Research Article

Improved ECC-Based Three-Factor Multiserver Authentication Scheme

Tsu-Yang Wu 1, Lei Yang 1, Zhiyuan Lee 1, Chien-Ming Chen 1, Jeng-Shyang Pan 1, and SK Hafizul Islam 2

1College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
2Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, Kalyani, West Bengal 741235, India

Correspondence should be addressed to Chien-Ming Chen; chienmingchen@ieee.org

Received 20 October 2020; Revised 25 November 2020; Accepted 29 December 2020; Published 19 January 2021

Academic Editor: Stelvio Cimato

Copyright © 2021 Tsu-Yang Wu 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.

A multiserver environment can improve the efficiency of mobile network services more effectively than a single server in managing the increase in users. Because of the large number of users, the security of users’ personal information and communication information is more important in a multiserver environment. Recently, Wang et al. proposed a multiserver authentication scheme based on biometrics and proved the security of their scheme. However, we first demonstrate that their scheme is insecure against a known session-specific temporary information attacks, user impersonation attacks, and server impersonation attacks. To solve the security weakness, we propose an improved scheme based on Wang et al.’s scheme. The security of our improved scheme is also validated based on the formal security analysis, Burrows–Abadi–Needham (BAN) logic, ProVerif, and informal security analysis. Security and performance comparisons prove the security and efficiency of our scheme.

1. Introduction

With the development of information technologies [1–8] and the widespread application of the Internet of Things [9–12], mobile communication has emerged in many network communication environments. The multiserver environments in mobile communication improve the efficiency of user communications; therefore, it is more popular than single-server environments for users. The multiserver environment overcomes the limited storage and computing of the single-server environment and can provide more remote services. A typical multiserver environment is shown in Figure 1.

Owing to the convenience of multiserver environments, authentication problems in the communication process cannot be disregarded. To date, three methods can be used to achieve user authentication in the environment. The first is password-based authentication [13–17]. This is the simplest method to perform authentication; however, an attacker can easily guess or steal a password from a party and impersonate as a valid user. The second is two-factor authentication, which is based on a password and a smart card [18–24]. Compared with password-based authentication, two-factor authentication improves security. However, if the smart card is stolen, then the information stored in the smart card may be recovered. This will result in well-known attacks, such as offline guessing attacks. In the past few years, Wang et al. have proposed some two-factor authentication schemes in different application scenarios. In 2014, they proposed an anonymous two-factor authentication scheme in a distributed system [19]. In the same year, they proposed an anonymous two-factor authentication scheme in a wireless sensor network [20]. In 2016, Wang et al. [25] compared and evaluated some representative two-factor authentication schemes and proposed a new evaluation standard for two-factor authentication schemes. In 2018, Wang et al. [26] proposed an evaluation framework for a two-factor authentication scheme for real-time data access in industrial wireless sensor networks and evaluated the relevant schemes. The third is three-factor authentication,
which is based on passwords, smart cards, and biometrics [27–39]. In a public channel, an attacker may eavesdrop, modify, or replay transmitted messages. This poses a significant threat to the security of users. Because only the password- or smart card-based authentication scheme exhibits low security, applying biometrics to authentication schemes can overcome the insecurity of password- or smart card-based schemes. Therefore, a secure and efficient authentication scheme based on biometrics must be designed.

Compared with Rivest–Shamir-Adleman (RSA) or ElGamal cryptosystems, elliptic curve cryptography (ECC) provides a small key size and computation efficiency under the same security level. In recent years, several biometric-based authentication schemes based on ECC have been proposed. In 2013, Pippal et al. [27] proposed a three-factor authentication scheme in a multiserver environment and claimed that their scheme can overcome all types of network attacks. In 2014, He and Wang [28] proposed a multiserver environment authentication scheme based on robust biometrics, claiming that their scheme was the first three-factor authentication scheme applicable to multiserver environments. In 2015, Odelu et al. [30] reported that the scheme proposed in [28] was vulnerable to a known session-specific temporary information attack and an impersonation attack and hence did not provide strong user anonymity; therefore, they proposed a secure multiserver authentication protocol based on biometric technology using smart cards. In the same year, Li et al. [31] discovered that Pippal et al.’s [27] scheme can provide incorrect authentication but could not overcome impersonation, stolen smart card, and internal attacks. Therefore, Li et al. [31] proposed an improved scheme to overcome the problems above. In 2017, Kumari et al. [32] proposed a provable secure multicloud server authentication scheme based on biometrics. However, in 2018, Feng et al. [33] discovered that the scheme presented in [32] could not guarantee user anonymity, three-factor security, perfect forward security, etc.; hence, they proposed a multiserver environment authentication scheme based on anonymous biometrics. In the same year, Ali and Pal [34] analyzed Li et al.’s [31] scheme and discovered that it could not overcome password-guessing, user impersonation, insider, and smart card theft attacks nor could they guarantee user anonymity.

Ali and Pal [34] proposed a three-factor multiserver authentication scheme based on an elliptic curve cryptosystem to solve the abovementioned issues. Unfortunately, Wang et al. [36] discovered that the scheme presented in [34] was vulnerable to user impersonation, server impersonation, privileged insider, and denial-of-service attacks, among others, and could not provide both forward and three-factor confidentiality. Therefore, Wang et al. proposed an improved multiserver authentication scheme based on biometrics and claimed that their scheme can overcome offline password-guessing, user impersonation, server impersonation, known specific session temporary information, three-factor security, user anonymity, and privileged internal attacks. Some important related works are summarized in Table 1.

In this study, we investigated Wang et al.’s scheme subject to known session-specific temporary information, user impersonation, and server impersonation attacks. To overcome the abovementioned attacks, we refer to Wang et al.’s scheme and propose an improved authentication scheme. Finally, we demonstrate that our scheme is semantically secure in the ROR model and overcome known attacks using the ProVerif tool and the BAN logic.

The remainder of this paper is organized as follows. A simple review and cryptanalysis of the scheme proposed by Wang et al. is discussed in Sections 2 and 3, respectively. Section 4 elaborates the proposed scheme in detail. Section 5 demonstrates the security analysis of the proposed scheme. Section 6 presents a comparison of performance and security. Section 7 summarizes the paper.

2. Review of Wang et al.’s Scheme

Wang et al.’s scheme includes initialization, server and user registration, and login authentication phases. Their scheme involves three types of entities: users, servers, and a registration center. The notations used in the scheme and their descriptions are shown in Table 2.

2.1. Initialization. In this phase, the registration center (RC) selects an elliptic curve $E_q$, and the basic point $P$ of $E_q$ defines two hash functions $H(\cdot)$ and $h(\cdot)$. Subsequently, the
Secure channel. When RC receives this message, it computes its identity SID. Subsequently, User selects a random number and computes \( P = H(\text{SID}||x) \), and sends \( \{\text{ID}, P, P_i\} \) to the RC. The RC receives this message and calculates the following:

\[
\begin{align*}
A_i &= H(x||\text{ID}_i), \\
B_i &= A_i \oplus P_i, \\
V_i &= H(P_i \oplus H(\text{ID}_i)) \mod n,
\end{align*}
\]

where \( 2^4 \leq n \leq 2^8 \). Note that \( H(P_i \oplus H(\text{ID}_i)) \) is the technique of fuzzy-verifier \([40]\). The RC stores \( B_i, V_i, E_{\text{key}}(\cdot), P_i, P_{\text{pub}}, n \) in the smart card (SC) and then sends the SC to User, in a secure channel. Subsequently, User stores \( r_i \) in the SC.

### 2.3. Login and Authentication

In this phase, User and Server complete a mutual authentication and establish a session key (SK) with the aid of the RC.

Step 1 User enters ID, and PW, imprints \( b \), and logs in the SC. Subsequently, the SC computes

\[
P_i' = H(\text{PW}, h(b_i)) \mod n, \\
V_i' = H(P_i' \oplus H(\text{ID}_i)) \mod n,
\]

and verifies if \( V_i' = V_i \). If they are equal, then User generates a random number \( N_1 \) and computes

\[
\begin{align*}
A_i' &= B_i \oplus P_i', \\
R_i &= N_1 P_i, \\
C_i &= H(N_1 P_{\text{pub}}), \\
L_i &= E_{C_i}(\text{ID}||A_i'||\text{SID}_i).
\end{align*}
\]

Next, User sends \( \{R_i, L_i\} \) to the RC in the public channel.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Cryptographic techniques</th>
<th>Limitations</th>
</tr>
</thead>
</table>
| Pippal et al. [27] | (1) Utilized one-way hash function  
(2) Based on Diffie–Hellman problem  
(3) Based on smart card | (1) Does not resist impersonation attacks  
(2) Does not resist internal attacks |
| Li et al. [31] | (1) Utilized one-way hash function  
(2) Based on Diffie–Hellman problem  
(3) Based on smart card | (1) Does not resist password-guessing attacks  
(2) Does not resist impersonation attacks  
(3) Does not resist internal attacks  
(4) Does not resist smart card theft attacks  
(5) Does not support user anonymity |
| Kumari et al. [32] | (1) Based on biometrics  
(2) Utilized one-way hash function  
(3) Based on anonymous authentication | (1) Does not support user anonymity  
(2) Does not resist man-in-the-middle attacks |
| Feng et al. [33] | (1) Utilized ECC  
(2) Based on smart card  
(3) Based on biometrics | (1) Does not provide three-factor secrecy  
(2) Does not resist known session-specific temporary information attack |
| Ali and Pal [34] | (1) Utilized ECC  
(2) Three-factor security  
(3) Based on data encryption scheme | (1) Does not resist impersonation attacks  
(2) Does not resist internal attacks  
(3) Does not provide forward secrecy  
(4) Does not provide three-factor secrecy  
(5) Does not resist known session-specific temporary information attack |
| Wang et al. [36] | (1) Utilized ECC  
(2) Based on biometrics  
(3) Based on data encryption scheme | (1) Does not resist impersonation attacks  
(2) Does not resist known session-specific temporary information attack |

### Table 2: Notations and descriptions.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC</td>
<td>The registration center</td>
</tr>
<tr>
<td>Server(_i)</td>
<td>The (i)-th server</td>
</tr>
<tr>
<td>User(_i)</td>
<td>The (i)-th user</td>
</tr>
<tr>
<td>A</td>
<td>The attacker</td>
</tr>
<tr>
<td>(P_{\text{pub}})</td>
<td>The public key of RC</td>
</tr>
<tr>
<td>(H(\cdot))</td>
<td>Hash function</td>
</tr>
<tr>
<td>(h(\cdot))</td>
<td>Biohash function</td>
</tr>
<tr>
<td>(\text{ID}_i)</td>
<td>User identity</td>
</tr>
<tr>
<td>(\text{PW}_i)</td>
<td>User’s password</td>
</tr>
<tr>
<td>(b)</td>
<td>User’s biometrics</td>
</tr>
<tr>
<td>SC</td>
<td>Smart card</td>
</tr>
<tr>
<td>SK</td>
<td>Session key</td>
</tr>
<tr>
<td>(\oplus)</td>
<td>Concatenation</td>
</tr>
<tr>
<td>(\oplus)</td>
<td>Bitwise XOR</td>
</tr>
<tr>
<td>(E_{\text{key}}(\cdot)/D_{\text{key}}(\cdot))</td>
<td>Symmetric encryption/decryption algorithm with key</td>
</tr>
</tbody>
</table>

RC selects a random number \( x \) and computes the public key \( P_{\text{pub}} = xP \), where \( x \) is the RC’s secret key and publishes \( \{E_{\text{pub}}(x), P_{\text{pub}}, H(\cdot), h(\cdot)\} \).
Step 2 After the RC receives \( \{R_i, L_i\} \), it computes
\[
C_i' = H(xR_i),
\]
\[
(\text{ID}, A_i'|\text{SID}) = D_C, (L_i),
\]
\[
A_i = H(x|| \text{ID}_i),
\] (4)
and verifies if \( A_i' = A_i \). If they are equal, then the RC computes
\[
\text{SM}_j = H(\text{SID}_j || x),
\]
\[
Y_i = H(\text{SID}_j || \text{SM}_j),
\] (5)
\[
M_i = E_{\text{SM}_j}(\text{ID}_i || R_i || Y_i || H( A_i || C_i')).
\]

Next, the RC sends \( \{M_i\} \) to Server \( j \) in the public channel.

Step 3 After Server \( j \) receives \( \{M_i\} \), it computes
\[
(\text{ID}_j || R_j || Y_j || H( A_j || C_j')) = D_{\text{SM}_j}(M_i),
\] (6)
\[
Y_i' = H(\text{SID}_j || \text{SM}_j),
\]
and verifies if \( Y_i' = Y_i \). If they are equal, then Server \( j \) generates a random number \( N_j \) and computes
\[
R_j = N_j P,
\]
\[
E_i = N_j R_i,
\]
\[
\text{SK}_j = H(E_i || Ht( A_j || C_j')),
\] (7)
\[
F_i = H(\text{ID}_j || \text{SK}_j || tR_j || n\text{SID}_j).
\]

Subsequently, Server \( j \) sends \( \{R_j, F_i\} \) to User \( i \) in the public channel.

Step 4 After User \( i \) receives \( \{R_j, F_i\} \), he computes
\[
E_i' = N_i R_j,
\]
\[
\text{SK}_i = H(E_i' || Ht( A_i || C_i)),
\] (8)
\[
F_i' = H(\text{ID}_i || \text{SK}_i || tR_j || n\text{SID}_j).
\]
and verifies if \( F_i' = F_i \). If they are equal, then User \( i \) computes
\[
Q_i = H(\text{SK}_j || R_j).
\] (9)

Next, User \( i \) sends \( \{Q_i\} \) to Server \( j \) in the public channel.

Step 5 After Server \( j \) receives \( \{Q_i\} \), it computes
\[
Q_i' = H(\text{SK}_j || R_j),
\] (10)
and verifies if \( Q_i' = Q_i \). If they are equal, then \( \text{SK}_i = \text{SK}_j \) is the session key for User \( i \) and Server \( j \).

3. Cryptanalysis of Wang et al.’s Scheme

In this section, we demonstrate Wang et al.’s scheme subject to three security attacks. In our proposed attacks, we assumed that the attacker \( A \) is a legitimate user and has already registered with the RC.

3.1. Known Session-Specific Temporary Information Attack. A known session-specific temporary information attack refers to a security attack in which an attacker attempts to obtain the current SK when temporary secret values such as random numbers are disclosed [41].

In this attack, we assume that the attacker \( A \) obtains temporary information \( N_i, \) and captures \( \{R_i, L_i\} \) and \( \{R_j, F_i\} \), which are transmitted over the public channel. Based on the formulas above, \( A \) can compute
\[
C_i = H(N_i P_{\text{pub}}),
\]
\[
(\text{ID}_j || A_j' || \text{SID}) = D_C, (L_j),
\] (11)
\[
E_i' = N_i R_j.
\]

Subsequently, \( A \) obtains \( C_i, A_i', \) and \( E_i' \); hence, it can determine \( \text{SK}_j = H(E_i' || Ht( A_i || C_i)). \) Furthermore, based on the formulas above, \( A \) can obtain the user’s ID; in other words, the user’s anonymity is not protected.

3.2. User Impersonation Attack

Step 1 Based on Section 3.1, \( A \) can obtain \( \text{ID}_i, A_i', \) and \( \text{SID}_j. \) Subsequently, \( A \) generates a random number \( N_A \) and computes
\[
R_A = N_A P,
\]
\[
C_A = H(N_A P_{\text{pub}}),
\]
\[
L_A = E_{C_A}(\text{ID}_j || A_j' || \text{SID}_j).
\] (12)

\( A \) fake User \( i \) to send \( \{R_A, L_A\} \) to the RC.

Step 2 Upon receiving \( \{R_A, L_A\} \), the RC computes
\[
C_A' = H(xR_A),
\]
\[
(\text{ID}_j || A_j' || \text{SID}) = D_C, (L_A),
\] (13)
\[
A_i = H(x|| \text{ID}_i).
\]

It is clear that \( A_i' = A_i. \) Next, the RC computes
\[
\text{SM}_j = H(\text{SID}_j || x),
\]
\[
Y_i = H(\text{SID}_j || \text{SM}_j),
\] (14)
\[
M_A = E_{\text{SM}_j}(\text{ID}_j || R_A || Y_A || H(A_i || C_A)),
\]
and sends \( \{M_A\} \) to Server \( j. \)

Step 3 After receiving \( \{M_A\} \), Server \( j \) computes
\[
(\text{ID}_j || R_j || Y_j || H(A_i || C_A)) = D_{\text{SM}_j}(M_A),
\] (15)
\[
Y_i' = H(\text{SID}_j || \text{SM}_j).
\]
4. Improved Scheme

To overcome the attacks, we proposed an improved scheme based on Wang et al.’s scheme in this section. Our scheme still operates in a multiserver environment, including the initialization, modified server and user registration, and modified login and authentication phases. It is noteworthy that the initialization phase in our scheme is the same as that in Wang et al.’s scheme, and we used a rectangle to denote our modifications.

4.1. Modified Server and User Registration. The server $S$ selects its identity $SID_j$ and sends its identity to the RC through a secure channel. The RC receives this message and selects a random number $e_j$. Subsequently, the RC computes $SM_j = H(SID_j || e_j)$, stores $\{SID_j, e_j\}$, and sends $SM_j$ to $S$. When $S$ receives $SM_j$, it stores it in the database.

The user $U_i$ selects his ID $i$ and PW, and imprints $b_i$. Subsequently, $U_i$ selects a random number $r_i$ and computes

$$P_i = H(PW_i || h(b_i)) || r_i),$$

$$HID_i = H(ID_i || r_i),$$

and sends $\{HID_i, ID_i, P_i\}$ to the RC. The RC receives this message, selects a random number $d_i$, and computes

$$A_i = H(x || HID_i || r_i || nd_i),$$

$$B_i = A_i \oplus P_i,$$

$$V_i = H(P_i \oplus H(HID_i)) \mod n,$$

where $2^4 \leq n \leq 2^8$. The RC stores $\{HID_i, ID_i, d_i\}$ in the database, stores $\{B_i, V_i, E_{key}(\cdot), P_i, P_{pub} || n\}$ in the SC, and sends the SC to $U_i$ in a secure channel. Next, $U_i$ stores $r_i$ in the SC. The complete registration process is shown in Figure 2.

4.2. Modified Login and Authentication. In this phase, $U_i$ and $S$ complete a mutual authentication and use the RC as an information center to establish an SK. The complete login and authentication processes are shown in Figure 3.

Step 1 $U_i$ enters $ID_i$ and $PW_i$, imprints $b_i$, and logs in the SC. Next, the SC computes

$$P'_i = H(PW_i || h(b_i)) || r_i),$$

$$HID'_i = H(ID_i || r_i),$$

$$V'_i = H(P_i \oplus H(HID'_i)) \mod n,$$

and verifies if $V'_i = V_i$. If they are equal, $U_i$ generates a random number $N_i$ and computes
**User**

Registration phase:

- Choose SID
- Compute SM = H(SID || x || ej)
- Store {SID, ej} in database

Login and authentication phase:

- Input {IDi, PWi, bi}
- Compute P′i = H(PWi || h(bi) || ri)
- Compute A′i = H(x || HID′i || IDi || di)
- Compute D1 = D1 @ A′i @ HID′i @ N1
- Compute G1 = H(N1 @ A′i @ HID′i), R1 = G1 @ P
- Compute C1 = H(G1 @ Ppub), L1 = Ec(HID′i || A′i || SIDj)
- Compute E′i = N1 P
- Compute SKi = H(E′i) H(A′i || C1)
- Compute F′i = H(HID′i || SKi || rid || SIDj)
- Check F′i = Fj

**Server**

Choose SID

Choose ej

Choose ej

Choose ej

Compute SM = H(SID || x || ej)

Store {SID, ej} in database

Store {SM} in database

Choose IDi, PWi, bi

Choose ri

Compute P′i = H(PWi || h(bi) || ri)

HID′i = H(IDi @ ri)

Choose di

Choose dj

Choose dj

Choose dj

Choose dj

Choose dj

Compute A′i = H(x || HID′i || IDi || ri)

N′1 = D1 @ A′i @ HID′i

G′1 = H(N′1 @ A′i @ HID′i)

HID′i = H(IDi || di)

Check A′i? = Aji

Compute SM = H(SID || x || ej)

Yj = H(SID || SMj)

Mj = ESms(Mj)

Compute (HID′ || Yj || H(A′i || C1)) = Dmsj (Mj)

Check Yj? = Yj

Choose N1

Choose N1

Compute Rj = N1 P, Ej = Nj Rj

Compute SKj = H(Ej) H(A′i || C1)

Fj = H(HID || SKj || rid || SIDj)

Figures 2 and 3: Registration and login authentication phases.
\[ A_i' = B_i \oplus P_i, \]
\[ D_i = H(A_i' \oplus \text{HID}_i') \oplus N_i, \]
\[ G_i = H(N_i' \oplus A_i' \oplus \text{HID}_i'), \]
\[ R_i = G_i P, \]
\[ C_i = H(G_i P_{\text{pub}}), \]
\[ L_i = E_{C_i}(\text{HID}_i' \parallel A_i' \parallel \text{SID}_i). \]

Subsequently, User, sends \( M_1 = \{D_i, \text{HID}_i', L_i\} \) to the RC in the public channel.

Step 2 After the RC receives \( M_1 \), it retrieves \( \{\text{HID}_i', \text{ID}_j, d_i\} \) in the database and computes
\[ A_i' = H(x \parallel \text{HID}_i' \parallel \text{ID}_j \parallel nd_i), \]
\[ N_i' = D_i \oplus H(A_i' \parallel \text{HID}_i'), \]
\[ G_i = H(N_i' \parallel A_i' \parallel \text{HID}_i'), \]
\[ R_i = G_i P, \]
\[ C_i = H(xR_i), \]
\[ (\text{HID}_i' \parallel A_i' \parallel \text{SID}_i) = D_{C_i}(L_i), \]
and verifies if \( A_i' = A_i' \). If they are equal, the RC computes
\[ \text{SM}_j = H(\text{SID}_j \parallel x \parallel \text{te}_i), \]
\[ Y_i = H(\text{SID}_j \parallel \text{SM}_j), \]
\[ M_i = E_{\text{SM}_j}(\text{HID}_i' \parallel R_i' \parallel Y_i \parallel H(A_i' \parallel C_i)). \]

Next, the RC sends \( M_2 = \{M_i\} \) to Server, in the public channel.

Step 3 After Server, receives \( M_2 \), it computes
\[ (\text{HID}_i' \parallel R_i' \parallel Y_i \parallel H(A_i' \parallel C_i)) = D_{\text{SM}_j}(M_i), \]
\[ Y_i' = H(\text{SID}_j \parallel \text{SM}_j), \]
and verifies if \( Y_i' = Y_i \). If they are equal, Server, generates a random number \( N_2 \) and computes
\[ R_S = N_2 P, \]
\[ E_S = N_2 R_S, \]
\[ SK_j = H(E_S \parallel H(A_i' \parallel C_i)), \]
\[ F_i = H(\text{HID}_i' \parallel SK_j \parallel tR_S \parallel n\text{SID}_j). \]

Subsequently, Server, sends \( M_3 = \{R_S, F_i\} \) to User, in the public channel.

Step 4 After User, receives \( M_3 \), it computes
\[ E_i' = N_i R_S, \]
\[ \text{SK}_i = H(E_i' \parallel H(A_i' \parallel C_i)), \]
\[ F_i' = H(\text{HID}_i' \parallel \text{SK}_i \parallel tR_S \parallel n\text{SID}_j), \]
and verifies if \( F_i' = F_i \). If they are equal, SK is the session key for User, and Server.

5. Security Analysis

5.1. Formal Security Analysis. In this section, we show the security analysis of our improved scheme in the random oracle model [42]. First, we define the adversarial model [25, 26, 43–47] and simulate the adversary capabilities in a real attack. In the proposed scheme, three participants, User, Server, and RC, are involved. We use \( \prod_{i=1}^{x} \prod_{j=1}^{y} \prod_{\text{RC}} \) to represent the \( x \)th communication of User, the \( y \)th communication of Server, and the \( z \)th communication of RC, respectively. To perform a formal security analysis, we defined the following query model for the attacker \( A \).

Execute (\( \prod_{i=1}^{x} \prod_{j=1}^{y} \prod_{\text{RC}} \)): A performs this query to eavesdrop and record the messages transmitted on the public channel, such as the messages between the \( U \) and the RC, the messages between the RC and the \( S \), and the messages between the \( S \) and the \( U \).

Hash (\( \prod_{i=1}^{x} M_1 \)): based on this query, \( A \) can get the hash value if each item in the hash function is known, where \( E = \{U, S, \text{RC}\} \).

Send (\( \prod_{i=1}^{x} M_1 \)): \( A \) executes this query with message \( M_1 \) and then receives the response message from the entity \( E \).

Reveal (\( \prod_{i=1}^{x} E \)): \( A \) executes this query to obtain the return result of current session key SK generated by \( E \).

Corrupt (\( \prod_{i=1}^{x} E \)): \( A \) executes this query to obtain information \( \{B_i, V_i, r_i, E_{\text{key}}(\cdot), P, P_{\text{pub}}, n\} \) in the smart card.

Test (\( \prod_{i=1}^{x} E \)): based on this query, an unbiased coin \( c \) begins to be flipped. If \( c = 0 \), \( A \) returns SK to a random string, and if \( c = 1 \), \( A \) returns SK to a session key.

In the ROR model, the following theorem describes the security of our proposed scheme \( P \).

Theorem 1. If \( A \) runs \( P \) in an ROR model against a scheme in polynomial time, \( I \) represents the total number of bits of the biometric. The \( \text{Adv}_{\text{A KE}}^{\text{A KE}}(\cdot) \) that \( A \)‘s advantage breaks the security of \( SK \) in \( \text{AKE} \) scheme, and then \( \text{Adv}_{\text{A}}^{\text{A KE}}(\cdot) \leq \frac{q_{\text{A KE}}}{I} \cdot \text{Hash} + 2 \max\{C' \cdot q_{\text{endP}}, (q_{\text{endP}}/2)\} + 2 \text{Adv}_{\text{A}}^{\text{P}}(\cdot) \), where \( q_{\text{endP}} \) and \( q_{\text{A}} \) are the number of Send (\( \prod_{i=1}^{x} M_1 \)) and Hash (\( \prod_{i=1}^{x} M_1 \)) \( \text{Hash} \) is the range space of \( h(\cdot) \); \( C' \) and \( s \) are parameters of Zipp’s law [48]; \( \text{Adv}_{\text{A KE}}^{\text{A KE}}(\cdot) \) is the advantage of \( A \) breaking the symmetric cipher \( \Omega \).

Proof. We define a sequence of five games, namely, \( \text{GM}_i \) (\( i = 0, 1, 2, 3, 4 \)). Let \( \text{Succ}_{\text{GM}_i}^{\text{GM}_i} \) represent the event that \( U_A \) wins \( \text{GM}_i \). The \( \text{Adv}_{\text{A}}^{\text{GM}_i}(\cdot) \) represents the advantage of \( A \) winning \( \text{GM}_i \), where \( \text{Pr}[E] \) is the probability of event \( E \). The \( \text{Adv}_{\text{A}}^{\text{GM}_i}(\cdot) \) represents the advantage of \( U_A \) that breaks the security of SK in the proposed scheme. The detailed description of \( \text{GM}_i \) is as follows.

Game \( \text{GM}_0 \): \( \text{GM}_0 \) is the first game that represents a real attack on the ROR model. At this point, select coin \( c \) to
start GM₀. From semantic security, we can get \( \text{Adv}^P_{\text{GM}} = [2 \cdot \text{Adv}^P_{\text{GM}}] - 1 \).

**Game GM₁:** GM₁ means that A can perform the Execute query and get the message \( \{D_i, \text{HID}_i, L_i, \{M_i\} \} \) and \( \{R_i, F_i\} \) transmitted in the scheme. At the end of the game, A will perform Revel and Test queries to determine whether \( \text{SK} = H(E_i \parallel H(A_i \parallel C_i)) \) can be obtained. But A cannot derive \( E_i, A_i, \) and \( C_i \), so the probability of GM₁ is the same as that of GM₀, that is, \( \text{Adv}^P_{\text{GM}, 1} = \text{Adv}^P_{\text{GM}} \).

**Game GM₂:** GM₂ has added Hash and Send queries, \( E_i, A_i, \) and \( C_i \), which are all protected by \( h(\cdot) \). But \( E_i, A_i, \) and \( C_i \) are not directly obtained in the transmission channel, and according to the birthday paradox, we can get \( |\text{Adv}^P_{\text{GM}, 2} - \text{Adv}^P_{\text{GM}}| \leq (q_i^2/2)[\text{Hash}] \).

**Game GM₃:** Corrupt query is added in GM₃ and A can get the information \( \{B_i, V_i, r_i, E_{\text{key}}(\cdot), P, P_{\text{pub}}, n\} \) in the smart card. The User uses the password and biometric information to register, and A wants to guess \( P = H(PW_i \parallel h(b_1) \parallel r_i) \), but the probability of guessing the biometrics is 1/2 [49], which is almost negligible. Using Zipf's law [48], we can get \( |\text{Adv}^P_{\text{GM}, 3} - \text{Adv}^P_{\text{GM}}| \leq \max\{C' \cdot q_{\text{send}}^2, q_{\text{send}}(2)\} \).

**Game GM₄:** GM₄ is the last part of the game. At this time, A attempts to decrypt the information \( \{L, M\} \) and uses the obtained information \( \{B_i, V_i, r_i, E_{\text{key}}(\cdot), P, P_{\text{pub}}, n\} \) to infer SK. Without the master key of RC, A cannot compute \( C_i = H(L_i \parallel \text{HID}_i \parallel \text{ID}_i \parallel n) \) and \( C_i = H(GP_{\text{pub}}) = H(H(A_i \parallel \text{HID}_i \parallel P_{\text{pub}})) \). According to the security of \( \Omega \) symmetric encryption algorithm, we can obtain \( |\text{Adv}^P_{\text{GM}, 4} - \text{Adv}^P_{\text{GM}}| \leq \text{Adv}^P_{\Omega}(k) \).

All queries are performed by A. After querying the test query, only the coin c of GM₄ is left. Thus, the probability of guessing coin c is \( \text{Adv}^P_{\text{GM}, 4} = 1/2 \). In summary, we can deduce

\[
\frac{1}{2} \text{Adv}^P_A = \left| \text{Adv}^P_{\text{GM}, 2} - \frac{1}{2} \right| \leq \left| \text{Adv}^P_{\text{GM}, 1} - \frac{1}{2} \right| = \left| \text{Adv}^P_{\text{GM}} - \text{Adv}^P_{\text{GM}, 4} \right|
\]

\[
\leq \left| \text{Adv}^P_{\text{GM}, 3} - \text{Adv}^P_{\text{GM}, 4} \right| + \left| \text{Adv}^P_{\text{GM}, 4} - \text{Adv}^P_{\text{GM}} \right|
\]

\[
\leq \frac{q_i^2}{2}[\text{Hash}] + \max\{C' \cdot q_{\text{send}}^2, q_{\text{send}}(2)\} + \text{Adv}^P_{\Omega}(k).
\]

Therefore, the advantage of A breaking the scheme is \( \text{Adv}^P_A \leq \left(\frac{q_i^2}{2}\right)[\text{Hash}] + 2 \max\{C' \cdot q_{\text{send}}^2, q_{\text{send}}(2)\} + 2 \text{Adv}^P_{\Omega}(k) \).

\[\square\]

5.2. Formal Security Analysis by BAN Logic. In this subsection, we demonstrate through the BAN logic that after our scheme verifies the authenticity of each other’s identity and that the determined SK will not be obtained by others. In fact, the BAN logic is a rule used to define and analyze the communication process between two parties. Specifically, the conclusions obtained by the BAN logic are through rigorous logic analysis, which further explains the confidentiality and credibility of the communication information. The notations and rules of the BAN logic used in the BAN logic calculation performed in this study are cited in [24, 27, 28, 30, 31, 36, 50, 51]. The proof of our scheme is as follows:

5.2.1. Rules

rule (1) Nonce verification rule: \( (P) \equiv \#(X), (P) \equiv Q_i \sim X \equiv (P) \equiv Q_i \equiv X \)

rule (2) Message meaning rule: \( (P) \equiv P \rightarrow Q, P \leftrightarrow X \equiv (P) \equiv Q_i \sim X \equiv (P) \equiv X \)

rule (3) Jurisdiction rule: \( (P) \equiv Q_i \equiv X, (P) \equiv Q_i \equiv X \equiv (P) \equiv X \)

rule (4) Jurisdiction rule: \( (P) \equiv \#(X) \equiv (P) \equiv \#(X, Y) \)

5.2.2. Goals

Goal 1. \( \text{U} \equiv \text{S} \equiv \text{SK} \)

Goal 2. \( \text{U} \equiv \text{S} \)

Goal 3. \( \text{S} \equiv \text{U} \equiv \text{SK} \)

Goal 4. \( \text{S} \equiv \text{U} \equiv \text{S} \)

5.2.3. Idealize the Communication Messages

\[
\begin{align*}
M_1: \text{U} & \rightarrow \text{RC}: \{D_i, \text{HID}_i', \{\text{HID}_i A_i \rightarrow \text{RC, SID}_i\}\} \rightarrow \text{C}_i \in \text{GM}_I; \\
M_2: \text{RC} & \rightarrow \text{S}: \{\text{HID}_i', R_i, Y_i, U^H(A_i C_i) \rightarrow S\}_{\text{SK}_i}; \\
M_3: \text{S} & \rightarrow \text{U}: \{R_i, \text{HID}_i', S \rightarrow \text{U}, R_i, \text{SID}_i\} \rightarrow H(A_i C_i) \rightarrow \text{SM}_i.
\end{align*}
\]

5.2.4. Initial Assumptions. \( A_i: \text{S} \equiv \text{SK} \rightarrow \text{RC} \)

\( A_2: \text{S} \equiv \#(N_i) \)

\( A_3: \text{S} \equiv \text{RC} \rightarrow \text{U}^H(A_i C_i) \rightarrow \text{S} \)

\( A_4: \text{U} \equiv \text{U}^H(A_i C_i) \rightarrow \text{S} \)

\( A_5: \text{U} \equiv \#(N_i) \substack{\text{SK}_i \rightarrow \text{S} \equiv \text{RC}} \)

\( A_6: \text{U} \equiv \text{S} \rightarrow \text{U} \equiv \text{SK}_i \rightarrow \text{S} \equiv \text{RC} \)

\( A_7: \text{S} \equiv \#(N_2) \)

\( A_8: \text{S} \equiv \text{U} \equiv \text{SK}_i \rightarrow \text{S} \equiv \text{RC} \rightarrow \text{U} \equiv \text{SK}_i \rightarrow \text{S} \)

5.2.5. The Proof of Our Proposed Scheme. For Goal 1 By M₂, we have \( S \equiv \{\text{HID}_i', R_i, Y_i, U^H(A_i C_i) \rightarrow \text{S}\} \). Based on A₁, S₁, and rule (2), we have \( S_2: \text{S} \equiv \text{SK} \rightarrow \text{RC} \sim \{\text{HID}_i', R_i, Y_i, U^H(A_i C_i) \rightarrow \text{S}\} \). Based on A₂ and rule (4), we obtain \( S_3: \text{S} \equiv \#(N_2) \equiv \{\text{HID}_i', R_i, Y_i, U^H(A_i C_i) \rightarrow \text{S}\} \). Using S₃, S₂, and rule (1), S₄: \( \text{S} \equiv \text{SK} \sim \{\text{HID}_i', R_i, Y_i, U^H(A_i C_i) \rightarrow \text{S}\} \). Subsequently, we have \( S_5: \text{S} \equiv \text{SK} \equiv U^H(A_i C_i) \rightarrow \text{S} \). Based on A₃,
Security and Communication Networks

5.3. Security Verification by ProVerif. We used the verification tool ProVerif to test the security of our proposed scheme. ProVerif is an important verification tool for verifying security fundamentals such as authentication, confidentiality, anonymity, and privacy [11, 24, 51, 52]. Furthermore, ProVerif can automatically verify the security of a scheme. It handles basic elements such as public key cryptography and the Diffie–Hellman mechanism.

The definition of the ProVerif code is shown in Figure 4. Our scheme comprised three entities: User, RC, and Server. Figures 5–7 show the user, RC, and server processes in our code, respectively. Five events were involved: UserAuthed, UserStarted, RCAcUser, ServerAcRC, and UserAcServer. Event UserAuthed means that User has been successfully authenticated. Event UserStarted means that User has started authentication. Event RCAcUser means that the RC has successfully authenticated the User. Event ServerAcRC means that Server has successfully authenticated the RC. Event UserAcServer means that User has successfully authenticated Server.

Next, we used ProVerif to query whether the attacker can obtain the identities of User and Server as well as the SK and whether the events above were executed in sequence. Figure 8 shows the events and queries in the code.

Finally, we executed the code to perform authentication, and the results are shown in Figure 9. The result shows that ProVerif confirmed the security of our scheme. Therefore, the attacker cannot obtain parameters \( \{ SK_1, SK_2, ID_1, SID_1 \} \),\( \{ SK_1, SK_2, ID_2, SID_2 \} \),\( \{ SK_1, SK_2, ID_3, SID_3 \} \), and rule (3), we have \( S_9: S = U^{SK_1} = U \rightarrow S, R_1, SID_3 \).

Based on \( A_9, S_9 \), rule (4), and rule (1), we obtain \( S_9: U \rightarrow S \).

For \( \text{Goal} 2 \), based on \( A_6, S_10 \), and rule (3), we have \( S_{10}: U \rightarrow S. \) (\( \text{Goal} 1 \)).

For \( \text{Goal} 3 \), based on \( M_3, S_{11} \), we have \( S_{11}: U \rightarrow S. \) (\( \text{Goal} 2 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{15}: U \rightarrow S. \) (\( \text{Goal} 3 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{16}: U \rightarrow S. \) (\( \text{Goal} 4 \)).

5.3. Security Verification by ProVerif. We used the verification tool ProVerif to test the security of our proposed scheme. ProVerif is an important verification tool for verifying security fundamentals such as authentication, confidentiality, anonymity, and privacy [11, 24, 51, 52]. Furthermore, ProVerif can automatically verify the security of a scheme. It handles basic elements such as public key cryptography and the Diffie–Hellman mechanism.

The definition of the ProVerif code is shown in Figure 4. Our scheme comprised three entities: User, RC, and Server. Figures 5–7 show the user, RC, and server processes in our code, respectively. Five events were involved: UserAuthed, UserStarted, RCAcUser, ServerAcRC, and UserAcServer. Event UserAuthed means that User has been successfully authenticated. Event UserStarted means that User has started authentication. Event RCAcUser means that the RC has successfully authenticated the User. Event ServerAcRC means that Server has successfully authenticated the RC. Event UserAcServer means that User has successfully authenticated Server.

Next, we used ProVerif to query whether the attacker can obtain the identities of User and Server as well as the SK and whether the events above were executed in sequence. Figure 8 shows the events and queries in the code.

Finally, we executed the code to perform authentication, and the results are shown in Figure 9. The result shows that ProVerif confirmed the security of our scheme. Therefore, the attacker cannot obtain parameters \( \{ SK_1, SK_2, ID_1, SID_1 \} \),\( \{ SK_1, SK_2, ID_2, SID_2 \} \),\( \{ SK_1, SK_2, ID_3, SID_3 \} \), and rule (3), we have \( S_9: S = U^{SK_1} = U \rightarrow S, R_1, SID_3 \).

Based on \( A_9, S_9 \), rule (4), and rule (1), we obtain \( S_9: U \rightarrow S \).

For \( \text{Goal} 2 \), based on \( A_6, S_10 \), and rule (3), we have \( S_{10}: U \rightarrow S. \) (\( \text{Goal} 1 \)).

For \( \text{Goal} 3 \), based on \( M_3, S_{11} \), we have \( S_{11}: U \rightarrow S. \) (\( \text{Goal} 2 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{15}: U \rightarrow S. \) (\( \text{Goal} 3 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{16}: U \rightarrow S. \) (\( \text{Goal} 4 \)).

5.3. Security Verification by ProVerif. We used the verification tool ProVerif to test the security of our proposed scheme. ProVerif is an important verification tool for verifying security fundamentals such as authentication, confidentiality, anonymity, and privacy [11, 24, 51, 52]. Furthermore, ProVerif can automatically verify the security of a scheme. It handles basic elements such as public key cryptography and the Diffie–Hellman mechanism.

The definition of the ProVerif code is shown in Figure 4. Our scheme comprised three entities: User, RC, and Server. Figures 5–7 show the user, RC, and server processes in our code, respectively. Five events were involved: UserAuthed, UserStarted, RCAcUser, ServerAcRC, and UserAcServer. Event UserAuthed means that User has been successfully authenticated. Event UserStarted means that User has started authentication. Event RCAcUser means that the RC has successfully authenticated the User. Event ServerAcRC means that Server has successfully authenticated the RC. Event UserAcServer means that User has successfully authenticated Server.

Next, we used ProVerif to query whether the attacker can obtain the identities of User and Server as well as the SK and whether the events above were executed in sequence. Figure 8 shows the events and queries in the code.

Finally, we executed the code to perform authentication, and the results are shown in Figure 9. The result shows that ProVerif confirmed the security of our scheme. Therefore, the attacker cannot obtain parameters \( \{ SK_1, SK_2, ID_1, SID_1 \} \),\( \{ SK_1, SK_2, ID_2, SID_2 \} \),\( \{ SK_1, SK_2, ID_3, SID_3 \} \), and rule (3), we have \( S_9: S = U^{SK_1} = U \rightarrow S, R_1, SID_3 \).

Based on \( A_9, S_9 \), rule (4), and rule (1), we obtain \( S_9: U \rightarrow S \).

For \( \text{Goal} 2 \), based on \( A_6, S_10 \), and rule (3), we have \( S_{10}: U \rightarrow S. \) (\( \text{Goal} 1 \)).

For \( \text{Goal} 3 \), based on \( M_3, S_{11} \), we have \( S_{11}: U \rightarrow S. \) (\( \text{Goal} 2 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{15}: U \rightarrow S. \) (\( \text{Goal} 3 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{16}: U \rightarrow S. \) (\( \text{Goal} 4 \)).

5.3. Security Verification by ProVerif. We used the verification tool ProVerif to test the security of our proposed scheme. ProVerif is an important verification tool for verifying security fundamentals such as authentication, confidentiality, anonymity, and privacy [11, 24, 51, 52]. Furthermore, ProVerif can automatically verify the security of a scheme. It handles basic elements such as public key cryptography and the Diffie–Hellman mechanism.

The definition of the ProVerif code is shown in Figure 4. Our scheme comprised three entities: User, RC, and Server. Figures 5–7 show the user, RC, and server processes in our code, respectively. Five events were involved: UserAuthed, UserStarted, RCAcUser, ServerAcRC, and UserAcServer. Event UserAuthed means that User has been successfully authenticated. Event UserStarted means that User has started authentication. Event RCAcUser means that the RC has successfully authenticated the User. Event ServerAcRC means that Server has successfully authenticated the RC. Event UserAcServer means that User has successfully authenticated Server.

Next, we used ProVerif to query whether the attacker can obtain the identities of User and Server as well as the SK and whether the events above were executed in sequence. Figure 8 shows the events and queries in the code.

Finally, we executed the code to perform authentication, and the results are shown in Figure 9. The result shows that ProVerif confirmed the security of our scheme. Therefore, the attacker cannot obtain parameters \( \{ SK_1, SK_2, ID_1, SID_1 \} \),\( \{ SK_1, SK_2, ID_2, SID_2 \} \),\( \{ SK_1, SK_2, ID_3, SID_3 \} \), and rule (3), we have \( S_9: S = U^{SK_1} = U \rightarrow S, R_1, SID_3 \).

Based on \( A_9, S_9 \), rule (4), and rule (1), we obtain \( S_9: U \rightarrow S \).

For \( \text{Goal} 2 \), based on \( A_6, S_10 \), and rule (3), we have \( S_{10}: U \rightarrow S. \) (\( \text{Goal} 1 \)).

For \( \text{Goal} 3 \), based on \( M_3, S_{11} \), we have \( S_{11}: U \rightarrow S. \) (\( \text{Goal} 2 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{15}: U \rightarrow S. \) (\( \text{Goal} 3 \)).

For \( \text{Goal} 4 \), based on \( A_8, S_{15} \), and rule (3), we have \( S_{16}: U \rightarrow S. \) (\( \text{Goal} 4 \)).
and all events are executed normally. Note that Figures 4–9 are shown in Appendix.

5.4. Informal Security Analysis

5.4.1. Known Session-Specific Temporary Information Attacks. Upon completing the login and authentication phase, if \( N_1 \) or \( N_2 \) is compromised, then \( A \) intercepts information \( \{ R_1, F_1 \} \) and computes \( E_i = N_1 R_1 \), but it cannot compute \( A_1 = H(\pi | HID_1 | ID_1 | nd) \) and \( C_i = H(G, P_{pub}) = H(H(N_1 \oplus A_1 | HID_1 | P_{pub}) \). Therefore, \( A \) cannot compute the SK, and the scheme successfully overcomes known session-specific temporary information attacks.

5.4.2. User Impersonation Attacks. Assume that the \( A \) pretends to be a user and forges a message \( M_1 = \{ D_1, HID_1, L_1 \} \). Even if \( A \) forges a random number \( N_1 \), it cannot compute \( A_1 \) to forge \( D_1 \) and \( L_1 \). \( A \) cannot obtain \( A_1 \) for two reasons. First, upon completing the login and entering the authentication phase, \( A \) is encrypted by \( C_1 \), and \( A \) cannot compute \( C_1 \) to decrypt \( A_1 \); therefore, \( A_1 \) cannot be obtained. Second, in the registration phase, if the SC is stolen by a malicious user, then \( A \) can obtain \( B_i \). However, because \( A_1 = B_i \oplus P_i \), \( A \) requires \( \{ PW_i, b_i, r_i \} \) to compute \( P_i \), which is impossible. Therefore, the scheme successfully overcomes user impersonation attacks.

5.4.3. Server Impersonation Attacks. Assume that \( A \) pretends to be the server and forges a message \( M_3 = \{ R_3, F_3 \} \). Therefore, \( A \) must generate a random number \( N_A \) and compute \( \{ R_A = N_A P_i, E_A = N_A R_i, SK_A = H(E_A || H(T_A || C_i)) \} \). However, \( A \) cannot obtain \( \{ R_i, A_i, C_i \} \). Even if \( A \) can obtain temporary information \( N_1 \), it cannot compute \( \{ R_i, A_i, C_i \} \), nor can it obtain sensitive information by
5.4.4. Man-in-the-Middle Attacks. Upon completing the login and authentication phase, the A intercepts the messages transmitted between User and Server, to impersonate the user or server. The A may intercept $M_3$ to impersonate Server. However, A cannot compute $F_1 = H(\text{HID}\|\text{SK}\|\text{RS}\|\text{SID})$; therefore, the session is terminated. In another case, A may intercept $\{M_1, M_2\}$ to impersonate User. However, A cannot compute $A_i$; therefore, it cannot pass the RC verification. Therefore, the scheme can overcome man-in-the-middle attacks.

5.4.5. Replay Attacks. Suppose that message $M_1$, $M_2$, or $M_3$ is replayed by A. However, our scheme overcomes this attack by refreshing random numbers $\{N_1, N_2\}$. By replaying one of the messages $\{M_1, M_2, M_3\}$, the mutual authentication values $F_i$ for the user will not pass, and the session will be terminated. Therefore, this scheme can overcome replay attacks.

5.4.6. Stolen SC Attacks. Suppose that the SC is stolen by a malicious user A who will obtain $\{B_i, V_i, E_{key}(\cdot), P, P_{pub}, n, r_i\}$. However, based on those values, A cannot compute $A_i = H(x\|\text{HID}\|\text{ID}\|\text{nd})$. In addition, A cannot obtain $N_1$ to compute $C_i = H(GP_{pub}) = H(H(N_1@ A\@\text{HID})P_{pub})$ and $E_i = N_1R_S$. Therefore, A cannot compute $SK = H(E_i\|H(A_i\|C_i))$. Hence, it is clear that the scheme can successfully overcome stolen SC attacks.

5.4.7. Offline Password-Guessing Attacks. According to A, obtains $\{B_i, V_i, E_{key}(\cdot), P, P_{pub}, n, r_i\}$. Moreover, A can be biometric $b_i$ by shoulder surfing. A launches an offline password-guessing attack by comparing $P_i = H(PW\|h(b_i))$. In addition, $P_i = A_i@B_i = H(x\|\text{HID}\|\text{ID}\|\text{nd})\|B_i$. However, A cannot obtain HID, $x$, and $d_i$; therefore, the attacker cannot compute $P_i$. Hence, the scheme can overcome offline password-guessing attacks.

5.4.8. Privileged Insider Attacks. Assume that the privileged insider A is $\{\text{HID}, \text{ID}, d_i\}$ stored in the RC database. However, A cannot obtain $x$ and the user’s ID; therefore, it cannot compute $A_i = H(x\|\text{HID}\|\text{ID}\|\text{nd})$. Because $E_i = N_2R_i = N_2H(N_1@ A\@\text{HID})P$ and SK $= H(E_i\|H(A_i\|C_i))$, A cannot compute the SK. Therefore, the scheme can overcome privileged insider attacks.

5.4.9. Perfect Forward Secrecy. Suppose that A obtains the RC’s long-term key $x$ and attempts to obtain the SK. If A obtains $N_1$ and intercepts $\{R_i, F_i\}$, then it computes $E_i = N_1R_S$. However, A cannot compute $A_i = H(x\|\text{HID}\|\text{ID}\|\text{nd})$ and $C_i = H(GP_{pub}) = H(H(N_1@ A\@\text{HID})P_{pub})$. In other words, A cannot compute $SK = H(E_i\|H(A_i\|C_i))$. Therefore, this scheme provides perfect forward secrecy.

5.4.10. User Anonymity. In the registration phase of the improved scheme, User, computes HID, to protect the real identity of the user. In the authentication phase, the user transmits the virtual identity HID, and the attacker cannot obtain the real identity of the user. Therefore, our scheme provides user anonymity.

5.4.11. Three-Factor Secrecy. The three factors refer to the password, SC, and biometrics. Based on a previous analysis, $A_i$ and $C_i$ are the key parameters for launching an attack to compute the SK. A obtains two of the three factors, i.e., the password and SC. Even if A obtains the password and extracts the parameters from the SC, it cannot compute $A_i$ and $C_i$ to perform any attack. Passwords and biometrics: if A obtains the password and biometrics to calculate $A_i$, it must obtain $B_i$ and $P_i$. However, $B_i$ is stored in an SC, whereas $P_i$ is protected by a random number. Biometrics and smart cards: if A obtains the biometrics and SC to calculate $P_i$, it must obtain the PW. Therefore, A cannot compute $A_i = B_i@P_i$.

After analyzing the security of our improved scheme, we can conclude that our proposed scheme is “provably secure” against several well-known attacks with a higher probability. However, it does not mean that our scheme is a “perfectly secure” authentication scheme because many special attack approaches or tricks exist [19].
improved scheme overcame all known attacks. However, our scheme was subject to known session-specific temporary information, user impersonation, and server impersonation attacks. Nonetheless, our scheme utilized in a practical application, the 13.5 ms difference was significantly lower than [34, 36] and the computation cost is 1600 bits. In our scheme, the messages in the login and authentication phases were hash values, and \{DID, HID, F_i\} are ciphertexts. The total communication cost of Ali et al.'s scheme was 3712 bits. In Wang et al.'s scheme, the messages in the login and authentication phases were \{R_i, L_i, M_i\}, \{R_{i0}, F_i\}, and \{Q_i\}, where \{R_i, R_{i0}\} belong to ECC, \{F_i, Q_i\} are hash values, and \{L_i, M_i\} are ciphertexts. The total communication cost of Wang et al.'s scheme was 1664 bits. In our scheme, the messages in the login and authentication phases were \{D_0, HID_0, L_0, M_0\}, \{R_{i0}, F_i\}, and \{Q_i\}, where \{R_{i0}, R_i\} belong to ECC, \{D_0, HID_0, F_0\} are hash values, and \{L_0, M_0\} are ciphertexts. The total communication cost of our scheme was 1600 bits. Through the analysis of computation cost and communication cost, the communication cost of our scheme is significantly lower than [34, 36] and the computation cost is also acceptable. Combined with the previous security

### Table 3: Security comparison.

<table>
<thead>
<tr>
<th>Attack methods</th>
<th>Ali and Pal’s scheme [34]</th>
<th>Wang et al.’s scheme [36]</th>
<th>Our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>User anonymity</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Offline password attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Stolen smart card attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Known session-specific temporary information attack</td>
<td>× [36]</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>User impersonation attack</td>
<td>× [36]</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Server impersonation attack</td>
<td>× [36]</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Replay attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Perfect forward secrecy</td>
<td>× [36]</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Three-factor secrecy</td>
<td>× [36]</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Note. ✓, able to overcome the attack; ×, unable to overcome the attack.

### Table 4: Computation cost comparison.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>User computations</th>
<th>Server computations</th>
<th>RC</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ali et al.’s scheme [34]</td>
<td>$3T_m + 4T_p \approx 42.42\text{ ms}$</td>
<td>$4T_m + 3T_p + T_s \approx 55.56\text{ ms}$</td>
<td>$3T_m + 3T_p + 2T_s \approx 42.18\text{ ms}$</td>
<td>$10T_m + 10T_p + 3T_s \approx 140.16\text{ ms}$</td>
</tr>
<tr>
<td>Wang et al.’s scheme [36]</td>
<td>$3T_m + 1T_s \approx 40.62\text{ ms}$</td>
<td>$2T_m + 1T_s \approx 27.12\text{ ms}$</td>
<td>$1T_m + 2T_s \approx 13.74\text{ ms}$</td>
<td>$6T_m + 4T_s \approx 71.01\text{ ms}$</td>
</tr>
<tr>
<td>Our scheme</td>
<td>$3T_m + 1T_s \approx 40.62\text{ ms}$</td>
<td>$2T_m + 1T_s \approx 27.12\text{ ms}$</td>
<td>$2T_m + 2T_s \approx 27.24\text{ ms}$</td>
<td>$7T_m + 4T_s \approx 94.98\text{ ms}$</td>
</tr>
</tbody>
</table>

$T_m$, time for executing elliptic curve scalar point multiplication. $T_p$, time for performing elliptic curve point addition operation. $T_s$, time for executing symmetric encryption/decryption operation.

### Table 5: Comparison of communication and massage rounds.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Communication cost (bits)</th>
<th>Massage rounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ali and Pal’s scheme [34]</td>
<td>3712</td>
<td>4</td>
</tr>
<tr>
<td>Wang et al.’s scheme [36]</td>
<td>1664</td>
<td>4</td>
</tr>
<tr>
<td>Our scheme</td>
<td>1600</td>
<td>3</td>
</tr>
</tbody>
</table>

6. Performance Comparison

In this section, we compare our improved scheme with those of Ali and Pal [34] and Wang et al. [36] in terms of security and efficiency. Table 3 presents a comparison of security among the abovementioned schemes. It is evident that our scheme is secure against well-known attacks. Ali and Pal’s scheme [34] could not overcome known session-specific temporary information, user impersonation, and server impersonation attacks, nor could it provide three-factor and perfect forward secrecy. Although Wang et al.’s scheme [36] guaranteed perfect forward secrecy, it could not overcome known session-specific temporary information, user impersonation, and server impersonation attacks. Hence, it is clear that only our proposed protocol successfully overcame all known attacks and achieved a certain degree of security.

A comparison of the computational costs is shown in Table 4. We used JPBC-2.0.0 (Pairing-Based Cryptography Library) [53], IntelliJ IDEA 2020.2.1 community edition, and a Windows 10 computer with a 2.3 GHz Intel (R) Core i5 processor and 16 GB of memory to simulate the computational costs. It is noteworthy that a widely accepted Type A pairing was constructed on the curve $y^2 = x^3 + x$ over $F_{q^r}$ where $q$ is a prime satisfying $q = 3 \mod 4$. In our experimental results, $T_m$ was 13.5 ms, $T_p$ was 0.48 ms, and $T_s$ was 0.12 ms. As shown in Table 4, the computational cost of our scheme was lower than that of the scheme in [34], whereas it was 13.5 ms higher than that of the scheme in [36]. However, when our scheme was utilized in a practical application, the 13.5 ms difference was almost negligible. Meanwhile, the scheme in [36] was subject to known session-specific temporary information, user impersonation, and server impersonation attacks. However, our improved scheme overcame all known attacks.

Table 5 shows a comparison of the communication costs. We assumed that the ECC points accounted for 320 bits because two 160-bit parameters form an ECC point. The hash operation was considered to be 256 bits, and the identity was 64 bits. The length of the ciphertext for a symmetric encryption was 256 bits. In Ali et al.’s scheme, the messages in the login and authentication phase were $\{\text{DID}_i, \text{E}_i, \text{C}_i, \text{D}_i\}, \{\text{DID}_{\text{new}}^\text{rev}, \text{K}_i, \text{L}_i, \text{F}_i\}, \{\text{DID}_{\text{new}}^\text{rev}, \text{Q}_i, \text{M}_i, \text{K}_i\}$, and $\{Z_i\}$, where $\{\text{E}_i, \text{C}_i, \text{K}_i, \text{Q}_i, Z_i\}$ belong to ECC, $\{D_0, L_0, M_0\}$ are hash values, and $\{\text{DID}_0, \text{DID}_{\text{new}}^\text{rev}, \text{F}_i\}$ are ciphertexts. The total communication cost of Ali et al.’s scheme was 3712 bits. In Wang et al.’s scheme, the messages in the login and authentication phases were $\{\text{R}_i, \text{L}_i, \{\text{M}_i\}, \{\text{R}_{i0}, \text{F}_i\}, \{\text{Q}_i\}$, where $\{\text{R}_i, \text{R}_{i0}\}$ belong to ECC, $\{\text{F}_i, \text{Q}_i\}$ are hash values, and $\{\text{L}_0, \text{M}_i\}$ are ciphertexts. The total communication cost of Wang et al.’s scheme was 1664 bits. In our scheme, the messages in the login and authentication phases were $\{\text{D}_0, \text{HID}_0, \text{L}_0, \{\text{M}_0\}$, $\{\text{R}_{i0}, \text{F}_i\}$, and $\{\text{Q}_i\}$, where $\{\text{R}_{i0}, \text{R}_i\}$ belong to ECC, $\{\text{D}_0, \text{HID}_0, \text{F}_0\}$ are hash values, and $\{\text{L}_0, \text{M}_0\}$ are ciphertexts. The total communication cost of our scheme was 1600 bits.

Through the analysis of computation cost and communication cost, the communication cost of our scheme is significantly lower than [34, 36] and the computation cost is also acceptable. Combined with the previous security
analysis mentioned in Table 3, our scheme also has strong security. Hence, our scheme is worthy of being adopted in secure three-factor authentication.

7. Conclusion

In this study, we performed a security analysis of Wang et al.’s scheme and discovered that their scheme could not overcome known session-specific temporary information, user impersonation, and server impersonation attacks. Additionally, we have proven the security of our proposed scheme through formal and informal security analysis. Subsequently, the communication security of our scheme was validated by the ProVerif tool, and the BAN logic indicated that mutual authentication can be completed safely. Finally, through a comparison of performance and security, the security and efficiency of our proposed scheme was proven. However, the computational cost of our scheme is still high. It will lead us to design lightweight authentication schemes in the future.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

References


