumption of the non-tamper resistant smart card. Even after gathering this information, the attacker has to at least guess both ID_i and P_i correctly at the same time, because it has been demonstrated that our scheme can provide identity protection. It is impossible to guess these two parameters correctly at the same time in polynomial time, and thus the proposed scheme can resist offline password guessing attack with smart card security breach.

(3) Denial of service attack: Assume that an adversary $\mathcal{A}$ has got the legitimate user U_i's smart card. However, in our scheme, the smart card computes $A_i^* = h(ID_i \parallel h(b \parallel Pi))$ and compares it with the stored value of A_i in its memory to checks the validity of submitted ID_i and P_i before

the password update procedure. It is not possible for $\mathcal{A}$ to guess out U_i's identity ID_i and password P_i correctly at the same time in polynomial time. Moreover, once the number of login failure exceeds a predefined system value, the smart card will be locked immediately. Therefore, the proposed protocol is secure against denial of service attack.

(4) Forward secrecy: Following our scheme, the client and the server can establish the same session key $SK = (C_1)^v = (C_2)^u = g^{uv} \bmod n$. Based on the difficulty of the computational Diffie-Hellman problem, any previously generated session keys cannot be revealed without knowledge of the ephemeral u and v. As a result, our scheme provides forward secrecy.

6 Conclusion

In this paper, we have demonstrated several attacks on Li et al.'s scheme and a robust authentication scheme is thus proposed to remedy these identified flaws. The security analysis demonstrates our scheme eliminates several hard security threats that are difficult to be solved at the same time in previous scholarship.

References

1. Chang, C.C., Wu, T.C.: Remote password authentication with smart cards. IEE Proceedings-E 138(3), 165–168 (1993)
2. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 99. LNCS, vol. 1666, pp. 388–397. Springer-Verlag, Berlin (1999)
3. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining Smart-Card Security under the Threat of Power Analysis Attacks. IEEE Transactions on Computers 51(5), 541–552 (2002)
4. Kasper, T., Oswald, D., Paar, C.: Side-channel analysis of cryptographic RFIDs with analog demodulation. In: Juels, A., Paar, C. (eds.) RFID. Security and Privacy, LNCS, vol. 7055, pp. 61–77. Springer Berlin / Heidelberg (2012)
5. Li, C.T., Lee, C.C., Liu, C.J., Lee, C.W.: A Robust Remote User Authentication Scheme against Smart Card Security Breach. In: Li, Y. (ed.) DBSec 2011. LNCS, vol. 6818, pp. 231–238. Springer, Heidelberg (2011)
6. Kim, S.K., Chung, M.G.: More secure remote user authentication scheme. Computer Communications 32(6), 1018–1021 (2009)
7. Wang, D., Ma, C.G., Wu P.: Secure Password-based Remote User Authentication Scheme with Non-tamper Resistant Smart Cards. NSR Technical Report 2012/011 (2012), http://machunguang.hrbeu.edu.cn/Research/
8. Ma, C.G., Wang, D., Zhang, Q.M.: Cryptanalysis and improvement of Sood et al.'s dynamic id-based authentication scheme. In: Ramanujam, R., Ramaswamy, S., (eds.): Distributed Computing and Internet Technology, LNCS, vol. 7154, pp. 141–152. Springer Berlin / Heidelberg (2012)