Research Article

A Secure Three-Factor Multiserver Authentication Protocol against the Honest-But-Curious Servers

Hua Guo,1,2 Chen Chen,3 Ya Gao,4 Xiong Li,5,6 and Jiongchao Jin7

1School of Cyber Science and Technology, Beihang University, Beijing 100191, China
2Hefei Innovation Institute, Beihang University, Anhui 230012, China
3Informatization Office of Beihang University, Beijing 100191, China
4Beijing Key Laboratory of Network Technology, Beihang University, Beijing 100191, China
5School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
6Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
7School of Computer Science and Engineering, Beihang University, Beijing 100191, China

Correspondence should be addressed to Xiong Li; lixiongzhq@163.com

Received 13 April 2018; Revised 26 July 2018; Accepted 26 August 2018; Published 13 September 2018

Copyright © 2018 Hua Guo 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.

Three-factor multiserver authentication protocols become a prevalence in recent years. Among these protocols, almost all of them do not involve the registration center into the authentication process. To improve the protocol’s efficiency, a common secret key is shared among all servers, which leads to a serious weakness; i.e., we find that these protocols cannot resist the passive attack from the honest-but-curious servers. This paper takes Wang et al.’s protocol as an example, to exhibit how an honest-but-curious server attacks their protocol. To remedy this weakness, a novel three-factor multiserver authentication protocol is presented. By introducing the registration center into the authentication process, the new protocol can resist the passive attack from the honest-but-curious servers. Security analyses including formal and informal analyses are given, demonstrating the correctness and validity of the new protocol. Compared with related protocols, the new protocol possesses more secure properties and more practical functionalities than others at a relatively low computation cost and communication cost.

1. Introduction

Nowadays, with the rapid development of networks, remote communication becomes increasingly prevalent and provides highly useful services in many aspects. Consequently, communication security significantly attracts public’s attention. Cryptographic authentication allows users to submit their credentials and acquire authorization to access the various online services from remote networks [1–5]. Since Lamport [6] firstly proposed a password-based remote authentication protocol, great quantities of authentication protocols were proposed to make up continued emerging problems and provide authorized communication between remote entities. However, the traditional protocols gradually cannot catch up with the pace of increasing demand for more users and servers in communication. Multiserver authentication schemes became the mainstream, because most of the practical communication environments are based on several servers to alleviate the pressure of the increasing number of users.

Lots of authentication protocols for multiserver environments were proposed to satisfy the security requirements and provide versatile functionalities to make the scheme more convenient and practical to utilize in real occasions [7–20]. In 2001, Li et al. [7] proposed a remote multiserver authentication protocol with no verification table, which was found insecure by Lin et al. [8]. They also presented an improved protocol, while it was vulnerable to impersonation attack [9]. Juang et al. [10] adopted symmetric-key cryptosystem to propose a multiserver authentication protocol but it was cracked soon. In 2004, a novel protocol was presented by Chang and Lee [11]. However, all of them ignored user
anonymity [12]. In 2009, a remote multiserver authentication scheme which satisfies anonymity property was proposed [13], but it does not have forward security [14]. Besides, Hsiang and Shih [15] presented a new protocol to resist various attacks; however some drawbacks on mutual authentication are pointed out [16]. Recently, a big breakthrough, i.e., the inner relationships of evaluation criteria for anonymous two-factor authentication protocol, is explored by Wang et al. [17]. To improve the security of remote communication, smart card gradually came into use in authentication, which made it possible for more convenient authentication and communication. Some remote authentication protocols for multiserver environment with a smart card were proposed but proved to be insecure in the end [21–23].

Relying on smart card and password as the authentication method already cannot meet today’s needs. In the latest few years, more and more authentication protocols adopt biometrics messages in mutual authentication to strengthen the security and enhance the efficiency of the existing protocols. In 2010, Yang et al. [24] introduced a three-factor multiserver authentication protocol. Unfortunately, the protocol has low computation efficiency and cannot resist the insider attack. Li et al. proposed an efficient protocol [25] which allows users to change the password and the calculation cost is low. However, their scheme cannot provide appropriate certification and failed to resist man-in-the-middle attack [26]. Adopting elliptic curve cryptography, Yoon et al. [27] in 2011 designed a novel protocol; unfortunately Kim et al. [28] showed that Yoon et al. [27] protocol is insecure. In 2014, Chuang et al. [29] put forward an anonymous protocol, but Mishra et al. [30] broke their protocol. Later, Lu et al. [31] found that there are several weaknesses in Mishra et al.’s improved protocol, and they also presented an improved protocol which is broken by Reddy et al. [32]. Meanwhile, Wang et al. [33] also found that Mishra et al.’s improved protocol is insecure. Regrettably, some weaknesses in Wang et al.’s protocol [33] were shown by Yang et al. [24] and Reddy et al. [34] separately. Recently, Jiang et al. [35] and He et al. put forward multiserver authenticated protocols using elliptic curve cryptography(ECC), separately. Unfortunately, Odelu et al. found that there are flaws in He et al.’s protocol in login and password change phases and can not resist the impersonation attack.

All of the above three-factor multiserver authentication protocols can be categorized into two classes, i.e., the protocols which implement the authentication independent of the registration center and the protocols which need the help of the registration center in the authentication phase. After carefully examining the known three-factor multiserver authentication protocols, we find that almost all of the first kind of protocols cannot resist the passive attack from an honest-but-curious server since all servers share a common secret key. More precisely, an honest-but-curious server can compute session keys which are shared between a user and other servers by eavesdropping messages transmitting between the users and other servers. In this paper, we take Wang et al.’s protocol as an example, to show how an honest-but-curious server obtains a session key which should be kept secret from him. Moreover, we find some other drawbacks in their protocol. For example, in the reregister or revocation phase, a user can still use his original password to login and send message even if he is revoked.

To resist the passive attack from the honest-but-curious servers, a trivial solution is to distribute different secret keys to different servers, which would aggravate the user’s storage burden. Another method is introducing the registration center into the authentication phase to deal with secret messages. As we mentioned above, such protocols are based on either ECC or symmetric encryption cryptosystems, which heavily affect the computation efficiency. To balance the security problem brought by the honest-but-curious servers and the efficiency problem brought by involving the registration center into the authentication, we propose a novel multiserver authentication protocol. In authentication phase of new protocol, the involved registration center only adopts hash and XOR operations for the computation, instead of ECC and symmetric cryptosystem, thus greatly improving the protocol’s computation efficiency. As far as we known, this is the first time to consider the passive attack from honest-but-curious servers for multiserver authentication protocols. Moreover, the new protocol is the first protocol which only adopts hash and XOR operations for computation when involving the registration center into the authentication.

The remaining of the paper is organized as follows. Section 2 reviews and analyzes the security of Wang et al.’s protocol. In Section 3, we present the new three-factor multiserver authentication scheme in detail. Section 4 provides the formal and informal secure analysis of the new protocol. In Section 5, comparisons including security, functionalities, computation cost, and communication cost are conducted. The last section gives a conclusion.

2. Some Weakness of Wang et al.’s Scheme

We firstly give the details about Wang et al.’s protocol and then show how an honest-but-curious server attacks their protocol step by step.

2.1. Review of Wang et al.’s Protocol. Wang et al.’s protocol involves five phases, i.e., registration phase, login phase, authentication phase, password changing phase, and revocation/reregistration phase which are executed by the user $U_i$, the server $S_j$, and the registration center $RC$. The symbols and notations are listed in Table 1. Assume $RC$ is a trusted third party which is able to register for users and servers.

**Registration Phase**

(i) Registration phase of server

(a) $S_j$ sends a request message to $RC$.

(b) $RC$ authorizes $S_j$ once it receives the message and returns $PSK$ (preshared key) to $S_j$ securely.

(c) $S_j$ uses $PSK$ to check $U_i$’s legitimacy in authentication phase.

(ii) Registration phase of user

(a) The new user $U_i$ inserts $SC_i$ into the card reader, inputs $ID_i$, $PW_i$, and imprints $B_i$ at the
Table 1: Symbols and notations in Wang et al.’s scheme.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_i$</td>
<td>the $i$th user</td>
</tr>
<tr>
<td>$S_j$</td>
<td>the $j$th server</td>
</tr>
<tr>
<td>RC</td>
<td>registration centre</td>
</tr>
<tr>
<td>$ID_i$</td>
<td>$U_i$’s identity</td>
</tr>
<tr>
<td>$AID_j$</td>
<td>$U_j$’s dynamic identity</td>
</tr>
<tr>
<td>$SID_j$</td>
<td>$S_j$’s identity</td>
</tr>
<tr>
<td>$SC_i$</td>
<td>$U_i$’s smart card</td>
</tr>
<tr>
<td>$PW_i$</td>
<td>$U_i$’s password</td>
</tr>
<tr>
<td>$BIO_i$</td>
<td>$U_i$’s biometrics</td>
</tr>
<tr>
<td>$R_i$</td>
<td>$U_i$’s nearly random string</td>
</tr>
<tr>
<td>$PSK$</td>
<td>pre-shared key between server and registration centre</td>
</tr>
<tr>
<td>$x$</td>
<td>master secret key between user and registration centre</td>
</tr>
<tr>
<td>@</td>
<td>exclusive OR</td>
</tr>
<tr>
<td>$h()$</td>
<td>one-way hash function</td>
</tr>
<tr>
<td>$</td>
<td></td>
</tr>
</tbody>
</table>

sensor. After that, $(R_i, P_i)$ is extracted from $BIO_i$ through $Gen(BIO_i) \rightarrow (R_i, P_i)$. Finally, $U_i$ calculates $RPW_i = h(PW_i || R_i)$ and sends $[ID_i,RPW_i]$ to RC securely.

(b) RC generates $<ID_j,N_j = 1>$ and stores it to the database. Note that $N_j$ indicates the state of $U_i$’s account. When $U_i$ revokes his/her account, RC sets $N_j = 0$. When $U_i$ regenerates his/her account, RC sets $N_j = N_j + 1$. After that, RC calculates $A_{j_i} = h(ID_j || x || T_j)$, $B_{j_i} = RPW_i \oplus h(A_{j_i})$, $C_{j_i} = B_i \oplus h(PSK)$, $D_{j_i} = PSK \oplus A_{j_i} \oplus h(PSK)$, and $V_{j_i} = h(ID_j || RPW_i)$ where $T_j$ is the time of registration. Finally, $RC$ sends $[B_{j_i}, C_{j_i}, D_{j_i}, V_{j_i}]$ to $U_i$ securely.

(c) $U_i$ receives $[B_{j_i}, C_{j_i}, D_{j_i}, V_{j_i}]$ from RC, stores $[B_{j_i}, C_{j_i}, D_{j_i}, V_{j_i}]$ into $SC_i$, and stores $P_i$ in $SC_i$.

**Login Phase**

(i) $U_i$ inputs $ID_i$ and $PW_i$ with his/her smart card $SC_i$ and imprint $BIO_i^*$. $SC_i$ calculates $RPW_i = h(PW_i || R_i)$. After that, $SC_i$ checks whether $h(ID_i || RPW_i) = V_{j_i}$ holds or not. If it is right, $SC_i$ computes $h(PSK) = B_i \oplus C_i$.

(ii) $SC_i$ calculates $AID_j = h(N_j \oplus ID_i)$ after choosing a random number $N_j$. After that, $SC_i$ computes $M_1 = RPW_i \oplus h(PSK) \oplus N_j$ and $M_2 = h(AID_j || N_j || RPW_i || SID_j || T_j)$ where $T_j$ is a timestamp.

(iii) $SC_i$ sends $[B_{j_i}, D_{j_i}, M_1, M_2, AID, T_j]$ to $S_j$.

**Authentication Phase**

(i) $S_j$ checks whether $T_j - T_{j_i} \leq \Delta T$. Note that $T_j$ is the time that $S_j$ receives the login message, and $\Delta T$ means the time interval.

(ii) If the verification is valid, $S_j$ calculates $A_{j_i} = PSK \oplus D_j \oplus h(PSK)$, $RPW_{j_i} = h(A_{j_i} \oplus B_i)$, and $N_{j_i} = M_1 \oplus RPW_i \oplus h(PSK)$ and checks whether $h(AID_j || N_j || RPW_i || SID_j || T_j) = M_2$.

(iii) If this verification is valid, $S_j$ chooses a random number $N_{j_2}$ and calculates $SK_{j_2} = h(AID_j || SID_j || N_j || N_{j_2})$ as the session secret key. Then, $S_j$ computes $M_3 = h(PSK) \oplus h(AID) || N_j \oplus N_{j_2}$ and $M_4 = h(SID_j || N_j || AID_i)$. Finally, $S_j$ returns $\{M_3, M_4, SID_j\}$ to $U_i$.

(iv) $SC_i$ calculates $N_2 = h(PSK) \oplus h(AID_i || N_j) \oplus M_3$ and $SK_{j_2} = h(AID_j || SID_j || N_j || N_{j_2})$ and checks whether $h(SID_j || N_j || AID_i) = M_4$. If it is valid, $SC_i$ calculates $M_5 = h(SK_{j_2} || N_j || N_{j_2})$ and sends $M_5$ to $S_j$.

(v) $S_j$ checks whether the condition $h(SK_{j_2} || N_j || N_{j_2})$ matches with $M_5$. If holds, $S_j$ confirms the session key $SK_{j_2}$. Otherwise, $S_j$ terminates the session immediately.

**Password Change Phase**

(i) $U_i$ inserts his/her smart card, inputs $ID_i$ and $PW_i$, and imprints $BIO_i^*$. $SC_i$ disposes $B_i$ with $B_i^*$, $C_i$ with $C_i^*$, and $V_i$ with $V_i^*$, respectively.

**Revocation and Reregistration Phase**

(i) If $U_i$ is revoked, he needs to send verification message $RPW_{j_i}$ to RC securely.

(ii) $RC$ checks the validity of $U_i$. If $U_i$ is a valid user, $RC$ sets $<ID_j, N_j = 0>$. 
2.2. Analysis of Wang et al.'s Protocol. This subsection analyzes Wang et al.'s protocol and shows how to mount a passive attack by an honest-but-curious server.

2.2.1. Passive Attack from an Honest-But-Curious Server. In this attack, an honest-but-curious server (say \( S_j \)) only passively eavesdrops messages between the user \( U_i \) and other servers, so that he can obtain the session keys shared by the user \( U_i \) and other servers which should be kept secret from \( S_j \) using his secret key and eavesdropping messages. More precisely, suppose a user \( U_i \) has finished the protocol with a server \( S_j \) and is running the protocol with the other server \( S_k \). Now we will show how the server \( S_j \) obtains the session key between \( U_i \) and \( S_k \) step by step.

(i) Step 1. \( S_j \) finished the protocol with \( U_i \) successfully. Thus the server has knowledge of PSK.

(ii) Step 2. During the protocol process between \( U_i \) and \( S_k \), firstly intercepts \( \{B_i, D_i, M_1, M_2, AID_i, T_i\} \) sent by \( U_i \) to \( S_k \). From these messages, \( S_j \) obtains \( N_1 \) by calculating \( A_1 = D_i \oplus PSK \oplus h(PSK) \). \( RPW_i = B_i \oplus h(A_1) \), and \( N_1 = RPW_i \oplus M_1 \oplus h(PSK) \).

(iii) Step 3. After that, \( S_j \) intercepts the messages \( \{SID_j, M_2, M_3\} \) which are sent from \( S_k \) to \( U_i \). Then the server \( S_j \) obtains \( N_2 \) by calculating \( N_2 = M_2 \oplus h(AID_j \parallel N_1) \oplus h(PSK) \). In this case, the server \( S_j \) acquired \( N_1 \) and \( N_2 \) generated in this session.

(iv) Step 4. With the intercepted \( AID_i \) and \( SID_j \), \( S_j \) can obtain the session key \( SK_{ij} \) by calculating \( SK_{ij} = h(AID_i \parallel SID_j \parallel N_1 \parallel N_2) \) successfully which should be kept secret from him.

2.2.2. User's Anonymity. User's anonymity means that user's ID and other urgent information indicating user's identity directly should be protected carefully. In Wang et al.'s scheme, an honest-but-curious server \( S_j \) can compute \( N_1 \) by \( N_1 = RPW_i \oplus M_1 \oplus h(PSK) \). At the same time, \( S_j \) receives \( AID_i \) from \( U_i \). \( S_j \) can obtain \( U_i \)'s identity \( ID_i \) by computing \( ID_i = AID_i \oplus h(N_1) \). As a consequence, the server can obtain \( ID_i \) of the user \( U_i \). This does not guarantee the anonymity of the user's identity.

2.2.3. User Impersonation Attack. The honest-but-curious server \( S_j \) can collect \( B_i \) and \( D_i \) which are sent by \( U_i \) and thus can calculate \( RPW_i \) using \( PSK \). After that, \( S_j \) can pretend to be \( U_i \) and apply authentication from other servers. Specifically, \( S_j \) randomly choose a number \( N_1 \) and calculates \( AID_i = h(N_1) \oplus ID_i \), \( M_1 = RPW_i \oplus N_1 \oplus h(PSK) \), and \( M_2 = h(AID_i \parallel N_1 \parallel RPW_i \parallel SID_j \parallel T_i) \). Then \( S_j \) sends \( \{B_j, D_j, M_1, M_2, AID_j, T_j\} \) to other servers through a public channel. In this way, \( S_j \) can disguise as \( U_i \).

2.2.4. Wrong Revocation and Reregistration. In this phase, users are allowed to revoke or reregister when he confronts the situation about losing the smart card or his account. In Wang et al.'s scheme, when a user wants to revoke his account, he has to pass the authentication. After that, \( RC \) changes \( N_i \) to 0, indicating that \( ID_i \) is not available any more. In the reregistration phase, \( RC \) also changes \( N_i \) to \( N_i + 1 \). However, \( RC \) is not involved in login phase and authentication phase. As a result, there is no access for \( RC \) to check whether the user's account is revoked or not. Thus the user can access the legal servers only using his former password and his biometrics; even he already has been revoked.

2.3. Reasons for the Weakness. In Wang et al.'s protocol, two important temporary secret values \( N_1 \) and \( N_2 \) are protected by PSK. Unfortunately, all servers keep the same private key PSK. As a result, an honest-but-curious server \( S_j \) with PSK can obtain all session keys which should be kept secret from him. This attack usually exists in the multiserver environment. In the most cases, a legitimate server after registration is assumed to be completely trustworthy, without taking into account the possibility that a particular server can act as an honest-but-curious adversary.

To resist this attack, it is bound to distribute different secret keys to different servers, which can be implemented by involving the registration center in the authentication process. Unfortunately, in most of this kind of protocol, authentication process is excused between the user and the server independent of the registration center. Therefore, to design a secure three-factor authentication protocol against the passive attack from an honest-but-curious server, the registration center should be introduced into the authentication process to protect the important temporary secret values \( N_1 \) and \( N_2 \).

3. The New Protocol

In this section, we first discuss the threat model used in our protocol. We then give the list of notations used in our proposed scheme. Finally, we describe the different phases relate to our scheme.

3.1. Threat Model. In this subsection, we introduce a threat model following the definition of [36–39].

(i) The adversary \( \mathcal{A} \) is able to control the open communication channel completely; that is, he can intercept, modify, delete, block, and resend the messages over the open channel.

(ii) The adversary \( \mathcal{A} \) can list all pairs of ID from the space of identities and PW from the space of passwords in a polynomial time.

(iii) The adversary \( \mathcal{A} \) can either intercept the password of the user via the malicious device or extract the parameters from the smart card, but both methods cannot be used together. An honest-but-curious server does not have this ability.

(iv) When \( \mathcal{A} \) acts as an honest-but-curious server, he can just listen the messages via the open channel.
3.2. The Proposed Protocol. As Wang et al.'s protocol, the new protocol also involves five phases. Table 2 lists the notations used in the new protocol.

### Table 2: Notations in the new protocol.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_i, S_j$</td>
<td>$i$th user and $j$th server</td>
</tr>
<tr>
<td>RC</td>
<td>the registration centre</td>
</tr>
<tr>
<td>$ID_j, AID_j, SID_j$</td>
<td>$U_i$'s identity, dynamic identity and $S_j$'s identity</td>
</tr>
<tr>
<td>$SC_j, PW_j, BIO_j$</td>
<td>$U_i$'s smart card, password and biometrics</td>
</tr>
<tr>
<td>$R_i, P_i$</td>
<td>valid pre-shared key between server and registration centre</td>
</tr>
<tr>
<td>$PSK$</td>
<td>master secret key between server and registration centre</td>
</tr>
<tr>
<td>$X$</td>
<td>master secret key between user and registration centre</td>
</tr>
<tr>
<td>$h(\cdot, \cdot, \cdot)$</td>
<td>hash function, XOR operation and concatenating operation</td>
</tr>
<tr>
<td>$x_i$</td>
<td>pre-shared key between $U_i$ and registration centre</td>
</tr>
<tr>
<td>$y_j$</td>
<td>pre-shared key between $S_j$ and registration centre</td>
</tr>
</tbody>
</table>

![Figure 1: Registration phase of server.](image)

3.2.1. Registration Phase

(i) Server registration phase

During server registration, $S_j$ communicates with RC to authenticate his validity and become a legislative server after receiving the preshared key sent from RC. The whole process of the server registration phase is shown in Figure 1.

(a) $S_j$ sends a request message to RC.
(b) RC authorizes $S_j$ and adds a novel entry $< SID_j, y_j >$ to the database where $y_j$ is a random number. Then $h(PW_j, y_j)$ is sent to $S_j$ by applying IKEv2 securely.
(c) $S_j$ adopts $PSK, y_j$ to protect the urgent messages and generates the session key $SK_{ij}$.

(ii) User registration phase

A user sends his personal information to RC and gets his own smart card by executing the process listed in Figure 2.

(a) $U_i$ inputs $BIO_i$ at the sensor and can obtain $\{R_i, P_i\}$ using $Gen(BIO_i) \to (R_i, P_i)$. Then $U_i$ selects $ID_i$ and $PW_i$ and calculates $RPW_i = h(PW_i \ | \ R_i)$. $U_i$ finally sends $\{ID_i, RPW_i\}$ to RC securely.
(b) RC generates a novel entry $< ID_i, x_i >$ to the database where $x_i$ is a random number that records the validity of $U_i$. If $U_i$ has revoked its account or the account is not available at present, RC generates a negative random number $x_i$; otherwise, $x_i$ is a positive random number. At the same time, $x_i$ is a preshared key.

After that, RC computes $A_i = h(ID_i \ | \ X \ | \ T_1)$, $B_i = RPW_i \oplus h(A_j)$, $C_i = B_i \oplus h(PW_i)$, $D_i = PSK \oplus A_i \oplus h(PW_i)$, $V_i = h(ID_i \ | \ RPW_i)$, $E_i = RPW_i \oplus x_i$, and $F_i = RPW_i \oplus h(X)$ where $T_1$ is the registration time and $X$ is the masker secret key between the user and the registration center.

(c) RC puts $\{B_i, C_i, D_i, V_i, E_i, F_i\}$ into $SC_i$. After that, RC issues it to $U_i$ securely.
(d) With $SC_i$, $U_i$ keeps $P_i$ into $SC_i$ and initials the authentication.

3.2.2. Login Phase. A user $U_i$ tries to login to a server $S_j$ by executing the steps shown in Figure 3.

(i) $U_i$ inputs $ID_j, PW_j$, and $BIO_i$, then his smart card can recover $R_i^*$ using $Rep(BIO_i, P_i) \to R_i^*$.
(ii) $SC_j$ computes $RPW_j^* = h(PW_j \ | \ R_j^*)$ and then checks whether $h(ID_j \ | \ RPW_j^*) = V_j$ or not. If it is true, $SC_j$ calculates $h(PW_i) = B_j \oplus C_j \oplus x_j$ and $h(X) = RPW_i \oplus F_j$. Otherwise, $U_i$ does not pass the identity authentication.
(iii) $SC_j$ generates a number $N_j$ randomly for each session and calculates $h(A_j) = RPW_i \oplus B_j \oplus AID = ID_j \oplus h(X) \oplus h(A_j)$, $M_1 = RPW_i \oplus N_j \oplus h(PW_i) \oplus x_j$, and $M_2 = h(AID_j) \oplus RPW_i \oplus SID_j \oplus T_j$ where $T_j$ is a timestamp.
(iv) $SC_j$ sends $\{AID_j, M_1, M_2, B_j, D_j, T_j\}$ to $S_j$.

3.2.3. Authentication Phase. This phase offers the details of mutually authentication which are indicated in Figure 3.

(i) $S_j$ receives the information from $U_i$ and verifies whether $T_j - T_j^* \leq \Delta T$ holds or not. If it holds, $S_j$ calculates $A_j = PSK \oplus D_j \oplus h(PW_i)$ and $RPW_i = B_j \oplus h(A_j)$. Otherwise, $S_j$ will reject the login request. Then $S_j$ checks whether $M_2 = h(AID_j \ | \ RPW_i \ | \ SID_j \ | \ T_j)$ if $T_j$. If it fails, the protocol would be stopped.
Figure 2: Registration phase of user.

Figure 3: Login and authentication phase.
Otherwise, the server computes \( AID'_j = AID_j@h(A_i) \), \( M_3 = M_1 @ RPW_i @ y_j \), and \( M_4 = h(AID'_j \| SID_j \| y_j \| T_j) \) and sends messages \( \{AID'_j, SID_j, M_3, M_4, T_j\} \) to RC.

(ii) RC receives the messages and verifies whether \( T_j - T_{uc} \leq \Delta T \) holds or not. If it holds, RC checks whether \( M_4 = h(AID'_j \| SID_j \| y_j \| T_j) \). If it fails, the request would be stopped. Otherwise, RC computes \( ID_j = AID'_j \oplus h(X) \). Then RC goes through the database \( < ID_j, x_j, \rangle < SID_j, y_j > \) stored in RC to get \( x_j \) and \( y_j \). If \( x_j \) is a negative number, the request would be stopped. After that, RC computes \( N_1 = M_3 @ h(PSK) @ x_j @ y_j, M_5 = N_1 @ y_j @ PSK, \) and \( M_6 = h(M_2) @ T_j \), where \( T_j \) is an additional timestamp. Finally, RC returns \( \{M_2, M_6, T_j\} \) to \( S_j \).

(iii) Once \( S_j \) receives the message from RC, it verifies whether \( T_j - T_{uc} \leq \Delta T \) holds or not. If it holds, \( S_j \) checks whether \( M_6 = h(M_2) @ T_j \). If it fails, the request will be stopped. Otherwise, \( S_j \) calculates \( N_1 = M_2 @ y_j @ PSK \). Selecting a negative number \( N_2 \) randomly, \( S_j \) computes \( SK_{ij} = h(AID) @ SID_j @ N_1 @ N_2) \), \( M_7 = N_1 @ h(AID_j \| N_1) @ h(PSK), \) and \( M_8 = h(SID_j \| N_2) @ AID_j) \). Then, the server sends \( \{SID_j, M_7, M_8\} \) to \( U_i \).

(iv) \( U_i \) retrieves \( N_2 \) and calculates \( SK_{ij} \) by computing \( N_2 = M_7 @ h(AID) @ N_1 \oplus h(PSK) \) and \( SK_{ij} = h(AID) @ SID_j @ N_1 @ N_2 \). After that, \( U_i \) checks whether \( M_8 = h(SID_j \| N_2) @ AID_j) \) holds or not. If it is valid, the user calculates \( M_9 = h(SK_{ij} @ N_1 @ N_2) \) and sends \( M_9 \) to \( S_j \).

(v) Finally, \( S_j \) receives \( M_9 \) and checks whether the equation \( h(SK_{ij} @ N_1 @ N_2) = M_9 \) holds or not. If so, a secret session key is generated successfully and can be used in the following communication. Otherwise, \( S_j \) would reject the authentication.

### 3.2.4 Password Change Phase

Using this phase, \( U_i \)'s password can be changed without any exchanging message from both RC and \( S_j \).

(i) \( U_i \) inputs \( ID_i \), the old password \( PW_i \), and imprints \( BIO_i \) as well and computes \( RPW_i = h(PW_i \| R_i) \) which is used to pass the authentication.

(ii) \( U_i \) inputs a new password \( PW_i^{\text{new}} \). After that, \( U_i \) computes \( RPW_i^{\text{new}} = h(PW_i^{\text{new}} \| R_i) \), \( B_i^{\text{new}} = B_i \oplus RPW_i \oplus RPW_i^{\text{new}} \), \( C_i^{\text{new}} = B_i^{\text{new}} \oplus h(PSK) \), \( V_i^{\text{new}} = h(ID_i \| RPW_i^{\text{new}}), E_i^{\text{new}} = E_i \oplus RPW_i \oplus RPW_i^{\text{new}}, \) and \( F_i^{\text{new}} = F_i \oplus RPW_i \oplus RPW_i^{\text{new}} \).

(iii) \( SC_i \) replaces \( B_i, C_i, E_i, F_i, \) and \( V_i \) with \( B_i^{\text{new}}, C_i^{\text{new}}, E_i^{\text{new}}, F_i^{\text{new}}, \) and \( V_i^{\text{new}} \).

### 3.2.5 User Revocation or Reregistration Phase

This phase is used for revocation and reregistration when \( U_i \)'s smart card \( SC_i \) is stolen or lost.

(i) In revocation phase, \( U_i \) sends revocation requests to RC. RC chooses a positive random number and modifies the value of \( x_i \) corresponding to \( U_i \) as that random number.

(ii) In reregistration phase, \( U_i \) sends reregistration requests to RC. RC selects a positive random number and sets it as \( x_i \) of \( U_i \).

### 4. Security Analysis of the New Protocol

#### 4.1 Verifying the New Protocol with BAN Logic

Burrows-Abadi-Needham (BAN) logic is introduced by Burrows et al. [40] and widely used to analyze the security protocol. In this subsection, BAN logic is used to prove that mutual authentication can be obtained after running the new protocol successfully. The notations and postulates in BAN logic are listed in Table 3.

We first define the test goals which the new protocol should achieve using BAN logic:

- \((g1)\) \( S_j \equiv U_i \equiv U_i \xrightarrow{SK_{ij}} S_j \)
- \((g2)\) \( S_j \equiv U_i \xrightarrow{SK_{ij}} S_j \)
- \((g3)\) \( U_i \equiv S_j \equiv U_i \xrightarrow{SK_{ij}} S_j \)
- \((g4)\) \( U_i \equiv U_i \xrightarrow{SK_{ij}} S_j \)

Secondly, we give the idealized form of the new protocol as follows:

- \((m1)\) \( U_i \rightarrow S_j : (ID_i, RPW_i, U_i \xrightarrow{N_1} S_j) \xrightarrow{y_j} RC \)
- \((m2)\) \( S_j \rightarrow RC : (AID'_j, SID_j, RPW_i, U_i \xrightarrow{N_1} S_j) \xrightarrow{y_j} RC \)
- \((m3)\) \( RC \rightarrow S_j : (T_i, U_i \xrightarrow{N_1} S_j) \xrightarrow{y_j} RC \)
- \((m4)\) \( S_j \rightarrow U_i : (SID_j, N_1, U_i \xrightarrow{SK_{ij}} S_j) \xrightarrow{y_j} RC \)
- \((m5)\) \( U_i \rightarrow S_j : (N_1, N_2, U_i \xrightarrow{SK_{ij}} S_j) \xrightarrow{y_j} RC \)

Next, we list the following initiative premises of the new protocol:

- \((p1)\) \( U_i \equiv \#N_1 \)
- \((p2)\) \( S_j \equiv \#N_2 \)
- \((p3)\) \( U_i \equiv U_i \xrightarrow{x_j} RC \)
- \((p4)\) \( RC \equiv U_i \xrightarrow{x_j} RC \)
- \((p5)\) \( S_j \equiv S_j \xrightarrow{y_j} RC \)
- \((p6)\) \( RC \equiv S_j \xrightarrow{y_j} RC \)
- \((p7)\) \( S_j \equiv RC \xrightarrow{y_j} U_i \xrightarrow{N_1} S_j \)
- \((p8)\) \( U_i \equiv S_j \Rightarrow U_i \xrightarrow{SK_{ij}} S_j \)
Table 3: BAN logic notations and postulates.

<table>
<thead>
<tr>
<th>Notations and postulates</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \equiv X )</td>
<td>P believes the statement X is true</td>
</tr>
<tr>
<td>( P \preceq X )</td>
<td>P sees X</td>
</tr>
<tr>
<td>( P \models X )</td>
<td>P once said that X or has sent a message containing X</td>
</tr>
<tr>
<td>( P \Rightarrow X )</td>
<td>P has control over X</td>
</tr>
<tr>
<td>( #X )</td>
<td>X is fresh</td>
</tr>
<tr>
<td>( P \overset{K}{\rightarrow} Q )</td>
<td>P and Q can communicate using the shared key K, only P, Q or a trusted third party know K</td>
</tr>
<tr>
<td>( (X)_{k} )</td>
<td>The formula X is combined with the formula K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notations and postulates</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \models K \odot Q, P \models [X]_{K} )</td>
<td>Rule(a): The message-meaning rule</td>
</tr>
<tr>
<td>( P \models Q \Rightarrow X, P \models Q \models X )</td>
<td>Rule(b): The jurisdiction rule</td>
</tr>
<tr>
<td>( P \models #(X), P \models Q \models X )</td>
<td>Rule(c): The nonce-verification rule</td>
</tr>
<tr>
<td>( P \models #(X), P \models (X, Y) )</td>
<td>Rule(d): The freshness-conjunctanation rule</td>
</tr>
<tr>
<td>( P \models X, P \models Y, P \models Q \models (X, Y) )</td>
<td>Rule(e): The belief rule</td>
</tr>
</tbody>
</table>

Finally, we analyze the new protocol using the BAN logic rules and the assumptions.

From message (m3), we obtain

\[ (p9) S_j \equiv U_i \Rightarrow U_i \xrightarrow{SK_{ij}} S_j. \]

From message (m4), we have

\[ (p10) U_i \equiv U_i \xrightarrow{N_1} S_j \]

From (p10), (S7), and Rule(a), we get

\[ (S7) U_i \triangleq (SID_j, N_1, U_i \xrightarrow{SK_{ij}} S_j) \xrightarrow{N_1} S_j. \]

From (p10), (S7), and Rule(a), we get

\[ (S8) U_i \equiv S_j \sim (SID_j, N_1, U_i \xrightarrow{SK_{ij}} S_j). \]

From (p1) and Rule(d), we get

\[ (S9) U_i \equiv \#(SID_j, N_1, U_i \xrightarrow{SK_{ij}} S_j). \]

From (S8), (S9), and Rule(c), we get

\[ (S10) U_i \equiv S_j \equiv (SID_j, N_1, U_i \xrightarrow{SK_{ij}} S_j). \]

From (S10) and Rule(e), we get

\[ (S11) U_i \equiv S_j \equiv U_i \xrightarrow{SK_{ij}} S_j (g3). \]

From (P8), (S11), and Rule(b), we get

\[ (S12) U_i \equiv U_i \xrightarrow{SK_{ij}} S_j (g4). \]

Finally, we analyze the new protocol using the BAN logic rules and the assumptions.

From message (m5), we have

\[ (S13) S_j \triangleq (N_1, N_2, U_i \xrightarrow{SK_{ij}} S_j) \xrightarrow{N_1, N_2} S_j. \]

From (p11), (S6), (S13), and Rule(a), we get

\[ (S14) S_j \equiv U_i \sim (N_1, N_2, U_i \xrightarrow{SK_{ij}} S_j). \]
From (p2) and Rule(d), we get
\[
(S15) \ S_j \equiv \#(N_1, N_2, U_j \leftrightarrow S_j).
\]
From (S14), (S15), and Rule(c), we get
\[
(S16) \ S_j \equiv U_j \equiv (N_1, N_2, U_j \leftrightarrow S_j).
\]
From (S16) and Rule(e), we get
\[
(S17) \ S_j \equiv U_j \equiv U_i \leftrightarrow S_j (g1).
\]
Finally, From (P9), (S17), and Rule(b), we get
\[
(S18) \ S_j \equiv U_i \leftrightarrow S_j (g2).
\]
According to (g1), (g2), (g3), and (g4), we conclude that the new protocol provides the mutual authentication and a shared secret key between the user and the server after a successful running of the protocol.

4.2. Formal Security Analysis. Recent research has shown that user-chosen passwords follow the Zipf’s law [41], a vastly different distribution from the uniform distribution. In this subsection, we provide a formal security analysis of the new protocol.

**Theorem 1.** Let \( l \) be the length of the biometric key \( BIO_i \), let \( \text{Hash} \) be the range space of hash function \( h(\cdot) \), and both \( C \) and \( s \) are the Zipf’s parameters [41]. Let \( q_{\text{send}} \) be Send queries and \( q_h \) be Hash oracle queries. For any adversary \( \mathcal{A} \) in polynomial time \( t \) against the new protocol \( P \) in the random oracle, the advantage of \( \mathcal{A} \) breaking the SK – security of \( P \) is
\[
\text{Adv}^\text{pke}_p \leq \frac{q_h}{\text{Hash}} + \frac{C \cdot q_{\text{send}}}{2^{l-1}}.
\]

**Proof.** Let \( E_i \) be the event that \( \mathcal{A} \) guesses bit \( b \) for \( G_i \) in the test session successfully. According to the new protocol, \( \mathcal{A} \) does not need to guess or compute the user’s identity since there is only one user. The games \( G_0 \) to \( G_3 \) are listed as follows.

**Game \( G_0 \)**. This game corresponds to the real attack in the random oracle model. Hence
\[
\text{Adv}^\text{pke}_p = |2\text{pr}[E_0] - 1|.
\]

**Game \( G_1 \)**. We simulate \( \mathcal{A} \)’s eavesdropping attack by querying \( \text{Test} \) oracles. Then, \( \mathcal{A} \) sends the Test query and decides whether the outcome of Test query matches with SK which can be calculated as \( SK_{ij} = h(AID_j \parallel SID_j \parallel N_1 \parallel N_2) \). \( \mathcal{A} \) cannot get the message about PSK, \( h(X) \), and \( AID_j \), due to the security of \( S_j \)’s PSK and \( U_j \)’s \( h(X) \). Thus \( \mathcal{A} \) cannot increase the chance of winning game \( G_1 \). Hence we have
\[
\text{pr}[E_0] = \text{pr}[E_1].
\]

**Game \( G_2 \)**. We simulate \( \mathcal{A} \)’s active attack by querying \( \text{Send} \) and \( \text{Hash} \) oracles. \( \mathcal{A} \) will manage to find the collisions of \( \text{Hash} \) in the way of make queries, but it is impossible for him to know the message of both \( \{AID_j, M_1, M_2, B, D_j, T_j\} \) and \( \{SID_j, M_2, M_3\} \) without the knowledge of \( T_j, N_1 \), and \( N_2 \). Hence there is no collision when querying \( \text{Send} \) oracles. Using the birthday paradox, we obtain
\[
|\text{pr}[E_2] - \text{pr}[E_1]| \leq \frac{q_h^2}{2^{2l}}.
\]

**Game \( G_3 \)**. This game simulates the smart card lost attack by querying \( \text{CorruptSC} \) oracle. If \( \mathcal{A} \) wants to obtain the secret information of users, he tries online dictionary attack due to the low entropy of password or other computing modes to get \( P_i \) which is used as the biometrics key with the message from \( SC_i \). Unfortunately, \( \mathcal{A} \) has to know \( R_i \in \{0, 1\}^l \) with the probability approximated as \( 1/2^l \), because we use fuzzy extractor function to extract at most \( l \) nearly random bits of \( R_i \). Even if using the Zipf’s law on passwords, we still have
\[
|\text{pr}[E_3] - \text{pr}[E_2]| \leq \frac{C \cdot q_{\text{send}}^2}{2^{l-1}}.
\]

Moreover, \( \mathcal{A} \) cannot get any useful messages about the value of \( c \) because of the independence and randomness of each session key. Thus, we have
\[
\text{pr}[E_3] = \frac{1}{2}.
\]

Combined the above steps, we can get the result as follows:
\[
\text{Adv}^\text{pke}_p \leq \frac{q_h}{\text{Hash}} + \frac{C \cdot q_{\text{send}}}{2^{l-1}}.
\]

4.3. Informal Security Analysis. In this subsection, informal security analysis is conducted to show that the new protocol can withstand various attacks.

**Replay Attack.** If \( \mathcal{A} \) replays a former piece of user’s messages to server, he will not success since a timestamp is used in each session to guarantee the freshness of the message. If the information in a previous session is replayed, the interval between \( T_j \) and \( T_j \) will not be in an endurable range. Therefore, in the authentication phase \( \mathcal{A} \) cannot pass the authentication in the first step. Hence, the new protocol can resist the replay attack.

**Modification Attack.** It is assumed that an adversary \( \mathcal{A} \) intercepts the information transmitted on the public channel and intends to modify the information to pass the authentication. Unfortunately, the integrity of the transmitted messages in the new scheme is protected by using one-way hash function. Moreover, \( \mathcal{A} \) cannot retrieve \( N_1 \) and \( N_2 \) from the intercepted messages, thus he cannot generate a legitimate authentication message. Therefore the new protocol can resist the modification attack.

**Server Session Key Attack.** In our proposed scheme, on one hand, session key \( SK_{ij} \) contains \( N_1, N_2, AID_j, SID_j, N_1 \), and \( N_2 \) which are different in every session and thus cannot
be retrieved directly by a malicious adversary $\sigma$. On the other hand, our scheme provides mutual authentication in the authentication phase and makes an improvement, i.e., both of the user and the server know whether $SK_{ij}$ has already been generated by each other. If the server $S_1$ wants to obtain the session key $SK_{ij}$ by calculation, he has to obtain $y_j$ since $N_1 = M_2 \oplus y_j \oplus PSK$. Unfortunately, the specific value of $y_j$ is known only to $S_2$ and $RC$. After receiving the messages transmitted from the user, the server calculates $M_4 = h(SK_{ij} \parallel N_1 \parallel N_2)$ which means that the authentication is passed and valid session key has already been generated by each other. Therefore our scheme holds the security of session Key.

**User Impersonation Attack.** If $\sigma$ is going to impersonate a valid user, $\sigma$ has to retrieve $BIO_i$, $PW_i$, and $ID_i$ of $U_i$ to pass the authentication in calculating $h(ID_i' \parallel RPW_i') = V_i$ in login phase. It is impossible for him to make it as a result of our perfect user anonymity and the uniqueness of biometric message. If the adversary wants to get access to $S_1$ as a valid user with the messages $AID_j$, $T_j$, $B_j$, $D_j$, $M_1$, and $M_2$, he cannot pass the check $M_2 = h(AID_j \parallel RPW_j \parallel SID_j \parallel T_j)$ and form a session key with the server he communicates with.

**Forgery Attack.** The forgery attack refers to the existence of a legitimate but malicious user $\sigma$ who attempts to falsify the identity information of another legitimate user to login and authenticate. In the communication between the legal server $S_i$ and the user $U_j$, the real identity $ID_j$ of $U_j$ is protected by $AID_j$, i.e., $AID_j = ID_j \oplus h(x) \oplus h(A_i)$. In addition, the identity $A_i$ is different for each user. Therefore, the malicious user $\sigma$ cannot obtain the real identity $ID_j$ of another legitimate user. Therefore, our scheme can prevent forgery attack.

**Masquerade Attack.** Under this attack, $\sigma$ can authenticate with the server $S_j$ as a legal user and attempt to acquire the session key $SK_j$ using the information transmitted at the authentication phase. In order to resist this attack, all messages transmitted in the public channel contain the destination or source information, such as $M_2 = h(AID_j \parallel RPW_j \parallel SID_j \parallel T_j)$ and $M_4 = h(AID_j' \parallel SID_j \parallel y_j \parallel T_j)$, with $AID$ or $SID$. So that $U_j$ and $S_j$ verify whether one wants to be authenticated by the other. Therefore our protocol can resist the masquerade attack.

**Smart Card Attack.** If the user’s smart card is stolen or lost and all the messages stored have been divulged by the adversary, there still no way for him to pass the authentication. At first, after acquiring $B_j$, $D_j$, $V_j$, $E_j$, and $F_j$, $\sigma$, still cannot get $ID_j$ and $RPW_j$. So $\sigma$ is not capable of forging a valid user $U_j$. Also, $\sigma$ cannot get any useful messages such as $RPW_j$, $A_j$, and $PSK$ using the messages stored in a smart card. Therefore, the new protocol is resistant to the stolen or lost smart card attack.

**Offline Guessing Attack.** $\sigma$ may get $B_i$, $C_i$, $D_i$, $E_i$, $F_i$, and $V_i$ by side channel attack such as SPA and DPA. However, he cannot change the user’s password without $BIO_i$, $h(PSK)$, $X$, or $x_i$ during the offline environment. In addition, one-way hash function is adopted to protect user’s password. Since it is impossible for different user to own the same biometric template, offline guessing attack can be avoided in the new protocol.

**DoS Attack.** DoS attack can seriously affect the efficiency of the server, causing the server to lose availability. However, all messages transmitted to the server and RC would be time stamped. With the help of the timestamp, the server and RC would verify the freshness and legitimacy of the message by checking $M_2$, $M_4$, and $M_6$. In addition, login operations require a fuzzy extractor to meet the biometric requirements. Therefore, our scheme can resist DoS attack.

**Server Spoofing Attack.** If $\sigma$ attempts to imitate a valid server, he is supposed to have the preshared key, a long-term secret key shared between $RC$ and $S_j$. In the new protocol, $h(PSK)$ and $y_j$ function as the preshared key which are transmitted through a secure channel and is unavailable to anyone other than $RC$ and servers. Without $y_j$, it is impossible for the adversary to calculate $N_j$ in the authentication phase since $N_1 = M_5 \oplus y_j$. And also, without $h(PSK)$ and $N_1$, the adversary cannot get $N_j$ since $N_2 = M_5 \oplus h(AID_j \parallel N_j) \parallel h(PSK)$. Thus the adversary cannot imitate a valid server.

**User Anonymity.** The users real identity is protected by $h(x)$ by replacing $ID_j$ with $AID_j$ where $AID_j = ID_j \oplus h(x) \oplus h(A_i)$. Also, due to the hash function and the secret key, either an outside adversary $\sigma$ or an honest-but-curious server cannot figure out $ID_j$ through $AID_j$. Thus the weak anonymity of the user is guaranteed.

Regrettably, the anonymity of the new scheme is not perfect. For example, assuming that the server cooperates with a malicious user, the malicious user provides $h(x)$ by calculating $h(x) = RPW_j \oplus F_j$, and the server calculates $A_j$ through $A_j = PSK \oplus D_j \oplus h(PSK)$, then the server can calculate $ID_j$ by calculating $ID_j = AID_j \oplus h(x) \oplus h(A_i)$. Moreover, an adversary $\sigma$ with $U_i$’s lost smart card can also compute $U_i$’s identity. Therefore, our scheme just provides the weak anonymity.

### 5. Efficiency Analysis

Efficiency analysis is conducted in this section to evaluate the new protocol. The comparisons including the resistance, functionality, and performance are summarized. In Table 5, let (S1) denote Chuang et al.’s protocol [29], (S2) denote Wang et al.’s protocol [33], (S3) denote Yang et al.’s protocol [24], (S4) denote Reddy et al.’s protocol [34], (S5) denote HE-WANG’s protocol [42], and (S6) denote Odelu et al.’s protocol [43]. The following notations are defined in Table 4.

Security comparison is offered by Table 5. In Table 5, “/” denotes that the security has not been analyzed until now. From Table 5, it is easy to see that protocols of (S1), (S2), (S3), and (S4), which do not include the registration center into the authentication phase, can not resist the passive attack from an honest-but-curious server. Although (S5) is resistant to above attack, it can not resist the user impersonate attack and smart card attack. The new protocol, together with (S6), achieves all resistance requirements, since they implement the authentication with the help of the authentication center. Thus they are more secure than the first five protocols.
Table 4: Notations in security comparison table.

<table>
<thead>
<tr>
<th>R1</th>
<th>resistance to replay attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>R2</td>
<td>resistance to modification attack</td>
</tr>
<tr>
<td>R3</td>
<td>resistance to Server session key attack</td>
</tr>
<tr>
<td>R4</td>
<td>resistance to user impersonate attack</td>
</tr>
<tr>
<td>R5</td>
<td>resistance to forgery attack</td>
</tr>
<tr>
<td>R6</td>
<td>resistance to masquerade attack</td>
</tr>
<tr>
<td>R7</td>
<td>resistance to smart card attack</td>
</tr>
<tr>
<td>R8</td>
<td>resistance to off-line guessing attack</td>
</tr>
<tr>
<td>R9</td>
<td>resistance to Dos attack</td>
</tr>
<tr>
<td>R10</td>
<td>resistance to server spoofing attack</td>
</tr>
</tbody>
</table>

Table 5: The security comparison.

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>/</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R3</td>
<td>No</td>
<td>No</td>
<td>/</td>
<td>/</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R4</td>
<td>No</td>
<td>No</td>
<td>/</td>
<td>/</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R5</td>
<td>Yes</td>
<td>Yes</td>
<td>/</td>
<td>/</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R6</td>
<td>No</td>
<td>No</td>
<td>/</td>
<td>/</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R7</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R8</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R9</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R10</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>/</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 6: Notations in functionality comparison table.

<table>
<thead>
<tr>
<th></th>
<th>anonymity</th>
<th>mutual authentication</th>
<th>session key agreement</th>
<th>perfect forward secrecy</th>
<th>user revocation/re-registration</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Functionalities comparison is listed in Table 7. The notations that appear in Table 7 are lists in Table 6. It can be seen that (S1), (S2), (S3), and (S4) do not provide user revocation/re-registration functionality, and (S5) does not offer anonymity property. Only our new protocol and (S6) provide all five basic functionality requirements.

Now we conduct the efficiency analysis including computation overhead and communication overhead. To compare with other related works, only login and authentication phase are considered.

Tables 9 and 10 list the computation cost comparisons from different aspects. The notations that appear in Table 9 are listed in Table 8. For the computation efficiency, we only calculate the number of hash functions, while ignore Exclusive OR operation and concatenating operation since they require little computational cost. Let $T_w$ denote the computation time for symmetric-key encryption/decryption which is known as about 0.005ms, $T_h$ denote the computation time for one-way hash function which is known as about 0.002ms, and $T_m$ denote the computation time for elliptic curve point multiplication which is known as about 2.226ms.

Table 9 compares the computation time according to protocol’s different phase. From Table 9, we can find that the new protocols, together with (S1), (S2), (S3), and (S4), spend almost the same time since only hash function contributes to computation cost. On the other hand, (S5) and (S6) take more time for computation due to the expensive elliptic curve point multiplication operations.

Table 10 compares the computation time according to different participants. The user’s executing time in the new protocol only needs 0.014ms, which proves that the new protocol provides the most efficient user’s computation. In terms of server’s executing time, the new protocol spends almost the same time as that of the most efficient protocols, i.e., (S1), (S2), and (S3). To resist the passive attack from the honest-but-curious servers, (S5), (S6), and the new protocol introduce the registration center into the authentication phase, which would bring extra burden for the trustworthy registration center. As shown in Table 10, $RC$ needs extra 4.47ms for (S5) and extra 2.263ms for (S6). In the new protocol, $RC$ is only used to transmit the secret information instead of authenticating user and server. As a result, the extra executing time for $RC$ in the new protocol is only 0.008ms, which is much less than that of (S5) and (S6). Therefore, the new protocol is the most efficient one among the second kind of multiserver authentication protocols. In conclusion, among all of the multiserver protocols against the passive attack from
Table 7: The functionality comparison.

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F2</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F3</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F4</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F5</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 8: Notations in computation comparison table.

<table>
<thead>
<tr>
<th></th>
<th>computation overhead in the login phase</th>
<th>execution overhead in the login phase</th>
<th>computation overhead in the authentication phase</th>
<th>execution overhead in the authentication phase</th>
<th>total execution overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9: Computation cost comparison in different phase.

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>4T_h</td>
<td>4T_h</td>
<td>5T_h</td>
<td>6T_h+2T_m</td>
<td>9T_h+3T_m</td>
<td>7T_h+2T_m+2T_w</td>
<td>3T_h</td>
</tr>
<tr>
<td>C2</td>
<td>0.008ms</td>
<td>0.008ms</td>
<td>0.01ms</td>
<td>2.238ms</td>
<td>4.458ms</td>
<td>4.467ms</td>
<td>0.006ms</td>
</tr>
<tr>
<td>C3</td>
<td>13T_h</td>
<td>1T_h</td>
<td>13T_h</td>
<td>9T_h+3T_m</td>
<td>18T_h+6T_m</td>
<td>19T_h+4T_m+5T_w</td>
<td>17T_h</td>
</tr>
<tr>
<td>C4</td>
<td>0.026ms</td>
<td>0.022ms</td>
<td>0.026ms</td>
<td>6.696ms</td>
<td>13.392ms</td>
<td>8.967ms</td>
<td>0.034ms</td>
</tr>
<tr>
<td>C5</td>
<td>0.034ms</td>
<td>0.03ms</td>
<td>0.036ms</td>
<td>8.934ms</td>
<td>17.85ms</td>
<td>13.434ms</td>
<td>0.04ms</td>
</tr>
</tbody>
</table>

Table 10: Computation cost comparison in different participants.

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>User cost</td>
<td>9T_h</td>
<td>8T_h</td>
<td>9T_h</td>
<td>9T_h+2T_m</td>
<td>7T_h+3T_m</td>
<td>7T_h+2T_m+2T_w</td>
<td>7T_h</td>
</tr>
<tr>
<td>User time</td>
<td>0.018ms</td>
<td>0.016ms</td>
<td>0.018ms</td>
<td>4.47ms</td>
<td>6.692ms</td>
<td>6.697ms</td>
<td>0.014ms</td>
</tr>
<tr>
<td>Server cost</td>
<td>8T_h</td>
<td>7T_h</td>
<td>9T_h</td>
<td>6T_h+2T_m</td>
<td>5T_h+3T_m</td>
<td>6T_h+2T_m+2T_w</td>
<td>9T_h</td>
</tr>
<tr>
<td>Server time</td>
<td>0.016ms</td>
<td>0.014ms</td>
<td>0.018ms</td>
<td>4.464ms</td>
<td>6.688ms</td>
<td>4.474ms</td>
<td>0.018ms</td>
</tr>
<tr>
<td>RC cost</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>4T_h</td>
</tr>
<tr>
<td>RC time</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>2.263ms</td>
</tr>
<tr>
<td>Total cost</td>
<td>17T_h</td>
<td>15T_h</td>
<td>18T_h</td>
<td>15T_h+4T_m</td>
<td>21T_h+8T_m</td>
<td>24T_h+6T_m+6T_w</td>
<td>20T_h</td>
</tr>
<tr>
<td>Total time</td>
<td>0.034ms</td>
<td>0.03ms</td>
<td>0.036ms</td>
<td>8.934ms</td>
<td>17.85ms</td>
<td>13.434ms</td>
<td>0.04ms</td>
</tr>
</tbody>
</table>

an honest-but-curious attack, the new protocol is the most computational efficient one.

Table 12 lists the new protocol’s communication cost together with the other related protocols. Suppose the random number x_i is 160 bits, the length of the user identity is 160 bits, the length of the timestamp is 16 bits, and the output length of one-way hash function is 160 bits if SHA-1 is adopted. Table 11 shows the notations that appear in Table 12. In the new protocol, when U_i logs in, he has to transmit M_1, AID_i, M_2, T_i, B_i, and D_i; thus the length of these messages is (160+5+16)/8 = 102 bytes. In the authentication phase, we introduce the registration center, so the communication cost is a little more than (S1), (S3), and (S4), about 180 bytes or so. Among all of the multiserver protocols against the passive attack from an honest-but-curious attack, the new protocol has the high communication efficient.

Combined with the security properties and the functionalities, we conclude that the new protocol and (S6) achieve all basic security properties and satisfy all functionalities. In terms of efficiency, (S6) spends much more computation time, bandwidth, and storage space compared with the new protocol. In conclusion, the new protocol is the most efficient multiserver authentication protocol which satisfies all basic security properties and functionalities.

6. Conclusion

In this paper, we found that a kind of multifactor multiserver authentication protocols can not resist the passive attack from an honest-but-curious servers. We took Wang et al.’s protocol as an example, to exhibit how an honest-but-curious server step by step obtained a session key which should
be kept secret from him. Moreover, we observed that the revocation and reregistration process in their protocol is incorrect. To remedy these weaknesses, this paper proposed a novel multiserver authentication protocol. The new protocol satisfies comprehensive demands of security and provides versatile and practical functionalities. Compared with the related protocols in computation cost and communication cost, the new protocol is the most efficient multiserver authentication protocol which satisfies all basic security properties and functionalities. Therefore, the new protocol is secure and relatively efficient in the remote distributed authentication networks. We have noticed that this kind of attack may also exist in other likewise environment, such as the multifactor multigateway authentication protocol in the wireless sensor networks. As a future work, we would apply the passive attack from an honest-but-curious gateway to the multifactor multigateway authentication protocol in the wireless sensor network and try to design secure protocols for multigateway wireless sensor network.

**Data Availability**

The paper does not use any data set.

**Conflicts of Interest**

The authors declare that they have no conflicts of interest.

**Acknowledgments**

This work was supported by the National Natural Science Foundation of China (nos. 61572027, 61772194, and U1636208), special foundation for coconstruction project of Beijing, the Hunan Provincial Natural Science Foundation of China under Grant no. 2018JJ3191, and the Guangxi Key Laboratory of Trusted Software (no. KX201707).

**References**


