Hindawi
Security and Communication Networks
Volume 2018, Article ID 9062675, 15 pages
https://doi.org/10.1155/2018/9062675

*Research Article*

# A Secure and Anonymous Two-Factor Authentication Protocol in Multiserver Environment

**Chenyu Wang**,[1] **Guoai Xu**,[1] **and Wenting Li**[2]

[1] *School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China*
[2] *School of Software and Microelectronics, Peking University, Beijing, China*

Correspondence should be addressed to Guoai Xu; xga@bupt.edu.cn

Received 28 August 2017; Accepted 29 January 2018; Published 15 April 2018

Academic Editor: Shujun Li

Copyright © 2018 Chenyu Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the great development of network technology, the multiserver system gets widely used in providing various of services. And the two-factor authentication protocols in multiserver system attract more and more attention. Recently, there are two new schemes for multiserver environment which claimed to be secure against the known attacks. However, after a scrutinization of these two schemes, we found that (1) their description of the adversary's abilities is inaccurate; (2) their schemes suffer from many attacks. Thus, *firstly*, we corrected their description on the adversary capacities to introduce a widely accepted adversary model *and then* summarized fourteen security requirements of multiserver based on the works of pioneer contributors. *Secondly*, we revealed that one of the two schemes fails to preserve forward secrecy and user anonymity and cannot resist stolen-verifier attack and off-line dictionary attack and so forth and also demonstrated that another scheme fails to preserve forward secrecy and user anonymity and is not secure to insider attack and off-line dictionary attack, and so forth. *Finally*, we designed an enhanced scheme to overcome these identified weaknesses, proved its security via BAN logic and heuristic analysis, and then compared it with other relevant schemes. The comparison results showed the superiority of our scheme.

## 1. Introduction

The development of network technology has greatly changed the way people live and work. Internet brings our society into an information age, and it has become an indispensable element of people's life. Nowadays, with the maturity and rapid development of Internet technology, people's schedule was more convenient and efficient due to the increasing online services. However, the openness and virtuality of the Internet have resulted in the fact that the network environment became untrusted which is accompanied by the information security and privacy issues. In recent years, we have heard too many events about user privacy information being leaked; for example, in 2015, about 10 G user data of Ashley Madison (the world's largest extramarital affairs web site who offers dating services for married people) has been exposed. In this event, many celebrities were exposed, and the whole society was surrounded by fear; Anthem lost 80 million user datasets including user name, birthday, social insurance code, phone number, email, and so on, which is the largest medical institution user data exposed event in the United States. For a more secure network environment, the cryptographic approach is one of the key technologies, among which a necessary part is to provide authentication and key agreement for remote entities. And this mechanism is called user authentication.

Usually, a well-defined authentication scheme should promise that only the legitimate user can enjoy the service, and the corresponding server is exactly real and legitimate. At the beginning, the passwords, with its facility and accessibility, have been used widely in authentication process. While it has been found that the password-based single-server authentication protocols always risk in stolen-verifier attack, because the server has to maintain a password related table. Thus, the smart card, as a second security factor, gets widely used [1–5].

Furthermore, the increasing demands in network life greatly prompted that service providers extend the traditional single-server environment into a multiserver one to offer more kinds of services and improve their quality of services.

Then multiserver system comes into being. However, the single-server-based authentication scheme is not suitable to multiserver system any more: the single-server-based scheme asks a user to register on each server one by one, and so the users have to remember many different identities and passwords, which is bound to bring unnecessary trouble for the users; in order to remember a mountain of identities and passwords, a user is more likely to choose the same identity and password; thus, the information disclosure emerged in multiserver system.

To solve this problem, scholars put forward the multiserver environment authentication mechanism whose goals are that the user only needs to register to the registration center, and then he/she can login to the corresponding different application servers using the same account. This ideal is also of high reference value to cloud computing, Internet of things, car networking, and so forth. In 2001, Li et al. [6] proposed a neural-networks-based scheme for multisever system: its communication and computation costs are very high and, furthermore, the users have to store large amount of data. In 2003, Lin et al. [7] proposed a new scheme with lower costs, which was pointed out to be inefficient by Juang [8]. Thus he recommended a symmetric-cryptography-based protocol which resolved the problem of reregistration with high computational efficiency. Unfortunately, Chang and Lee [9] revealed that Juang's scheme suffers from off-line password guessing attack, and the users cannot change their password; therefore, they proposed a new improved scheme. In 2004, Tsaur et al. [10] demonstrated that Chang and Lee's scheme is vulnerable to insider attack and forgery attack, so they designed a RSA-based scheme. Once again, their scheme was noted to be subjected to impersonation attack [11].

Those schemes above have a common problem: the user identity is static; thus they usually fail to achieve perfect user anonymity. To remedy this problem, Liao and Wang [12] in 2009 proposed a dynamic-identity-based protocol, while later it was proved to be insecure to impersonation attack and insider attack by Hsiang and Shih [13]. Unfortunately, Sood et al. [14] revealed that Hsiang and Shih's scheme is not as secure as they claimed.

*1.1. Contributions.* Recently, Li et al. [15] and Sood [16] proposed a user authentication scheme in multiserver networks; they both claimed to be secure to various known attacks. However, in 2016, Amin [17] demonstrated the two schemes cannot resist off-line guessing attack, insider attack, and so on, therefore providing a new enhanced protocol overcoming those weaknesses. In the same year, Maitra et al. [18] reexamined Leu and Hsieh's scheme [19] and Li et al.'s scheme [20] and found that their schemes were subject to many security threats; thus they also put forward a new scheme using symmetric cryptosystem and aiming to resist various attacks with some desire attributes. Unfortunately, according to our analysis, their schemes, once again, fail to be a sound authentication protocol. To point out the common issues in the user authentication scheme, we use these two advanced and representative schemes as study case to show the possible weakness in most schemes. Then, based on the analysis, we propose an improved scheme trying to show a possible way to overcome those weakness. In a word, our contributions can be summarized as follows:

(1) We revealed the description of adversary's abilities in many schemes are inaccurate and thus redescribed a widely accepted and practical adversary model.

(2) We summarized fourteen security requirements of multiserver environment based on the works of pioneer contributors.

(3) We demonstrate that Maitra et al.'s scheme [18] fails to preserve forward secrecy and user anonymity and cannot resist stolen-verifier attack and off-line dictionary attack and so on; Amin's scheme [17] fails to preserve forward secrecy and user anonymity and is not secure to insider attack and off-line dictionary attack and so on.

(4) We propose an enhanced scheme with user anonymity and proved its security via BAN logic and heuristic analysis and, furthermore, compared it with other relevant schemes. The comparison result shows that our scheme, though increasing the costs slightly, achieves all the fourteen security requirements, so it is more suitable to multiserver.

*1.2. Construction of the Paper.* In Section 2, we described the preliminaries and then analyzed Maitra et al.'s scheme [18] and Amin's scheme [17] in Sections 3 and 4, respectively. And we proposed a new scheme in Section 5, proved its security in Section 6, and analyzed its performance in Section 7. The conclusion was given in Section 8.

## 2. Preliminaries

For better understanding of the two-factor authentication scheme in multiserver environment, it is necessary to describe the computational problems, communication model, adversary model, and security requirements firstly.

### *2.1. Computational Problems*

(1) Discrete logarithm problem: given $(g, g^{\alpha} \bmod p)$, it is hard to compute $\alpha$ ($\alpha \in Z_p^*$) within the polynomial time, where $g$ is the generator of cyclic group $Z_p^*$.

(2) Computational Diffie-Hellman problem: given $(g^{\beta}, g^{\alpha} \bmod p)$, it is hard to compute $g^{\alpha\beta}$ within the polynomial time, where $\alpha, \beta \in Z_p^*$.

*2.2. Communication Model.* A multiserver environment (shown in Figure 2) refers to the fact that a service provider can offer a variety of services for the users, for example, Google, who not only provides mail service but also provides news, video, and other services. To a user, he/she only needs to have one account of Google and then can enjoy all the services provided by it. And the way to implement this function is what we know as the user authentication protocol in a multiserver environment. Usually, people may be more familiar with distributed systems (shown in Figure 1) where each service corresponds to a server, and it only

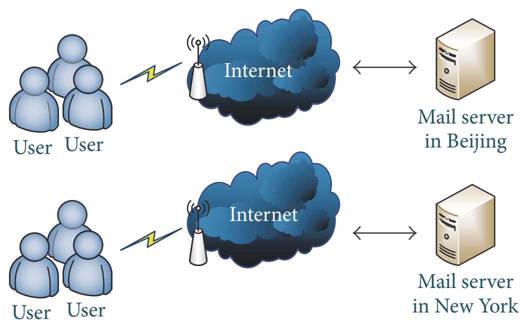


Figure 1: The architecture of the distributed system.

Table 1: Notations and abbreviations

| Symbol | Description |
|---|---|
| $U_i$ | $i$th user |
| $S_j$ | $j$th remote server |
| $RC$ | The register center |
| $\mathscr{A}$ | The adversary |
| $x$ | The secret key of $S$ |
| $ID_i$ | Identity of $U_i$ |
| $PW_i$ | Password of $U_i$ |
| $SID_j$ | Identity of $S_j$ |
| $Skey_j$ | The shared secret key between $RC$ and $S_j$ |
| $\oplus$ | Bitwise XOR operation |
| $\|$ | Concatenation operation |
| $h(\cdot)$ | One-way hash function |
| $\rightarrow$ | A common channel |
| $\Rightarrow$ | A secure channel |

involves two participants: a set of users and a single server. However, typically, a two-factor authentication protocol in multiserver environment involves three participants: a set of users, a set of servers, and a register center. Among these participants, only the register center is trusted; it may store some sensitive information in the database. Furthermore, the authentication process usually consists of four basic phases: registration, login, authentication, and password change phase. The registration phase includes two parts: user registration phase, where a user submits his/her personal information, and then the register center issues the user a smart card containing security messages; server registration phase, where the servers send their identities to the register center to get a secret key. In login phase, the user selects a server to offer service and sends a login request to the server. Then in authentication phase, the user and the server need to verify the legitimacy of each other. Furthermore, according to whether the registration center is involved in the authentication process, the multiserver authentication protocols can be divided into two categories: the registration center involved one; the registration center did not involve one. Among the four phases, only the registration phase is carried out via a secure channel and the others are all conducted via an insecure channel. And the notations used in the protocols are shown in Table 1.

*2.3. Adversary Model.* In fact, both Amin [17] and Maitra et al. [18] described the capacities of the adversaries inaccurately. In their adversary model, there are three obvious flaws which are overlooked but critical in authentication protocol.

The first one is about "whether an adversary can exhaust the password space and identity space to conduct off-line dictionary attack simultaneously?". Many schemes [17, 18, 20, 21] think that $\mathscr{A}$ can exhaust either the password space or the identity space, but not simultaneously. While it is really not practical, Wang et al. [22] for the first time revealed that user-chosen passwords follow the Zipf-like distribution, a distribution far from uniform. This indicates that user-chosen passwords are prone to static guessing attacks. Furthermore, in [23] Section 4.2, we can see that even the adversary guesses the password and identity simultaneously and the whole attack can be finished within limited time. Therefore, the adversary can exhaust the password and identity space simultaneously, and many scholars follow this principle [2, 24–27].

The second one is about "whether an adversary can easily get a user's identity once owning the user's smart card?"; the answer is also positive. As Wang et al. [28] explained, on the one hand, the identity usually is a static short string with limited space. And the same user is accustomed to using the same identity even for different service providers. So it is of high possibility that an adversary learns the identity from other common service providers; on the other hand, the users do not regard identity as a secret parameter, and, for easy remembrance, they will even write the identity on the card directly. So when cryptanalyzing a scheme, it is more practical to assume that the identity is an open parameter.

The third one is about "whether an adversary can get the long term secret key?". Maitra et al.'s work [18] just ignored this problem and supposed that the adversary can never learn about the long term secret key; as, for Amin's work, he assumed that a valid user can always know the secret information and may provide it to the adversary, while, in fact, these two statements are both not accurate enough. A widely accepted assumption is that an adversary can know the long term secret key only when evaluating the forward secrecy [28–32].

Besides, it is widely accepted that an adversary has the full control of the channel; that is, $\mathscr{A}$ can intercept, delete, modify, resend, and reroute the messages in an open channel [33–35]. Furthermore, $\mathscr{A}$ may also learn users' passwords via a malicious terminal or extract the parameters from the smart card by side-channel attack, but cannot achieve both [2, 27, 36]. We summarize the capacities of the adversary $\mathscr{A}$ in Table 2.

*2.4. Security Requirements.* According to the user authentication protocols in multiserver environment [34, 37, 38] and some works on analysis of security requirements in user authentication scheme [2, 4, 28], we describe the security requirements in a two-factor authentication scheme of multiserver in Table 3.

## 3. Review of the Scheme of Maitra et al.

In 2016, Maitra et al. [18] criticized two recent protocols, namely, Leu and Hsieh's scheme [19] and Li et al.'s

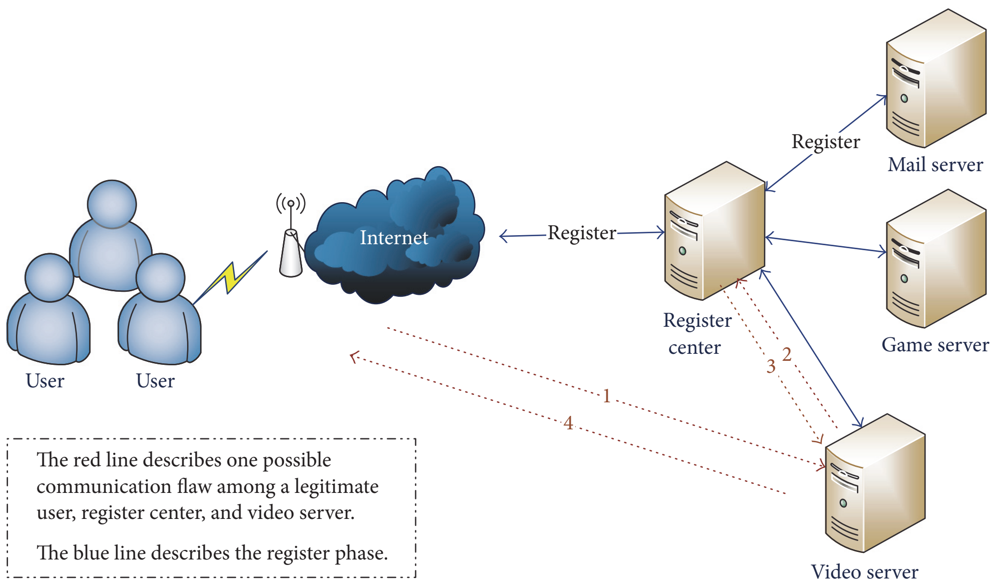

Register
Mail server
Internet
Register
Register center
Game server
User
User
1
2
3
4
The red line describes one possible communication flaw among a legitimate user, register center, and video server.
The blue line describes the register phase.
Video server

Figure 2: The architecture of the multiserver system.

Table 2: The capacities of the adversary.

| | |
|---|---|
| (1) | $\mathscr{A}$ can intercept, modify, delete, and resend the messages between the users and the servers over the open channel. |
| (2) | $\mathscr{A}$ can enumerate all the items in the password space and identity space simultaneously. |
| (3) | When cryptanalyzing a scheme, $\mathscr{A}$ can know the identity of $U_i$. |
| (4) | $\mathscr{A}$ knows the identity of $S_j$. |
| (5) | $\mathscr{A}$ can learn the password of $U_i$ by a malicious card reader, or extract the parameters from the smart, but cannot achieve both. |
| (6) | When evaluating forward secrecy, $\mathscr{A}$ knows the long term secret key of the register center. |

scheme [20], and pointed out that the two schemes are vulnerable to various attacks, such as forgery attack, and password guessing attack; therefore, they designed a new enhanced scheme being confident to resist a variety of known attacks and with some attractive attributes such as freely changing password and identity. However, when we reexamined their scheme, we found some serious security threats of the scheme and revealed that the scheme is not secure against verifier-stolen attack and off-line password guessing attack and also fails to provide forward secrecy and user anonymity.

### 3.1. The Scheme of Maitra et al.

In this section, we review Maitra et al.'s scheme [18] briefly, and as password change phase and identity change phase have little relevance to our work, we omit them.

#### 3.1.1. Initialization Phase.

*RC* selects a secret long key $x$ and a symmetric key encryption/decryption Enc/Dec algorithm (AES); then there is a hash function $h(*): \{0,1\}^* \rightarrow \{0,1\}^n$.

#### 3.1.2. Server Registration Phase

*Step 1.* $S_j \Rightarrow RC$: $\{SID_j\}$.

*Step 2.* $RC \Rightarrow S_j$: $\{Skey_j, SID_j\}$. *RC* first checks the availability of $SID_j$, then calculates $Skey_j = h(SID_j \parallel x)$, and adds $\{SID_j, key_{s_j}\}$ into the *Server-list*, where $key_{s_j} = Skey_j \oplus x$, finally sends $\{Skey_j, SID_j\}$ to $S_j$; otherwise, *RC* rejects the server $S_j$'s request.

*Step 3.* $S_j$ stores $Skey_j$.

#### 3.1.3. User Registration Phase

*Step 1.* $U_i \Rightarrow RC$: $\{DID_i, PWR_i\}$. $U_i$ selects $PW_i$, $ID_i$, and a random number $b_i$, computes $DID_i = h(ID_i \parallel b_i)$, $PWR_i = h(PW_i \parallel b_i)$, and then sends $\{DID_i, PWR_i\}$ to *RC*.

*Step 2.* $RC \Rightarrow U_i$: a smart card with $\{C_i, D_i, E_i\}$. *RC* tests $DID_i$ by the *User-list*, chooses a random number $y_i$, calculates $C_i = h(x \parallel y_i) \oplus DID_i \oplus PWR_i$, $E_i = h(h(x \parallel y_i) \parallel DID_i \parallel PWR_i)$, and $D_i = Enc_x[DID_i \parallel PWR_i]$, then stores $DID_i$ into the *User-list*, and issues $U_i$ a smart card with $\{C_i, D_i, E_i\}$.

*Step 3.* $U_i$ computes $\widetilde{b}_i = b_i \oplus h(ID_i \parallel PW_i)$ and stores it.

#### 3.1.4. Login and Authentication Phase

*Step 1.* $U_i \rightarrow S_j$: $Login_{U_i}$. $U_i$ inputs $ID_i$ and $PW_i$. The card computes $b_i' = \widetilde{b}_i \oplus h(ID_i \parallel PW_i)$, $DID_i' = h(ID_i \oplus b_i')$, $PWR_i' = h(PW_i \parallel b_i')$, $[h(x \parallel y_i)]' = C_i \oplus DID_i' \oplus PWR_i'$, and $E_i' = h([h(x \parallel y_i)]' \parallel DID_i' \parallel PWR_i')$. If $E_i' \neq E_i$, end the session. Otherwise, the card chooses a random number $r_i$ and timestamp $T_i^1$, computes $G_i = r_i \oplus h(DID_i' \parallel PWR_i' \parallel T_i^1)$, $F_{ij} = h(DID_i' \parallel PWR_i' \parallel r_i \parallel T_i^1 \parallel SID_j)$, and sends $Login_{U_i} = \{G_i, F_{ij}, D_i, F_i, SID_j, T_i^1\}$ to $S_j$.

Table 3: Security requirements.

| | |
|---|---|
| S1: mutual authentication | $RC$, $S_j$ and $U_i$ should authenticate each other to ensure their eligibility. |
| S2: user anonymity | $\mathscr{A}$ can neither compute the identity of $U_i$ nor link the message flows to $U_i$. |
| S3: key agreement | $S_j$ and $U_i$ should share a session key for further communication. |
| S4: forward secrecy | Even gets the long term secret key $x$, $\mathscr{A}$ still cannot compute the session key. |
| S5: password friendly | The user $U_i$ can select and change his password locally. |
| S6: sound repairability | $U_i$ can revoke the breached smart card and re-register with the same identity. |
| S7: no stolen-verifier attack | Even $RC$ stores a verifier table, $\mathscr{A}$ gains no benefits from it. |
| S8: no insider attack | The participants with the message their know cannot conduct an attack. |
| S9: no dictionary attack | With all the abilities in Table 2, $\mathscr{A}$ still cannot guess the $ID_i$ and $PW_i$. |
| S10: no replay attack | $\mathscr{A}$ cannot replay the eavesdropped messages to conduct an attack. |
| S11: no parallel session attack | $\mathscr{A}$ may construct multi-session simultaneously, but $\mathscr{A}$ gains no benefits from it. |
| S12: no desynchronization attack | On the one hand, the scheme should not suffer from desynchronization attack On the other hand, it needs not to synchronize the clock. |
| S13: no impersonation attack | $\mathscr{A}$ cannot impersonate the user or any other participants. It needs to note that (1) $\mathscr{A}$ here cannot breach the smart card, while in dictionary attack $\mathscr{A}$ has that capability; (2) $\mathscr{A}$ can be a legitimate user or server. |
| S14: no known key attack | Knowing the current session keys, $\mathscr{A}$ cannot compute other session key in the future or in the past. |

*Step 2.* $S_j \rightarrow RC$: $Login_{S_j}$. $S_j$ first checks $SID_j$ and the freshness of $T_i^1$, then computes $H_{ji} = Enc_{Skey_j}[r_j \parallel Login_{U_i} \parallel T_j^1 \parallel SID_j]$, and sends $Login_{S_j} = \{H_{ji}, SID_j, T_j^1\}$ to $RC$.

*Step 3.* $RC \rightarrow S_j$: $\{R_{rc_{ij}}\}$. $RC$ first tests $T_j^1$ and $SID_j$, catches corresponding $Skey_j$ ($Skey_j = key_{S_j} \oplus x$) from the $Server\text{-}list$, decrypts $H_{ji}$ with $Skey_j$, and then checks $T_j^1$ ?= $T_j^{1*}$ and $SID_j$ ?= $SID_j^*$ ?= $SID_j^{**}$. If one of the equations does not hold, end the session. Otherwise, $RC$ computes $(DID_i^* \parallel PWR_i^*) = Dec_x[D_i^*]$, $r_i = G_i^* \oplus h(DID_i^* \parallel PWR_i^* \parallel T_i^{1*})$ and compares $h(DID_i^* \parallel PWR_i^* \parallel T_i^{1*} \parallel SID_j)$ with $F_{ij}^*$. If they are not equal, end the session. Otherwise $RC$ computes $P_{rc_{ij}} = h(x \parallel r_{rc} \parallel r_j \parallel r_i)$, $K_{rc_{ij}} = h(DID_i \parallel r_i \parallel PWR_i) \oplus P_{rc_{ij}}$, $L_{rc_{ij}} = P_{rc_{ij}} \oplus r_j$, $N_{rc_{ij}} = h(r_j \parallel SID_j \parallel P_{rc_{ij}})$, $R_{rc_{ij}} = Enc_{Skey_j}[K_{rc_{ij}} \parallel L_{rc_{ij}} \parallel N_{rc_{ij}} \parallel T_{rc}]$ and finally answers $S_j$ with $\{R_{rc_{ij}}\}$.

*Step 4.* $S_j \rightarrow U_i$: $\{Q_{ji}, W_{ji}, K_{rc_{ij}}{}^{\dagger}, T_j^2\}$. $S_j$ computes $(K_{rc_{ij}}^{\dagger} \parallel L_{rc_{ij}}^{\dagger} \parallel N_{rc_{ij}}^{\dagger} \parallel T_{rc}^{\dagger}) = Dec_{Skey_j}[R_{rc_{ij}}]$, checks $T_{rc}^{\dagger}$, then computes $P_{rc_{ij}}^{\dagger} = L_{rc_{ij}}^{\dagger} \oplus r_j$. If $h(r_j \parallel SID_j \parallel P_{rc_{ij}}^{\dagger}) == N_{rc_{ij}}^{\dagger}$, $S_j$ computes $[h(DID_i \parallel r_i \parallel PWR_i)]^{\dagger} = P_{rc_{ij}}^{\dagger} \oplus K_{rc_{ij}}^{\dagger}$, $W_{ji} = n_j \oplus [h(DID_i \parallel r_i \parallel PWR_i)]^{\dagger}$, $Q_{ji} = h([h(DID_i \parallel r_i \parallel PWR_i)]^{\dagger} \parallel n_j \parallel T_j^2)$, where $n_j$ is a random number, and then sends $U_i$ with $\{Q_{ji}, W_{ji}, K_{rc_{ij}}{}^{\dagger}, T_j^2\}$. Otherwise, exit.

*Step 5.* $U_i \rightarrow S_j$: $\{V_i, T_i^2\}$. The smart card first tests $T_j^2$ and then computes $n_j' = W_{ji} \oplus h(DID_i \parallel r_i \parallel PWR_i)$, $Q_{ji}' = h(h(DID_i \parallel r_i \parallel PWR_i) \parallel n_j' \parallel T_j^2)$. If $Q_{ji}' == Q_{ji}$, the smart card computes $P_{rc_{ij}}' = K_{rc_{ij}}^{\dagger} \oplus h(DID_i \parallel r_i \parallel PWR_i)$, the session key $SK_i = h(P_{rc_{ij}}' \parallel n_j')$, $V_i = h(SK_i \parallel T_i^2)$ and sends $\{V_i, T_i^2\}$ to $S_j$. Otherwise, end.

*Step 6.* After checking the freshness of $T_i^2$, $S_j$ computes $SK_j = h(P_{rc_{ij}}^{\dagger} \parallel n_j)$; $V_i' = h(SK_j \parallel T_i^2)$, compares $V_i'$ with the received $V_i$. If they are equal, the authentication is finished successfully; both $U_i$ and $S_j$ accept the session key $SK_i$ $(= SK_j)$.

### 3.2. Cryptanalysis of Maitra et al.'s Scheme.

It has to admit that Maitra et al.'s scheme has many attractive advantages, such as providing password and identity change phase. Furthermore, the way to protect the real identity and password is somewhat illuminating. While it is regrettable that this scheme is still not secure against various attacks, including stolen-verifier attack, off-line password guessing attack, and no forward secrecy and user anonymity.

#### 3.2.1. Off-Line Dictionary Attack.

In Section 2.3, we explain that $\mathscr{A}$ can guess the identity and password simultaneously and also can learn the identity. No matter whether $\mathscr{A}$ knows about the identity, he/she can carry out the off-line dictionary attack. Here, we take $\mathscr{A}$ not knowing the identity as an example. Suppose $\mathscr{A}$ steals $U_i$'s smart card and extracts $\{C_i, E_i, \widetilde{b}_i\}$ from the smart card; then he can perform off-line dictionary attack as the following steps.

*Step 1.* Guess the value of $PW_i$ to be $PW_i^*$ from the password dictionary space $\mathscr{D}_{pw}$, the value of $ID_i$ to be $ID_i^*$ from the identity dictionary space $\mathscr{D}_{id}$.

*Step 2.* Compute $b_i' = \widetilde{b}_i \oplus h(ID_i^* \parallel PW_i^*)$.

*Step 3.* Compute $DID_i' = h(ID_i^* \oplus b_i')$.

*Step 4*. Compute $PWR_i' = h(PW_i^* \parallel b_i')$.

*Step 5*. Compute $[h(x \parallel y_i)]' = C_i \oplus DID_i' \oplus PWR_i'$.

*Step 6*. Compute $E_i' = h([h(x \parallel y_i)]' \parallel DID_i' \parallel PWR_i')$.

*Step 7*. Verify the correctness of $PW_i$ and $ID_i$ by checking if $E_i' =?E_i$.

*Step 8*. Repeat Steps 1~6 until the correct value of $PW_i$ and $ID_i$ are found.

With $PW_i$ and $ID_i$, the adversary $\mathscr{A}$ can impersonate the user $U_i$ to enjoy the service.

The time complexity of the above attack is $\mathcal{O}(|\mathscr{D}_{pw}| * |\mathscr{D}_{id}| * 4T_H)$, where $T_H$ is the running time for hash computation; $|\mathscr{D}_{pw}|$ and $|\mathscr{D}_{id}|$ denote the number of passwords in $\mathscr{D}_{pw}$ and the number of identities in $\mathscr{D}_{id}$, respectively. $|\mathscr{D}_{pw}|$ is very limited due to the Zipf's law in passwords [22]; $|\mathscr{D}_{id}|$ is also very limited as generally $|\mathscr{D}_{id}| < |\mathscr{D}_{pw}|$. So the attack can be finished in the polynomial time.

*3.2.2. Forward Secrecy.* Suppose an adversary $\mathscr{A}$ somehow learns the long term secret key $x$ and eavesdrops the message in the open channel to get $H_{ji}$, $R_{rc_{ij}}$, $W_{ji}$; then he/she can compute the session key between $S_j$ and $U_i$ as follows.

*Step 1*. Compute $S_j$'s secret key $Skey_j = h(SID_j \parallel x)$, where $SID_j$ is an open parameter.

*Step 2*. Decrypt $H_{ji}$ with $Skey_j$, and get $r_j$, $Login_{U_i}$, where $Login_{U_i} = \{G_i, F_{ij}, D_i, F_i, SID_j, T_i^1\}$ and $H_{ji}$ is from the open channel.

*Step 3*. Decrypt $D_i$ with $x$ to get $DID_i$ and $PWR_i$, where $D_i$ is from the open channel.

*Step 4*. Decrypt $R_{rc_{ij}}$ with $Skey_j$ to get $L_{rc_{ij}}$, where $R_{rc_{ij}}$ is from the open channel.

*Step 5*. Compute $P_{rc_{ij}} = L_{rc_{ij}} \oplus r_j$.

*Step 6*. Compute $n_j = W_{ji} \oplus h(DID_i \parallel r_i \parallel PWR_i)$, where $W_{ji}$ is from the open channel.

*Step 7*. Compute the session key $SK_i = h(P_{rc_{ij}} \parallel n_j)$.

The time complexity of the above attack is $\mathcal{O}(3T_H + 3T_S)$, where $T_S$ is the running time of symmetric encryption operation. According to the TABLE VI. in [2], the attack can be finished within seconds. So the above attack can be completed in the polynomial time.

*3.2.3. Verifier-Stolen Attack.* As we mentioned before, only the register center is trusted, the user and the server are both likely to be an adversary $\mathscr{A}$ to conduct an attack. Consider such a condition where the legitimate server $S_j$ somehow gets the verifier table in the database of register center. Then this adversary $\mathscr{A}$ can also compute $x$ and, furthermore, damage the whole system as follows.

*Step 1*. Compute $x = key_{s_j} \oplus Skey_j$, where $key_{S_j}$ is from the list $\{SID_j, key_j\}$ of the verifier table.

*Step 2*. Compute any other server $S_k$'s private secret key $Skey_k = Key_{S_k} \oplus x$, where $Key_{S_k}$ is from the list $\{SID_k, key_k\}$ of the verifier table.

The operations in the above procedure are some lightweight operation and the procedure is very simple.

With $x$ and the verifier table, $\mathscr{A}$ has the same capacity with the register center. Thus $\mathscr{A}$ can impersonate $RC$ to the user and the other server. What is more, with $x$, once $\mathscr{A}$ intercepts the message $H_{ji}$ or $H_{ki}$ $(k \neq j)$, $\mathscr{A}$ can compute any user's $PWR_i$ and $DID_i$ as the way $RC$ do. Furthermore, with $Skey_k$, $\mathscr{A}$ has the same capacity with other server, so he/she can also impersonate other servers to $RC$ and the users. Therefore, the security of the whole system is compromised.

*3.2.4. User Anonymity.* In this era of information explosion, user privacy protection is extremely important to the individuals. And user anonymity, as a pivotal way to protect the user privacy, contains two requirements: do not expose the identity directly; keep the identity untraceable. Once user anonymity cannot get guaranteed, the adversary may link the different communication in open channel to the same user and thus learns his preference and personal information for marketing purpose or other horrible purpose.

In Maitra et al.'s scheme [18], there is a static value $D_i$ in the open channel. More specifically, to the same user, $D_i$ is unchanged ($D_i = Enc_x[DID_i \parallel PWR_i]$) unless $U_i$ changes his identity and password. While the frequency of changing the identity or the password is so low, which means every time $U_i$ initiates an access request to any servers, the same $D_i$ will be transmitted in the open channel in most occasions. Therefore, an adversary $\mathscr{A}$ can link the access request to the same user from the huge amounts of data to learn the user's habits and preferences. So this scheme violates user untraceability.

More specifically, an adversary can eavesdrop the open channel and then get the following message:

$$Login_{U_i^1} = \left\{G_i^1, F_{ij}^1, D_i^1, F_i^1, SID_j, T_i^{11}\right\},$$
$$Login_{U_m^1} = \left\{G_m^1, F_{mj}^1, D_m^1, F_m^1, SID_j, T_m^{11}\right\},$$
$$Login_{U_g^1} = \left\{G_g^1, F_{gj}^1, D_g^1, F_g^1, SID_j, T_g^{11}\right\},$$
$$Login_{U_i^2} = \left\{G_i^2, F_{ij}^2, D_i^2, F_i^2, SID_k, T_i^{12}\right\},$$
$$Login_{U_m^2} = \left\{G_m^2, F_{mj}^2, D_m^2, F_m^2, SID_k, T_m^{12}\right\},$$
$$Login_{U_g^2} = \left\{G_g^2, F_{gj}^2, D_g^2, F_g^2, SID_k, T_g^{12}\right\},$$
$$\vdots$$
$$Login_{U_i^n} = \left\{G_i^n, F_{ij}^n, D_i^n, F_i^n, SID_m, T_i^{1n}\right\},$$

$$Login_{U_m^n} = \left\{G_m^n, F_{mj}^n, D_m^n, F_m^n, SID_m, T_m^{1n}\right\},$$
$$Login_{U_g^n} = \left\{G_g^n, F_{gj}^n, D_g^n, F_g^n, SID_m, T_g^{1n}\right\}. \quad (1)$$

As $D_i^1 == D_i^2 == D_i^n == D_i ==$ a constant value, the adversary knows that among those messages, $Login_{U_i^1}$, $Login_{U_i^2}$, and $Login_{U_i^3}$ were sent by the same user; this user usually accesses $S_j$, $S_k$, and $S_m$ at times $T_i^{11}$, $T_i^{12}$, and $T_i^{1n}$, respectively. Thus the user untraceability is violated. Furthermore, once the adversary acquires $ID_i$ and $PW_i$ as we showed in Section 3.2.1, he can compute $D_i$ and thus traces the specific user $U_i$ and learns more about the victim's habits.

## 4. Review of the Scheme of Amin

In 2016, Amin [17] showed two protocols [15, 16] both suffer from off-line guessing attack, impersonation attack, and so forth; thus he improved the two schemes to a new one claiming to be resistant to all known attacks, while, once again, we found Amin's scheme is not as secure as his claim. In this section, we demonstrate that this scheme is vulnerable to off-line dictionary attack and insider attack and fails to achieve forward secrecy and user anonymity.

*4.1. The Scheme of Amin.* The authentication process of Amin's scheme [17] is shown as follows briefly.

*4.1.1. Server Registration Phase*

*Step 1.* $S_j \Rightarrow RC$: $\{SID_j\}$.

*Step 2.* $RC \Rightarrow S_j$: $RC$ calculates $Skey_j = h(SID_j \parallel x)$ and then sends $\{Skey_j\}$ to $S_j$.

*Step 3.* $S_j$ keeps $\{Skey_j\}$ as his secret key.

*4.1.2. User Registration Phase*

*Step 1.* $U_i \Rightarrow RC$: $\{ID_i, PWR_i\}$. $U_i$ selects $PW_i$, $ID_i$, and a random number $b_i$, computes $PWR_i = h(PW_i \parallel b_i)$, and then sends $\{ID_i, PWR_i\}$.

*Step 2.* $RC \Rightarrow U_i$: smart card $\{CID_i, E_i, T_i, y_i, h(\cdot)\}$. $RC$ calculates $CID_i = h(ID_i \oplus y_i \oplus x)$, where $y_i$ is a random number, then checks the availability of $CID_i$, stores $CID_i$ into *User-list*, computes $E_i = h(ID_i \parallel PWR_i \parallel CID_i)$, $T_i = h(CID_i \parallel x) \oplus PWR_i$, finally, stores $\{CID_i, E_i, T_i, y_i, h(\cdot)\}$ into a smart card, and sends it to $U_i$.

*Step 3.* $U_i$ inputs $b_i$ into the card.

*4.1.3. Login and Authentication Phase*

*Step 1.* $U_i \rightarrow RC$: $\{CID_i, SID_j, T_i, L_3, L_2, N_3\}$. $U_i$ inputs $ID_i$ and $PW_i$. The card computes $PWR_i^* = h(PW_i \oplus b_i)$, $E_i^* = h(ID_i \parallel PW_i \parallel CID_i)$. If $E_i^* \neq E_i$, exit the session. Otherwise, the card generates two random numbers $N_1$ and $N_2$ and computes $L_1 = T_i \oplus PWR_i^*$, $N_3 = N_1 \oplus N_2$, $L_2 = N_2 \oplus PWR_i^*$, $L_3 = h(L_1 \parallel SID_j \parallel N_1 \parallel L_2 \parallel N_3)$.

*Step 2.* $RC \rightarrow S_j$: $\{CID_i, A_4, A_3, N_5\}$. $RC$ first checks $CID_i$ and $SID_j$ and then computes $A_i = h(CID_i \parallel x)$, $PWR_i' = T_i \oplus A_i$, $N_2' = L_2 \oplus PWR_i'$, $N_1' = N_3 \oplus N_2'$, $L_3' = h(A_1 \parallel SID_j \parallel N_1' \parallel L_2 \parallel N_3)$. If $L_3' \neq L_3$, reject the request; otherwise, $RC$ computes $A_2 = h(SID_j \parallel x)$, $A_3 = A_2 \oplus N_4$, $N_5 = N_1' \oplus N_4$, $A_4 = h(A_2 \parallel N_4 \parallel N_1 \parallel CID_i)$.

*Step 3.* $S_j \rightarrow U_i$: $\{SID_j, A_5, N_7\}$. $S_j$ computes $N_4' = Skey_j \oplus A_3$, $N_1' = N_4' \oplus N_5$, $A_4' = h(Skey_j \parallel N_4' \parallel N_1' \parallel CID_i)$. If $A_4' \neq A_4$, exit; otherwise, $S_j$ authenticates $RC$, chooses a random number $N_6$, and computes $N_7 = N_1' \oplus N_6$, $SK_j = h(SID_j \parallel CID_i \parallel N_6 \parallel N_1')$, $A_5 = h(SK_j \parallel N_6)$.

*Step 4.* The smart card computes $N_6' = N_7 \oplus N_1$, $SK_i = h(SID_j \parallel CID_i \parallel N_6' \parallel N_1)$, $A_5' = h(SK_i \parallel N_6')$. If $A_5' == A_5$, $U_i$ authenticates $S_j$ and accepts $SK_i$ as their session key.

*4.2. Cryptanalysis of Amin's Scheme.* This section will demonstrate that Amin's scheme suffers from insider attack and off-line dictionary attackl furthermore, it fails to achieve forward secrecy and user anonymity.

*4.2.1. Off-Line Dictionary Attack.* If $\mathscr{A}$ steals $U_i$'s smart card and gets $\{CID_i, E_i, b\}$ from the card, then a dictionary attack can be performed as follows:

*Step 1.* Guess $PW_i$ to be $PW_i^*$ and $ID_i$ to be $ID_i^*$.

*Step 2.* Compute $PWR_i^* = h(PW_i \oplus b_i)$.

*Step 3.* Compute $E_i^* = h(ID_i \parallel PW_i \parallel CID_i)$.

*Step 4.* Verify the correctness of $PW_i$ and $ID_i$ by checking if $E_i^* == E_i$.

*Step 5.* Repeat Steps 1~4 until the correct values of $PW_i$ and $ID_i$ are found.

Once the adversary $\mathscr{A}$ gets $PW_i$ and $ID_i$, he/she can impersonate $U_i$. And the time complexity of the attack is $\mathcal{O}(|\mathscr{D}_{pw}| * |\mathscr{D}_{id}| * 2T_H)$, so the attack is efficient.

*4.2.2. User Impersonation Attack.* Suppose $\mathscr{A}$ is also a legitimate server $S_j$; then $S_j$ can impersonate $U_i$ to $RC$ as follows.

*Step 1.* Eavesdrop $\{CID_i, SID_j, T_i, L_3, L_2, N_3\}$ from $U_i$ via the open channel.

*Step 2.* Follow the protocol steps as a legitimate server to gain the response $\{CID_i, A_4, A_3, N_5\}$ from $RC$.

*Step 3.* Continue acting as a legitimate server to compute $N_4 = Skey_j \oplus A_3$, $N_1 = N_4 \oplus N_5$.

*Step 4.* Record $N_1$.

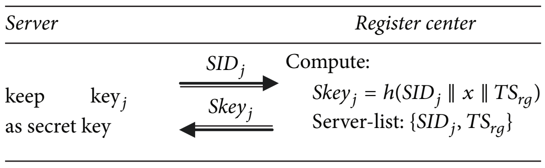

Server
Register center
$SID_j$
Compute:
keep key$_j$ as secret key
$Skey_j$
$Skey_j = h(SID_j \parallel x \parallel TS_{rg})$
Server-list: $\{SID_j, TS_{rg}\}$

Figure 3: Server registration phase.

*Step 5.* Compute $N_2 = N_3 \oplus N_1$, where $N_3$ is from Step 1.

*Step 6.* Compute $PWR_i = L_2 \oplus N_2$.

The above procedure only involves the lightweight XOR operation; thus it is quite efficient. Now, $\mathscr{A}$ (also $S_j$) knows $U_i$'s $PWR_i$; then he/she can forge $U_i$'s request message as $\{CID_i, SID_k, T_i, L_3^a, L_2^a, N_3^a\}$ to other server $S_k$ to enjoy the service. What is more, $\mathscr{A}$ can perform the above attack to all the users who have ever requested to login $S_j$. So such attack is terrible and has a huge effect to the system.

In fact, after recording $N_1$, the adversary $\mathscr{A}$ can directly replay the access request as $\{CID_i, SID_k, T_i, L_3^a, L_2, N_3\}$ to $S_k$, where $L_3^a = h(L_1 \parallel SID_k \parallel N_1 \parallel L_2 \parallel N_3)$; then, with the knowledge of $N_1$, $S_j$ can always compute the correct session key as $N_6' = N_7 \oplus N_1$, $SK_i = h(SID_k \parallel CID_i \parallel N_6' \parallel N_1)$.

*4.2.3. Forward Secrecy.* Assume that $\mathscr{A}$ gets $x$ and eavesdrops $CID_i$, $SID_j$, $T_i$, $L_2$, $N_3$, and $N_7$; then he/she can compute the session key by the following steps:

*Step 1.* Compute $A_i = h(CID_i \parallel x)$.

*Step 2.* Compute $PWR_i = T_i \oplus A_i$.

*Step 3.* Compute $N_2 = L_2 \oplus PWR_i$.

*Step 4.* Compute $N_1 = N_3 \oplus N_2$.

*Step 5.* Compute $N_1 = N_3 \oplus N_2$.

*Step 6.* Compute $N_6 = N_7 \oplus N_1$.

*Step 7.* Compute $SK = h(SID_j \parallel CID_i \parallel N_6 \parallel N_1)$.

Till now, $\mathscr{A}$ gets session key $SK$, and the time complexity of the attack is $2T_H$ which is a very short time.

*4.2.4. User Anonymity.* Similar to Maitra et al.'s scheme [18], this scheme also has the static parameters $T_i$ and $CID_j$ to uniquely identify $U_i$; thus it fails to provide user anonymity.

## 5. Proposed Scheme

To overcome the identified weaknesses, we designed a new enhanced scheme (shown in Figures 3, 4, and 5). For better comprehension, we sketch the ideas behind our scheme:

(i) We adopt a way of "honeywords" + "fuzzy-verifiers" which is introduced by D. Wang and P. Wang [2] to settle the off-line dictionary attack in these two schemes. As we mentioned above, the inherent reason for such attack is the critical parameter $E_i$ which can be used to test the correctness of the guessed $ID_i$ and $PW_i$. However, in the way of "honeywords" + "fuzzy-verifiers", $E_i$ is recalculated as $h(h(DID_i) \parallel h(PWR_i))$ mod $n_0$ where $n_0$ $(2^4 \le n_0 \le 2^8)$ is a integer to determine the size of $(ID, PW)$. Furthermore, there is a *Honey_List* maintained in $RC$ to record the numbers of failed logins. Thus even if $\mathscr{A}$ finds a pair of $\{ID_i', PW_i'\}$ that satisfies the equation, he/she still cannot know whether $ID_i' \overset{?}{=} ID_i$ and $PW_i' \overset{?}{=} PW_i$, for there are $|\mathscr{D}_{pw}| * |\mathscr{D}_{id}| \div n_0 \approx 2^{32}$ candidates of $\{ID_i, PW_i\}$ pair. Then $\mathscr{A}$ has to verify these candidates online, but it is stopped by *Honey_List*.

(ii) We follow the principle in [39] to deploy a public key algorithm to achieve user anonymity. We conceal the identity $ID_i$ in $D_i$, then the adversary cannot get $ID_i$ from $D_i$ unless he/she knows the secret long term key or solves the discrete logarithm program. Furthermore, $D_i$ is changed with the random number $r_i$ to avoid identity being traced.

(iii) From the verifier-stolen attack in Maitra et al.'s scheme, it is important to protect the long term secret key $x$, "XOR" operation on $x$ is a risky behavior which is likely to expose $x$. Thus, in our scheme, $x$ is used in a form of $h(SID_j \parallel x \parallel TS_{rg})$ and $h(x \parallel y_i \parallel T_{rg})$.

(iv) The server $S_j$ in multiserver environment is a special adversary, which should be treated carefully. In Section 4.2.2, we witnessed how $S_j$ carries out an attack. The key to prevent such attack is to let $S_j$ not know the key parameter of $U_i$ or $RC$. So we, on one hand, compute the shared key of $S_j$ and $RC$ as $h(SID_j \parallel x \parallel TS_{rg})$ to make $S_j$ learn nothing about $x$; on the other hand, we use the output of public key algorithm $C_2$ concealed in $P_{rc_{ij}}$ for $U_i$ to authenticate $S_j$ ($S_j$ does not know any key parameter such as $PWR_i$ of $U_i$).

*5.1. Initialization Phase.* $RC$ selects a generator $g$ of a multiplicative group $G$ of prime order $p$ and a secret long key $x$ $(x \in Z_p)$ and then computes the public key $y = g^x$ mod $p$. Then, similar to Maitra et al.'s scheme [18], there is a symmetric key encryption/decryption algorithm and also a hash function $h(\cdot)$: $\{0,1\}^* \rightarrow \{0,1\}^l$.

*5.2. Server Registration Phase*

*Step 1.* $S_j \Rightarrow RC$: $\{SID_j\}$.

*Step 2.* $RC \Rightarrow S_j$: $\{Skey_j\}$. $RC$ researches the *Server-list* to check the valid of $SID_j$. If it is not in it, it computes $Skey_j = h(SID_j \parallel x \parallel TS_{rg})$, where $T_{rg}$ is the register time. Then $RC$ adds $\{SID_j, TS_{rg}\}$ into the *Server-list*, finally sends $\{Skey_j\}$ to $S_j$; otherwise, $RC$ rejects.

*Step 3.* After getting $\{Skey_j\}$, $S_j$ keeps it as its secret key.

| *User* | | *Register center* |
|---|---|---|
| Choose $ID_i$, $\mathrm{PW}_i$, $b_i$<br>$PWR_i = h(PW_i \parallel b_i)$<br>$DID_i = h(ID_i \parallel b_i)$ | $\xrightarrow{(DID_i, PWR_i)}$ | Choose unique random number $y_i$<br>Compute:<br>$K_i = h(x \parallel y_i \parallel T_{rg})$<br>$C_i = K_i \oplus DID_i \oplus PWR_i$<br>$E_i = h(h(DID_i) \oplus h(PWR_i)) \bmod n_0$<br>user-list: $\{DID_i, y_i, T_{rg}, \text{Honey\_List} = 0\}$ |
| Compute $\tilde{b}_i = b_i \oplus h(ID_i \oplus PW_i)$<br>Input $\tilde{b}_i$ into the card | $\xleftarrow[(C_i, E_i, n_0, p, g, y, h(\cdot))]{\text{smart card with}}$ | |

Figure 4: User registration phase.

| *User U/smartcard* | *Server* | *Register center* |
|---|---|---|
| Input$(ID_i, PW_i)$<br>$b_i' = \tilde{b}_i \oplus h(ID_i \parallel PW_i)$<br>$DID_i' = h(ID_i \parallel b_i')$<br>$PWR_i' = h(PW_i \parallel b_i')$<br>$E_i ? = h(h(DID_i') \oplus h(PWR_i')) \bmod n_0$<br>Select a random number $r_i$<br>$C_1 = g^{r_i} \bmod p$<br>$C_2 = y^{r_i} \bmod p$<br>$D_i = DID_i' \oplus h(C_1 \parallel C_2)$<br>$K_i' = C_i \oplus DID_i' \oplus PWR_i'$<br>$F_{ij} = h(DID_i' \parallel C_1 \parallel C_2 \parallel K_i' \parallel SID_j)$<br>$\xrightarrow{\text{Login}_{U_i} = \{D_i, C_1, F_{ij}, SID_j\}}$ | | |
| | Choose a random number $n$<br>$H_{ji} = Enc_{Skey_j}[n \parallel \text{Login}_{U_i} \parallel SID_j]$<br>$\xrightarrow{\{H_{ji}, SID_j\}}$ | |
| | | Verify $\mathrm{SID_j}$<br>Get $TS_{rg}$ from Server-list<br>$Skey_j' = h(SID_j \parallel x \parallel TS_{rg})$<br>$(n^* \parallel \text{Login}^*_{U_i} \parallel SID_j^*) = Dec_{Skey_j'}[H_{ji}]$<br>$\text{Login}^*_{U_i} = \{D_i^*, C_1^*, F_{ij}^*, SID_j^{**}\}$<br>$SID_j ? = SID_j^* ? = SID_j^{**}$<br>$C_2^* = (C_1^*)^x \bmod p$<br>$DID_i^* = D_i^* \oplus h(C_1^* \parallel C_2^*)$<br>Get $T_{rg}$, $y_i$ from User-list<br>$K_i = h(x \parallel y_i \parallel T_{rg})$<br>$F_{ij}^* ? = h(DID_i^* \parallel C_1^* \parallel C_2^* \parallel K_i \parallel SID_j^*)$<br>$P_{rc_{ij}} = h(DID_i^* \parallel C_1^* \parallel C_2^*)$<br>$K_{rc_{ij}} = P_{rc_{ij}} \oplus n^*$<br>$N_{rc_{ij}} = h(n^* \parallel SID_j \parallel P_{rc_{ij}})$<br>$R_{rc_{ij}} = Enc_{h(Skey_j \parallel n^*)}[K_{rc_{ij}} \parallel N_{rc_{ij}}]$<br>$\xleftarrow{R_{rc_{ij}}}$ |
| | $K^\dagger_{rc_{ij}} \parallel N^\dagger_{rc_{ij}} = Dec_{h(Skey_j \parallel n)}[R_{rc_{ij}}]$<br>$P^\dagger_{rc_{ij}} = K^\dagger_{rc_{ij}} \oplus n$<br>$N^\dagger_{rc_{ij}} ? = h(n \parallel SID_j \parallel P^\dagger_{rc_{ij}})$<br>Select a random number $r_j$<br>$C_3 = g^{r_j} \bmod p$<br>$C_4 = (C_1)^{r_j} \bmod p$<br>$SK^\dagger = h(P^\dagger_{rc_{ij}} \parallel C_1 \parallel C_4)$<br>$Q_{ji} = h(P^\dagger_{rc_{ij}} \parallel C_1 \parallel C_3 \parallel C_4 \parallel SK^\dagger)$<br>$\xleftarrow{\{Q_{ji}, C_3\}}$ | |
| $C_4' = (C_3)^{r_i} \bmod p$<br>$P'_{rc_{ij}} = h(DID_i \parallel C_1 \parallel C_2)$<br>$SK = h(P'_{rc_{ij}} \parallel C_1 \parallel C_4')$<br>$Q_{ji} ? = h(P'_{rc_{ij}} \parallel C_1 \parallel C_3 \parallel C_4' \parallel SK)$<br>$V_i = h(SK \parallel C_4' \parallel C_3)$<br>$\xrightarrow{\{V_i\}}$ | | |
| | $V_i^\dagger ? = h(SK^\dagger \parallel C_4 \parallel C_3)$ | |

Figure 5: Login and authentication phase.

*5.3. User Registration Phase*

*Step 1.* $U_i \Rightarrow RC$: $\{DID_i, PWR_i\}$. $U_i$ chooses password $PW_i$, identity $ID_i$, and a random number $b_i$, computes $DID_i = h(ID_i \parallel b_i)$, $PWR_i = h(PW_i \parallel b_i)$, and then sends $\{DID_i, PWR_i\}$ to $RC$.

*Step 2.* $RC \Rightarrow U_i$: a smart card with $\{C_i, E_i, n_0, p, g, y, h(*)\}$. $RC$ tests the valid of $DID_i$ from the *user-list*. If it has been used by other users, it asks $U_i$ for a new identity; otherwise, it chooses a unique random number $y_i$ and calculates $K_i = h(x \parallel y_i \parallel T_{rg})$, $C_i = K_i \oplus DID_i \oplus PWR_i$, $E_i = h(h(DID_i) \parallel h(PWR_i)) \bmod n_0$ where $n_0$ is a integer and $2^4 \leq n_0 \leq 2^8$ and then stores $\{DID_i, y_i, T_{rg}, Honey\_List\}$ into the *User-list*. It should be noted that *Honey_List* is to record the number of login failures and is initialized to 0. At last, $RC$ issues $U_i$ a smart card with $\{C_i, E_i, n_0, p, g, y, h(*)\}$.

*Step 3.* $U_i$ computes $\tilde{b}_i = b_i \oplus h(ID_i \parallel PW_i)$ and enters $\tilde{b}_i$ into the smart card.

*5.4. Login and Authentication Phase*

*Step 1.* $U_i \rightarrow S_j$: $Login_{U_i} = \{D_i, C_1, F_{ij}, SID_j\}$. $U_i$ puts the smart card into a terminal and inputs $ID_i$ and $PW_i$. The card computes $b_i' = \tilde{b}_i \oplus h(ID_i \parallel PW_i)$, $DID_i' = h(ID_i \oplus b_i')$,

$PWR_i' = h(PW_i \parallel b_i')$ and then verifies the legitimacy of $U_i$ by testing $E_i ?= h(h(DID_i') \oplus h(PWR_i')) \bmod n_0$. If it is not equal, exit the session.

Otherwise, the card selects a random number $r_i$, computes $C_1 = g^{r_i} \bmod p$, $C_2 = y^{r_i} \bmod p$, $D_i = DID_i' \oplus h(C_1 \parallel C_2)$, $K_i' = C_i \oplus DID_i' \oplus PWR_i'$, and $F_{ij} = h(DID_i' \parallel C_1 \parallel C_2 \parallel K_i' \parallel SID_j)$, then sends $Login_{U_i} = \{D_i, C_1, F_{ij}, SID_j\}$ to $S_j$.

*Step 2.* $S_j \rightarrow RC$: $\{H_{ji}, SID_j\}$. $S_j$ chooses a random number $n$ as a "challenge", computes: $H_{ji} = Enc_{Skey_j}[n \parallel Login_{U_i} \parallel SID_j]$, and sends $\{H_{ji}, SID_j\}$ to $RC$.

*Step 3.* $RC \rightarrow S_j$: $\{R_{rc_{ij}}\}$. $RC$ first checks the valid of $SID_j$, then gets $TS_{rg}$ from the *Server-List*, and computes: $Skey_j' = h(SID_j \parallel x \parallel TS_{rg})$, $n^* \parallel Login_{U_i}^* \parallel SID_j^* = Dec_{Skey_j'}[H_{ji}]$ where $Login_{U_i}^* = \{D_i^*, C_1^*, F_{ij}^*, SID_j^{**}\}$; then $RC$ tests $SID_j ?= SID_j^* ?= SID_j^{**}$ to verify the legitimacy of $S_j$. If they are not equal, end the session.

Otherwise, $RC$ continues computing $C_2^* = (C_1^*)^x \bmod p$, $DID_i^* = D_i^* \oplus h(C_1^* \parallel C_2^*)$, then acquires $T_{rg}$ and $y_i$ from the *User-list*, computes $K_i = h(x \parallel y_i \parallel T_{rg})$, and checks $F_{ij}^* ?= h(DID_i^* \parallel C_1^* \parallel C_2^* \parallel K_i \parallel SID_j^*)$ to authenticate $U_i$. If $U_i$ is not a valid user, $RC$ sets *Hoeny_List* to be *Hoeny_List* + 1 and ends the session. Once the value of *Honey_List* ≥ the predetermined threshold (such as 10), it is likely that the information in the smart card was exposed; thus $RC$ suspends the card till $U_i$ reregisters.

Otherwise, $RC$ continues computing $P_{rc_{ij}} = h(DID_i^* \parallel C_1^* \parallel C_2^*)$, $K_{rc_{ij}} = P_{rc_{ij}} \oplus n^*$, $N_{rc_{ij}} = h(n^* \parallel SID_j \parallel P_{rc_{ij}})$, $R_{rc_{ij}} = Enc_{h(Skey_j \parallel n^*)}[K_{rc_{ij}} \parallel N_{rc_{ij}}]$, finally responding to $S_j$ with $R_{rc_{ij}}$.

*Step 4.* $S_j \rightarrow U_i$: $\{Q_{ji}, C_3\}$. $S_j$ first decrypts $R_{rc_{ij}}$ with $h(Skey_j \parallel n)$ to obtain $K_{rc_{ij}}^\dagger$ and $N_{rc_{ij}}^\dagger$, computes $P_{rc_{ij}}^\dagger = K_{rc_{ij}}^\dagger \oplus n^*$, and then compares $N_{rc_{ij}}^\dagger$ with $h(n \parallel SID_j \parallel P_{rc_{ij}}^\dagger)$ to authenticate $RC$. If the condition is not satisfied, exit.

Otherwise, $S_j$ selects a random number $r_j$, computes $C_3 = g^{r_j} \bmod p$, $C_4 = (C_1)^{r_j} \bmod p$, $SK^\dagger = h(P_{rc_{ij}}^\dagger \parallel C_1 \parallel C_4)$, $Q_{ji} = h(P_{rc_{ij}}^\dagger \parallel C_1 \parallel C_3 \parallel C_4 \parallel SK^\dagger)$, sends $\{Q_{ji}, C_3\}$ to $U_i$.

*Step 5.* $U_i \rightarrow S_j$: $\{V_i\}$. The smart card computes $C_4 = (C_3)^{r_i} \bmod p$, $P_{rc_{ij}}' = h(DID_i \parallel C_1 \parallel C_2)$, $SK = h(P_{rc_{ij}}' \parallel C_1 \parallel C_4')$. If $Q_{ji} == h(P_{rc_{ij}}' \parallel C_1 \parallel C_3 \parallel C_4' \parallel SK)$, $U_i$ believes that $S_j$ is the desired server and accepts $SK$ as the session key and then sends $V_i = h(SK \parallel C_4' \parallel C_3)$ to $S_j$. Otherwise, exit the session.

*Step 6.* $S_j$ computes $V_i^\dagger = h(SK^\dagger \parallel C_4 \parallel C_3)$. If $V_i^\dagger == V_i$, $S_j$ believes the legitimacy of $U_i$. Till now, the authentication phase finished successfully, and the session key is established.

*5.5. Password Change Phase.* When the user wants to change the password, he can perform the steps as follows.

*Step 1.* $U_i$ inputs $ID_i$, $PW_i$, and new password $PW_i^{new}$.

*Step 2.* The card computes $b_i' = \tilde{b}_i \oplus h(ID_i \parallel PW_i)$, $DID_i' = h(ID_i \oplus b_i')$, $PWR_i' = h(PW_i \parallel b_i')$, if $E_i \neq h(h(DID_i') \oplus h(PWR_i')) \bmod n_0$, the card rejects the request. Otherwise, it computes $C_i^{new} = C_i \oplus PWR_i' \oplus PWR_i^{new}$, $E_i^{new} = h(h(DID_i') \parallel h(PWR_i^{new})) \bmod n_0$ and replaces $C_i, E_i$ with $C_i^{new}, E_i^{new}$.

*5.6. Revocation Phase.* Once the user realized the card is not in the control of himself, he can revoke the account as follows.

*Step 1.* $U_i$ firstly gets authenticated by the card in the same way as in Step 1 in Section 5.4.

*Step 2.* $U_i \rightarrow RC$: $\{D_i, C_1, F_{ij}, revoke\_request\}$. The way to compute $D_i$, $C_1$, and $F_{ij}$ is similar to Step 1 in Section 5.4, expect $F_{ij} = h(DID_i' \parallel C_1 \parallel C_2 \parallel K_i')$.

*Step 3.* $RC$ authenticates $U_i$ by computing $C_2^* = (C_1^*)^x \bmod p$, $DID_i^* = D_i^* \oplus h(C_1^* \parallel C_2^*)$, $K_i = h(x \parallel y_i \parallel T_{rg})$, $F_{ij}^* ?= h(DID_i^* \parallel C_1^* \parallel C_2^* \parallel K_i \parallel)$. If $RC$ accepts $U_i$, it sets $y_i = NULL$ to revoke the account. Otherwise, $RC$ reject the request.

*5.7. Reregistration Phase.* If $U_i$ with correct password and identity is still rejected by $S_j$, then he can reregister as follows.

*Step 1.* $U_i \Rightarrow RC$: $\{DID_i, PWR_i, reregister\}$.

*Step 2.* $RC$ first researches $DID_i$ in the *User-list* and checks whether the account of $U_i$ is revoked or the card is suspended. If so, $RC$ accepts the request and conducts the register phase in Section 5.3.

## 6. Security Analysis

In this section, we first use the Burrows-Abadi-Needham (BAN) logic [40] to prove the security of our scheme formally, then analyze it in a heuristic method. The results demonstrate the security and practicability of our scheme.

*6.1. Formal Analysis Based on BAN Logic.* As an efficient and simple way to analyze the design logic and security of the authentication scheme, BAN logic [40] has been widely used. As shown in Table 4, it uses some particular notions to depict a protocol.

The goals of our proposed scheme are as follows: these four goals ensure that the server and the user get authenticated mutually (corresponding to our proposed S1, S10, and S13), and they build a session key successfully (corresponding to our proposed S14):

(1) Goal 1: $U_i \mid\equiv S_j \mid\equiv (U_i \overset{SK}{\longleftrightarrow} S_j)$.

(2) Goal 2: $U_i \mid\equiv (U_i \overset{SK}{\longleftrightarrow} S_j)$.

(3) Goal 3: $S_j \mid\equiv U_i \mid\equiv (U_i \overset{SK}{\longleftrightarrow} S_j)$.

(4) Goal 4: $S_j \mid\equiv (U_i \overset{SK}{\longleftrightarrow} S_j)$.

TABLE 4: Notations in BAN logic.

| | |
|---|---|
| $P \mid\equiv X$ | $P$ believes $X$, that is, the principal $P$ believes the statement $X$ is true. |
| $P \triangleleft X$ | $P$ sees $X$, that is, the principal $P$ receives a message that contains $X$. |
| $P \mid\Rightarrow X$ | $P$ has jurisdiction over $X$, that is, the principal $P$ can generates or computes $X$. |
| $P \mid\sim X$ | $P$ said $X$, that is, the principal $P$ has sent a message containing $X$. |
| $\sharp(X)$ | $X$ is fresh, that is, $X$ is sent in a message only at the current run of the protocol, it is usually a timestamp or a random number. |
| $P \overset{K}{\longleftrightarrow} Q$ | $K$ is the shared key for $P$ and $Q$. |
| $P \overset{Y}{\rightleftharpoons} Q$ | $Y$ is the secret known only to $P$ and $Q$ or some principals trusted by them. |
| $\langle X\rangle_Y$ | $X$ combined with $Y$, and $Y$ usually is a secret. |
| $\{X\}_K$ | $X$ encrypted with $K$. |
| $\dfrac{P \mid\equiv P \overset{K}{\longleftrightarrow} Q, P \triangleleft \{X\}_K}{P \mid\equiv Q \mid\sim X}$ or $\dfrac{P \mid\equiv P \overset{Y}{\rightleftharpoons} Q, P \triangleleft \langle X\rangle_Y}{P \mid\equiv Q \mid\sim X}$ | *RULE*(1): the message-meaning rule. This rule will be used in the proving process. |
| $\dfrac{P \mid\equiv \sharp(X), P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X}$ | *RULE*(2): the nonce-verification rule. This rule will be used in the proving process. |
| $\dfrac{P \mid\equiv Q \mid\Rightarrow X, P \mid\equiv Q \mid\equiv X}{P \mid\equiv X}$ | *RULE*(3): the jurisdiction rule. This rule will be used in the proving process. |
| $\dfrac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X,Y)}$ | *RULE*(4): the freshness-conjuncatenation rule. This rule will be used in the proving process. |

According to the BAN logic, we first transform the scheme to an idealized one:

$$M_1: U_i \to S_j: \langle DID_i, SID_j, C_1, U_i \overset{C_2}{\longleftrightarrow} RC\rangle_{U_i \overset{K_i}{\longleftrightarrow} RC}.$$

$$M_2: S_j \to RC: \langle SID_j, D_i, C_1, F_{ij}, n\rangle_{S_j \overset{Skey_j}{\longleftrightarrow} RC}.$$

$$M_3: RC \to S_j: \langle SID_j, n, U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j\rangle_{S_j \overset{Skey_j}{\longleftrightarrow} RC}.$$

$$M_4: S_j \to U_i: \langle C_1, C_3, U_i \overset{SK}{\longleftrightarrow} S_j\rangle_{U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j}.$$

$$M_5: U_i \to S_j: \langle C_3, U_i \overset{SK}{\longleftrightarrow} S_j\rangle_{U_i \overset{C_4}{\longleftrightarrow} S_j}.$$

Then, to analyze the scheme, we make some assumptions about its initial state as follows:

$H_1$: $U_i \mid\equiv \sharp(C_1)$.

$H_2$: $S_j \mid\equiv \sharp(C_3)$.

$H_3$: $S_j \mid\equiv \sharp(n)$.

$H_4$: $U_i \mid\equiv U_i \overset{K_i}{\longleftrightarrow} RC$.

$H_5$: $RC \mid\equiv U_i \overset{K_i}{\longleftrightarrow} RC$.

$H_6$: $U_i \mid\equiv U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j$.

$H_7$: $S_j \mid\equiv S_j \overset{Skey_j}{\longleftrightarrow} RC$.

$H_8$: $RC \mid\equiv S_j \overset{Skey_j}{\longleftrightarrow} RC$.

$H_9$: $S_j \mid\equiv U_i \overset{C_4}{\longleftrightarrow} RC$.

$H_{10}$: $U_i \mid\equiv S_j \mid\Rightarrow U_i \overset{K_i}{\longleftrightarrow} S_j$.

$H_{11}$: $S_j \mid\equiv U_i \mid\Rightarrow U_i \overset{K_i}{\longleftrightarrow} S_j$.

$H_{12}$: $S_j \mid\equiv RC \mid\Rightarrow U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j$.

Based on these assumptions above, we will prove the security of our protocol according to BAN logic as follows.

*From* $M_2$, we have

$$S_1: RC \triangleleft \left\langle SID_j, D_i, C_1, F_{ij}, n\right\rangle_{S_j \overset{Skey_j}{\longleftrightarrow} RC}. \quad (2)$$

Then according to H8, $S_1$, *RULE*(1), it is obvious that

$$S_2: RC \mid\equiv S_j \mid\sim \left\langle SID_j, D_i, C_1, F_{ij}, n\right\rangle. \quad (3)$$

*From* $M_1$, we have

$$S_3: RC \triangleleft \left\langle DID_i, SID_j, C_1, U_i \overset{C_2}{\longleftrightarrow} RC\right\rangle_{U_i \overset{K_i}{\longleftrightarrow} RC}. \quad (4)$$

Then according to $H_5$, $S_3$, *RULE*(1), it is obvious that

$$S_4: RC \mid\equiv U_i \mid\sim \left\langle DID_i, SID_j, C_1, U_i \overset{C_2}{\longleftrightarrow} RC\right\rangle. \quad (5)$$

*From* $M_3$, we have

$$S_5: S_j \triangleleft \left\langle SID_j, n, U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j\right\rangle_{S_j \overset{Skey_j}{\longleftrightarrow} RC}. \quad (6)$$

Then according to $H_7$, $S_5$, $RULE(1)$, it is obvious that

$$S_6\colon S_j \mid\equiv RC \mid\sim \left\langle SID_j, n, U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j \right\rangle. \tag{7}$$

And according to $H_3$, $S_6$, $RULE(4)$ and $RULE(2)$, we get

$$S_7\colon S_j \mid\equiv RC \mid\sim U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j. \tag{8}$$

And according to $H_{12}$, $S_7$, $RULE(3)$, we can get

$$S_8\colon S_j \mid\equiv U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j. \tag{9}$$

*From* $M_4$, we have

$$S_9\colon U_i \triangleleft \left\langle C_1, C_3, U_i \overset{SK}{\longleftrightarrow} S_j \right\rangle_{U_i \overset{P_{rc_{ij}}}{\rightleftharpoons} S_j}. \tag{10}$$

Then according to $H_6$, $S_9$, $RULE(1)$, it is obvious that

$$S_{10}\colon U_i \mid\equiv S_j \mid\sim \left\langle C_1, C_3, U_i \overset{SK}{\longleftrightarrow} S_j \right\rangle. \tag{11}$$

And according to $H_2$, $S_{10}$, $RULE(4)$ and $RULE(2)$, we get

$$S_{11}\colon U_i \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j \quad (\textit{Goal 1}). \tag{12}$$

And according to $H_{10}$, $S_{11}$, $RULE(3)$, we can get

$$S_{12}\colon U_i \mid\equiv \left(U_i \overset{SK}{\longleftrightarrow} S_j\right) \quad (\textit{Goal 2}). \tag{13}$$

*From* $M_5$, we have

$$S_{13}\colon S_j \triangleleft \left\langle C_3, U_i \overset{SK}{\longleftrightarrow} S_j \right\rangle_{U_i \overset{C_4}{\longleftrightarrow} S_j}. \tag{14}$$

Then according to $H_9$, $S_{13}$, $RULE(1)$, it is obvious that

$$S_{14}\colon S_j \mid\equiv U_i \mid\sim \left\langle C_3, U_i \overset{SK}{\longleftrightarrow} S_j \right\rangle. \tag{15}$$

And according to $H_2$, $S_{13}$, $RULE(4)$ and $RULE(2)$, we get

$$S_{15}\colon S_j \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j \quad (\textit{Goal 3}). \tag{16}$$

And according to $H_{11}$, $S_{15}$, $RULE(3)$, we can get

$$S_{16}\colon S_j \mid\equiv \left(U_i \overset{SK}{\longleftrightarrow} S_j\right) \quad (\textit{Goal 4}). \tag{17}$$

Thus with Goals 1~4, we proved that the user $U_i$ and the server $S_j$ have authenticated to each other; furthermore they accepted and shared the session key $SK$.

*6.2. Informal Analysis.* The heuristic method without complex formula is a direct and simple way for a quick analysis of the security of the protocol. It plays a significant role in cryptoanalysis of authentication protocols, though its analytic process heavily depends on human experience rather than a set of scientific tools. This section uses a heuristic method to prove that our scheme not only provides desire attributes but also is resistant to various attacks.

*6.2.1. User Anonymity.* In our scheme, on one hand, the adversary $\mathscr{A}$ cannot get $ID_i$: the user identity was concealed in $DID_i$ where $DID_i = h(ID_i \parallel b_i)$, so an adversary without $b_i$ cannot guess the value of $ID_i$ via dictionary attack; on the other hand, $\mathscr{A}$ cannot link the message flows to a certain user: though $DID_i$ is a fixed value, it is transmitted in a form of $D_i = DID_i \oplus h(C_1 \parallel C_2)$, where $C_1$ and $C_2$ are changed with different turns of the protocol, and only the user and the one knowing the long term secret key can compute $C_2$. This indicates that $D_i$ is changed with $C_1$ and $C_2$, and $\mathscr{A}$ cannot get $C_2$. Thus $\mathscr{A}$ fails to link the message flows to a certain user. So our scheme achieves user anonymity.

*6.2.2. Forward Secrecy.* The session key $SK$ of the proposed scheme consists of two "special" parameters: $P_{rc_{ij}}$ and $C_4$, where $P_{rc_{ij}} = h(DID_i \parallel C_1 \parallel C_2)$ and $C_4 = (C_3)^{r_i} \bmod p = (C_1)^{r_j} \bmod p = g^{r_i r_j} \bmod p$. Suppose that an adversary gets the secret key $x$, then he can intercept $D_i$ and $C_1$ to compute $DID_i$ and $C_2$. However, computing $C_4$ for $\mathscr{A}$ is equivalent to solving the DLP problem, which is bound to fail. So our scheme provides perfect forward security.

*6.2.3. Mutual Authentication.* In Step 3 of Section 5.4, $RC$ with $x$ verifies the validity of $U_i$ by checking $F_{ij}$, if $U_i$ is legitimate, $RC$ computes $P_{rc_{ij}}$ and constructs $R_{rc_{ij}}$ to $S_j$. So with the help of $RC$, $S_j$ authenticates $U_i$. In a short, both $RC$ and $S_j$ authenticate $U_i$.

In Step 3 of Section 5.4, $RC$ with $Skey_j$ authenticates $S_j$ by checking $SID_j$. If $S_j$ passes the test, $RC$ respond to $S_j$'s challenge $n$ with $R_{rc_{ij}}$ which is an encryption with the key $h(Skey_j \parallel n)$. Then $S_j$ verifies $RC$'s validity by checking $N_{rc_{ij}}$. Furthermore, with $P_{rc_{ij}}$ in $Q_{ji}$, $U_i$ verifies $RC$ and $S_j$. So all in all, $S_j$ authenticates $RC$ and $U_i$; $U_i$ authenticates $RC$ and $S_j$.

Therefore, our scheme achieves mutual authentication.

*6.2.4. Privileged Insider Attack.* In the registration phase of our scheme, $U_i$ submits $\{h(ID_i \parallel b_i), h(PW_i \parallel b_i)\}$ to $RC$. From this message, $RC$ learns nothing about $U_i$'s $PW_i$ or other useful information. Thus our scheme is resistant to privileged insider attack.

*6.2.5. Off-Line Dictionary Attack.* Suppose an adversary $\mathscr{A}$ has the full control of the open channel and obtains the information in the smart card; then we prove that our scheme can resist the off-line dictionary attack through two aspects.

On one hand, with $E_i$ and $\widetilde{b}_i$, $\mathscr{A}$ guesses $ID_i$ and $PW_i$ to be $ID_i'$ and $PW_i'$, respectively. Then $\mathscr{A}$ computes $b_i' = \widetilde{b}_i \oplus h(ID_i' \parallel PW_i')$, $DID_i' = h(ID_i' \oplus b_i')$, $PWR_i' = h(PW_i' \parallel b_i')$, then verifies the correctness of $ID_i'$ and $PW_i'$ by testing $E_i ?= h(h(DID_i') \oplus h(PWR_i')) \bmod n_0$, and repeats these processes till the equation is satisfied. However, even if $\mathscr{A}$ finds such a pair of $ID_i', PW_i'$, he still is not sure whether $ID_i' == ID_i$ and $PW_i' == PW_i$, for there are $|\mathscr{D}_{pw}| * |\mathscr{D}_{id}| \backslash n_0 \approx 2^{32}$ candidates of $\{ID_i, PW_i\}$ pair when $n_0 = 2^8$ and $|\mathscr{D}_{pw}| = |\mathscr{D}_{id}| = 2^6$ [2]. Then $\mathscr{A}$ has to verify $\{ID_i', PW_i'\}$ in a manner of online, which will be stopped by the *Honey_List*.

Table 5: Performance comparison among relevant schemes in multiserver.

| | Computation overhead | | Communication cost | | The proposed fourteen evaluation criteria | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Login | Authentication | Login | Authentication | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 | S10 | S11 | S12 | S13 | S14 |
| Leu and Hsieh (2009) [19] | $8T_H$ | $10T_H$ | 512 bits | 384 bits | √ | × | √ | × | √ | × | □ | √ | × | × | √ | √ | × | √ |
| Li et al. (2012) [20] | $T_E$+$5T_H$ | $3T_E$+$9T_H$ | 1280 bits | 1280 bits | √ | × | √ | √ | √ | × | □ | √ | × | √ | √ | √ | × | √ |
| Amin (2012) [17] | $3T_H$ | $9T_H$ | 768 bits | 896 bits | √ | × | √ | × | √ | × | √ | × | × | √ | √ | √ | × | √ |
| Maitra et al. (2016) [18] | $6T_H$ | $5T_S$+$14T_H$ | 640 bits | 1280 bits | √ | × | √ | × | √ | × | × | × | × | √ | √ | × | √ | √ |
| Kumari et al. (2017) [41] | $2T_M$+$3T_H$ | $6T_M$+$15T_H$ | 2176 bits | 9088 bits | √ | √ | √ | √ | √ | × | √ | √ | × | √ | √ | √ | √ | √ |
| Irshad et al. (2017) [42] | $3T_M$+$T_S$+$5T_H$ | $10T_M$+$T_S$+$10T_H$ | 2176 bits | 4480 bits | √ | √ | √ | √ | √ | × | □ | √ | × | √ | √ | √ | × | √ |
| Irshad et al. (2017) [43] | $8T_H$ | $2T_S$+$10T_H$ | 384 bits | 512 bits | √ | × | √ | × | √ | × | □ | × | × | √ | √ | √ | × | √ |
| Our scheme | $2T_E$+$8T_H$ | $4T_E$+$4T_S$+$16T_H$ | 1408 bits | 1664 bits | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |

$T_E$ is the time of modular exponentiation operation, $T_M$ is the time of scalar multiplication on elliptic curve, $T_H$ is the time of hash computation, and $T_S$ is the time of symmetric encryption/decryption; $T_E \gg T_M \gg T_H > T_S$ and the lightweight operation such as "XOR" and "||" can be ignored [2]. Let $n_0$ be 32-bit long; let $ID_i$, $PW_i$, $h(*)$, output of symmetric encryption, timestamp, and random numbers be 128-bit long; let $p$, $g$, $y$ be 1024-bit long [2]. √ means the property is satisfied; × means the property is not satisfied; □ means the property is not related to the scheme.

On the other hand, $\mathscr{A}$ may try another way to conduct an off-line dictionary attack: obtaining $\widetilde{b_i}$, $C_i$ and $\{D_i, C_1, F_{ij}, SID_j\}$, using $F_{ij}$ to check the correctness of the guessed $ID_i'$ and $PW_i'$, while $\mathscr{A}$ has to compute $C_2$ which is impossible for the entity (except $U_i$) without $x$ as we explained in Section 6.2.1.

In conclusion, our scheme is secure to dictionary attack.

*6.2.6. Verifier-Stolen Attack.* In our scheme, $RC$ maintains a *User-list* in form of $\{DID_i, y_i, T_{rg}, Hoeny\_List\}$ and a *Server-list* in form of $\{SID_j, TS_{rg}\}$, while the parameters in the two list are not security-related. Thus an adversary with the verifier table has no security threat to the system.

*6.2.7. Replay Attack.* We apply the random number to prevent replay attack. Suppose an adversary $\mathscr{A}$ eavesdrops the message in the open channel, such as $\{D_i, C_1, F_{ij}, SID_j\}$; then $\mathscr{A}$ replays the message flow to $S_j$. While without $r_i$, $\mathscr{A}$ cannot construct a correct $V_i$ to pass the verification of $S_j$. So $\mathscr{A}$ can neither gain any benefits from replaying the message nor be authenticated by $S_j$. Similarly, $\mathscr{A}$ also fails to carry out a replay attack on other message flows. Therefore, our scheme can resist replay attack.

*6.2.8. User Impersonation Attack.* According to the analysis above, the adversary $\mathscr{A}$ can neither guess $ID_i$ and $PW_i$ nor replay $\{D_i, C_1, F_{ij}, SID_j\}$ to impersonate $U_i$, so there is only one way left: constructing $\{D_a, C_{1a}, F_a, SID_j\}$. So $\mathscr{A}$ selects a random number $r_a$ and computes $\{C_{1a}, C_{2a}\}$, forges $DID_a$ and $K_a$, then computes $\{D_a, C_{1a}, F_a, SID_j\}$, and sends it to $RC$. However, after $RC$ computes $C_{2a}$ and obtains $DID_a$, $RC$ either cannot find such a $DID_a$ in *User-list* or computes $K_a'$ that is not equal to $K_a$. Both two conditions lead to the failure in authentication of $U_i$. That means that $RC$ is bound to find that $\{D_a, C_{1a}, F_a, SID_j\}$ is forged. Therefore, $\mathscr{A}$ cannot impersonate $U_i$.

*6.2.9. Server Impersonation Attack.* On one hand, according to the analysis above, the adversary $\mathscr{A}$ cannot replay the message flows to impersonate $S_j$; on the other hand, $\mathscr{A}$ finds no way to get the private key $Skey_j$. Thus our scheme is secure from server impersonation attack.

## 7. Performance Analysis

In this section, we compared our scheme with other two-factor authentication schemes for multiserver environment [17–20, 41–43]. As shown in Table 5, the result manifests the advantages of the proposed scheme in security attributes. We can see that our scheme satisfied all the security attributes, and it is the best one among these schemes, though its computation overhead and communication cost are higher, while others [17–20, 41–43] have weaknesses more or less. Actually, according to Wang et al. [28, 32], the public key algorithm is the key to achieve user anonymity and resistance against off-line dictionary attack, while the public key algorithm is bound to cost more than symmetric algorithm. So these schemes [17–19, 43] only using the symmetric algorithm need less communication cost than our scheme, but they are definitely not secure. Among the compared schemes, only these schemes [20, 41, 42] are equipped with the public key algorithm. Both the schemes of Kumari et al. [41] and Irshad et al. [42] cost more than our scheme. And our scheme does not spend much more communication cost or computation overhead than Lix et al.'s, while achieving all the fourteen evaluation criteria (Lix et al.'s scheme only nine). As a matter of fact, certain communication cost is a must for achieving better security. We think that ensuring the security of the protocol is the most important goal for an authentication scheme; furthermore, our scheme actually does not significantly increase the computation overhead and communication cost. Therefore, compared to those schemes vulnerable to attacks, our scheme is more suitable to multiserver environment.

## 8. Conclusion

In this paper, firstly, we described the communication model and adversary model of multiserver environment, pointing out that some of the adversary capacities in many schemes are impractical and unreasonable. Then based on the works of pioneer contributors, we summarized fourteen security requirements for user authentication in multiserver environment. Secondly, according to the adversary model and security requirements, we demonstrated the weakness in the scheme of Amin and Maitra et al. Thirdly, to overcome the identified weaknesses, we proposed a new improved scheme for multiserver environment and proved its security via BAN logic and heuristic analysis. Furthermore, the comparison results showed the superiority of our scheme.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This research are supported by the National Key Research and Development Program of China (no. 2017YFB0801900); the BUPT Excellent Ph.D. Students Foundation; the National High Technology Research and Development Program Foundation of China under Grant no. 2015AA017202; the Guangdong Provincial Science and Technology Department Frontier and Key Technology Innovation Project Foundation under Grant no. 2016B010110002; the State Grid Corporation of China Key Technology Innovation Project Foundation under Grant no. SGRIXTKJ [2017] 265.

## References

[1] X. Huang, Y. Xiang, A. Chonka, J. Zhou, and R. H. Deng, "A generic framework for three-factor authentication: Preserving security and privacy in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1390–1397, 2011.

[2] D. Wang and P. Wang, "Two Birds with One Stone: Two-Factor Authentication with Security Beyond Conventional Bound," *IEEE Transactions on Dependable and Secure Computing*, 2016.

[3] S. Kumari, M. K. Khan, X. Li, and F. Wu, "Design of a user anonymous password authentication scheme without smart card," *International Journal of Communication Systems*, vol. 29, no. 3, pp. 441–458, 2016.

[4] C. Wang, D. Wang, G. Xu, and Y. Guo, "A lightweight password-based authentication protocol using smart card," *International Journal of Communication Systems*, vol. 30, no. 16, Article ID e3336, 2017.

[5] A. K. Das, "Analysis and improvement on an efficient biometric-based remote user authentication scheme using smart cards," *IET Information Security*, vol. 5, no. 3, pp. 145–151, 2011.

[6] L. Li, I. Lin, and M. Hwang, "A remote password authentication scheme for multiserver architecture using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 12, no. 6, pp. 1498–1504, 2001.

[7] I. C. Lin, M. S. Hwang, and L. H. Li, "A new remote user authentication scheme for multi-server architecture," *Future Generation Computer Systems*, vol. 19, no. 1, pp. 13–22, 2003.

[8] W. S. Juang, "Efficient multi-server password authenticated key agreement using smart cards," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 251–255, 2004.

[9] C.-C. Chang and J.-S. Lee, "An efficient and secure multi-server password authentication scheme using smart cards," in *Proceedings of the Proceedings - 2004 International Conference on Cyberworlds, CW 2004*, pp. 417–422, jpn, November 2004.

[10] W. Tsaur, C. Wu, and W. Lee, "A smart card-based remote scheme for password authentication in multi-server internet services," *Computer Standards & Interfaces*, vol. 27, no. 1, pp. 39–51, 2004.

[11] J.-L. Tsai, "Efficient multi-server authentication scheme based on one-way hash function without verification table," *Computers & Security*, vol. 27, no. 3-4, pp. 115–121, 2008.

[12] Y. Liao and S. Wang, "A secure dynamic id based remote user authentication scheme for multi-server environment," *Computer Standards & Interfaces*, vol. 31, no. 1, pp. 24–29, 2009.

[13] H. Hsiang and W. Shih, "Improvement of the secure dynamic id based remote user authentication scheme for multi-server environment," *Computer Standards & Interfaces*, vol. 31, no. 6, pp. 1118–1123, 2009.

[14] S. K. Sood, A. K. Sarje, and K. Singh, "A secure dynamic identity based authentication protocol for multi-server architecture," *Journal of Network and Computer Applications*, vol. 34, no. 2, pp. 609–618, 2011.

[15] C. Li, C. Weng, and C. Fan, "Two-factor user authentication in multi-server networks," *International Journal of Security and its Applications*, vol. 6, pp. 261–267, 2012.

[16] S. Sood, "Dynamic idengtity based authentication protocol for two-sever architecture," *Journal of Information Security*, vol. 3, no. 4, pp. 326–334, 2012.

[17] R. Amin, "Cryptanalysis and efficient dynamic ID based remote user authentication scheme in multi-server environment using smart card," *International Journal of Network Security*, vol. 18, no. 1, pp. 172–181, 2016.

[18] T. Maitra, S. H. Islam, R. Amin, D. Giri, M. K. Khan, and N. Kumar, "An enhanced multi-server authentication protocol using password and smart-card: cryptanalysis and design," *Security and Communication Networks*, vol. 9, no. 17, pp. 4615–4638, 2016.

[19] J.-S. Leu and W.-B. Hsieh, "Efficient and secure dynamic ID-based remote user authentication scheme for distributed systems using smart cards," *IET Information Security*, vol. 8, no. 2, pp. 104–113, 2014.

[20] X. Li, J. Niu, S. Kumari, J. Liao, and W. Liang, "An Enhancement of a Smart Card Authentication Scheme for Multi-server Architecture," *Wireless Personal Communications*, vol. 80, no. 1, pp. 175–192, 2015.

[21] D. Mishra, "Design and Analysis of a Provably Secure Multi-server Authentication Scheme," *Wireless Personal Communications*, vol. 86, no. 3, pp. 1095–1119, 2016.

[22] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipfs law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017.

[23] D. Wang, Q. Gu, H. Cheng, and P. Wang, "The request for better measurement: A comparative evaluation of two-factor authentication schemes," in *Proceedings of 11th ACM Asia Conference on Computer and Communications Security, ASIA CCS*, pp. 475–486, China, June 2016.

[24] Q. Jiang, Z. Chen, B. Li, J. Shen, L. Yang, and J. Ma, "Security analysis and improvement of bio-hashing based three-factor authentication scheme for telecare medical information systems," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2017.

[25] S. H. Islam, "Design and analysis of an improved smartcard-based remote user password authentication scheme," *International Journal of Communication Systems*, vol. 29, no. 11, pp. 1708–1719, 2016.

[26] C. Wang and G. Xu, “Cryptanalysis of Three Password-Based Remote User Authentication Schemes with Non-Tamper-Resistant Smart Card,” *Security and Communication Networks*, vol. 2017, pp. 1–14, 2017.

[27] L. Xiong, D. Peng, T. Peng, H. Liang, and Z. Liu, “A Lightweight Anonymous Authentication Protocol with Perfect Forward Secrecy for Wireless Sensor Networks,” *Sensors*, vol. 17, no. 11, p. 2681, 2017.

[28] D. Wang, D. He, P. Wang, and C.-H. Chu, “Anonymous Two-Factor Authentication in Distributed Systems: Certain Goals Are Beyond Attainment,” *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 4, pp. 428–442, 2015.

[29] Q. Xie, N. Dong, D. S. Wong, and B. Hu, “Cryptanalysis and security enhancement of a robust two-factor authentication and key agreement protocol,” *International Journal of Communication Systems*, vol. 29, no. 3, pp. 478–487, 2016.

[30] J. Wei, W. Liu, and X. Hu, “Secure and efficient smart card based remote user password authentication scheme,” *International Journal of Network Security*, vol. 18, no. 4, pp. 782–792, 2016.

[31] F. Wu, L. Xu, S. Kumari, X. Li, and A. Alelaiwi, “A new authenticated key agreement scheme based on smart cards providing user anonymity with formal proof,” *Security and Communication Networks*, vol. 8, no. 18, pp. 3847–3863, 2015.

[32] C.-G. Ma, D. Wang, and S.-D. Zhao, “Security flaws in two improved remote user authentication schemes using smart cards,” *International Journal of Communication Systems*, vol. 27, no. 10, pp. 2215–2227, 2014.

[33] X. Huang, Y. Xiang, E. Bertino, J. Zhou, and L. Xu, “Robust multi-factor authentication for fragile communications,” *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 6, pp. 568–581, 2014.

[34] Q. Jiang, S. Zeadally, J. Ma, and D. He, “Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks,” *IEEE Access*, vol. 5, pp. 3376–3392, 2017.

[35] X. Li, J. Niu, S. Kumari, J. Liao, W. Liang, and M. K. Khan, “A new authentication protocol for healthcare applications using wireless medical sensor networks with user anonymity,” *Security and Communication Networks*, vol. 9, no. 15, pp. 2643–2655, 2016.

[36] Q. Jiang, J. Ma, C. Yang, X. Ma, J. Shen, and S. A. Chaudhry, “Efficient end-to-end authentication protocol for wearable health monitoring systems,” *Computers and Electrical Engineering*, 2017.

[37] D. He, S. Zeadally, N. Kumar, and W. Wu, “Efficient and Anonymous Mobile User Authentication Protocol Using Self-Certified Public Key Cryptography for Multi-Server Architectures,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 2052–2064, 2016.

[38] V. Odelu, A. K. Das, and A. Goswami, “A Secure Biometrics-Based Multi-Server Authentication Protocol Using Smart Cards,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1953–1966, 2015.

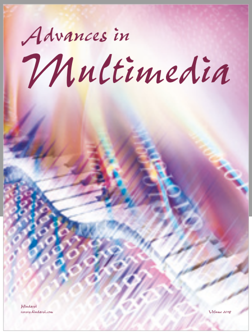
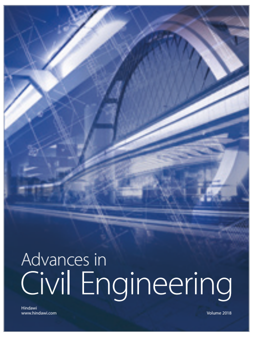
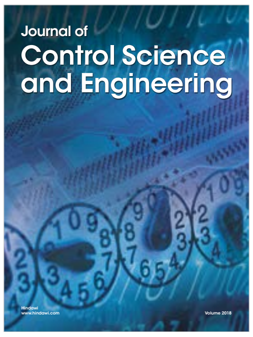
[39] D. Wang and P. Wang, “On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions,” *Computer Networks*, vol. 73, pp. 41–57, 2014.

[40] M. Burrows, M. Abadi, and R. Needham, “Logic of authentication,” *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.

[41] S. Kumari, X. Li, F. Wu, A. K. Das, K.-K. R. Choo, and J. Shen, “Design of a provably secure biometrics-based multi-cloud-server authentication scheme,” *Future Generation Computer Systems*, vol. 68, pp. 320–330, 2017.

[42] A. Irshad, M. Sher, S. A. Chaudhry et al., “A secure mutual authenticated key agreement of user with multiple servers for critical systems,” *Multimedia Tools and Applications*, 2017.

[43] A. Irshad, S. A. Chaudhry, S. Kumari, M. Usman, K. Mahmood, and M. S. Faisal, “An improved lightweight multiserver authentication scheme,” *International Journal of Communication Systems*, 2017.

Hindawi

Submit your manuscripts at
www.hindawi.com