IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 4, 2020, accepted January 15, 2020, date of publication January 28, 2020, date of current version February 14, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2969986*

# An Authenticated Key Exchange Protocol for Multi-Server Architecture in 5G Networks

**TSU-YANG WU[1], ZHIYUAN LEE[1], MOHAMMAD S. OBAIDAT[2,3,4], (Life Fellow, IEEE), SARU KUMARI[5], SACHIN KUMAR[6], AND CHIEN-MING CHEN[7]**

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
[2]College of Computing and Informatics, University of Sharjah, Sharjah 27272, UAE
[3]KASIT, University of Jordan, Amman 11942, Jordan
[4]School of Communication and Computing, University of Science and Technology Beijing (USTB), Beijing 100083, China
[5]Department of Mathematics, Chaudhary Charan Singh University, Meerut 250004, India
[6]Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad 201009, India
[7]School of Computer Science, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China

Corresponding authors: Saru Kumari (saryusiirohi@gmail.com) and Chien-Ming Chen (chienmingchen@ieee.org)

This work was supported in part by the Shenzhen Technical Project under Grant KQJSCX20170327161755607.

**ABSTRACT** Currently, the popularity of the Internet of Things (IoT) has brought about an increase in the amount of data, so multi-server distributed cloud computing has been widely used in various applications that have brought convenience to our daily lives. At the same time, the development of the fifth generation (5G) of mobile communication technology has gradually become the main driving force for the popularization of the IoT. Because the 5G network is a heterogeneous network with multiple servers and small cells, the mutual authentication protocol under multiple servers is also applicable to the 5G network environment. However, much of the data will have serious storage and security issues during transmission. Aiming at the security issues in a multi-server (M-S) architecture, in 2018, Wu *et al.* proposed an authentication protocol in a distributed cloud environment. They claimed that their protocol is secure and resistant to various known types of attacks. However, we found that their protocol does not guarantee perfect forward secrecy (PFS) and suffers from privileged insider (PI) attacks. Such attacks will cause data to be out of sync. Therefore, we improved Wu *et al.*'s protocol and proposed an improvement in the 5G network environment. Finally, we performed a security analysis on the proposed protocol, including the automatic encryption protocol tool ProVerif, BAN logic, and informal security analysis, which proved that our protocol is secure. Compared with similar existing schemes, we have proved the efficiency of the scheme and achieved higher security standards.

**INDEX TERMS** Authentication, multi-server, 5G networks, cryptanalysis, lightweight.

## I. INTRODUCTION

Today, the development of fifth generation (5G) technology has increasingly attracted researchers' interest. The development of 5G technology has become the main driving force for the growth of Internet-of-Things (IoT) related applications [1]. Future IoT applications will require new performance standards in areas such as security [2]–[8], big dada [9], [10], reliability, low latency, artificial intelligence [11]–[13], and wireless network coverage [14]–[16], which are applicable to many IoT devices. Additionally, 5G has higher energy efficiency requirements in these aspects than 4G, so many current single-server structures are not suitable for 5G networks. Then some scholars proposed the use of a multi-server architecture in a 5G network environment [17], [18]. The IoT connects objects all over the world to the Internet, such as in the military field, intelligent transportation, and smart homes. During the use of these objects, sensors installed on these objects collect data and transmit the data to other smart devices. People can get the data they need through certain devices. Therefore, the use of the IoT brings large amounts of data to people, and we must face how to protect the data. To solve this problem, cloud computing technology was introduced as a key technology for storing data on distributed cloud servers instead of local hosts. This technology introduces a control server that can control multiple private cloud services, and these private cloud servers are organized in a distributed manner (see Fig. 1).

The associate editor coordinating the review of this manuscript and approving it for publication was Isaac Woungang.

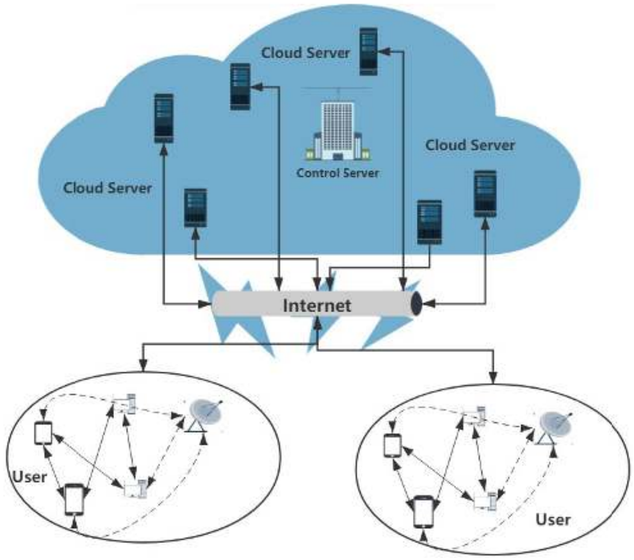


**FIGURE 1.** Distributed cloud computing environments in 5G networks.

Cloud computing is the storage and management of data. Today, cloud computing technology is relatively mature and widely used. In the multi-server architecture of a 5G network, the authentication process involves three entities. The first is users, who support mmWave technology and device-to-device technology and can use these technologies to access the server. The smart devices they use contain smart cards issued by the control server and private data accumulated by sensors. These smart devices have limited computing power. The second is a cloud server that can communicate with and provide services to users. There are many cloud servers in the entire system. The last one is the control server, which stores registration information for users and cloud servers to help both authenticate and generate session keys.

However, the IoT environment is fragile and vulnerable to unforeseen circumstances such as unexpected power outages and network disruptions. Much of the information transmitted in the IoT network is private and sensitive. How we ensure the security of this type of data is critical. In response to this problem, researchers have proposed numerous authentication schemes. Considering the computing power and service life of IoT devices, it is reasonable to design some low-energy and lightweight authentication protocols.

Because many IoT devices have limited computing and storage capabilities, we propose a secure, lightweight authentication scheme for distributed cloud computing environments that uses only hash functions and XOR operations. Authentication takes place between remote objects during communication. Lamport [19] first proposed an authentication mechanism using password over insecure networks in 1981. However, this protocol has some security problems, such as dependence on password tables, and high hash overhead. Later, researchers presented various improvements to the security issues that emerged in Lamport *et al.*'s protocol. Some of the early improvements [20]–[22] to the authentication scheme were to fix the vulnerabilities in [19]. Later, to improve the security of remote communication, researchers used other security factors based on traditional passwords. In 2001, Chang and Wu [23] and Hwang *et al.* [24] introduced smart card solutions. A series of smart-card-based authentication schemes were subsequently proposed [25]–[28]. Li *et al.* [29] first proposed using the neural network schemes for identity authentication in a M-S environment. Later, due to the inefficiency and insecurity of the Li *et al.* scheme, many researchers have made improvements to the authentication method [30]–[32]. Additionally, some protocols have begun to use biometrics to ensure security [33].

Because 5G networks are heterogeneous, users will have frequent authentication to prevent the various attacks. In addition, due to the limitation of computing resources in IoT systems, more efficient authentication and key exchange protocols need to be developed for complex M-S 5G networks [34]. M-S authentication protocols have been widely proposed in [35]–[43]. Recently, Wu *et al.* [44] proposed an authentication protocol for a distributed cloud environment. Their protocol is claimed to resist off-line password guessing (OPG) attacks, PI attacks, desynchronization attacks, forgery attacks, and user tracking attacks. In Wu *et al.*'s paper, it was mentioned that the protocols of Irshad *et al.* [43] and Amin *et al.* [45] had security issues. Irshad *et al.*'s protocol is vulnerable to PI attacks and cannot guarantee user anonymity (UA). Amin *et al.*'s protocol does not guarantee UA and is subject to OPG attacks.

The above discussion shows that designing the AKE protocol for a distributed cloud computing network to meet security requirements is a serious task. All existing solutions are neither resistant to all known attacks, nor can they guarantee the consumption of their own calculations. In this paper, we concentrate on analyzing the security of [44] and point out that their protocol fails to resist stolen smart card (SSC) attacks and PI attacks, and cannot provide pre-verification and perfect forward secrecy (PFS). To overcome the limitations, we propose an enhanced protocol based on the Wu *et al.*'s protocol for the multi-server architecture in the 5G IoT environment. In addition, we prove that the protocol provides a variety of security functions, including PFS and resistance to privileged internal attacks, stolen smart card attacks, etc. We use the ProVerif tool, BAN (Burrows-Abadi-Needham) logic, and informal security analysis to prove the security. Finally, we provide comparisons of various related schemes.

The rest of this paper is organized as follows: In Section 2, we briefly introduce the scheme of Wu *et al.* Cryptanalysis of the same scheme is given in Section 3. In Section 4, we present the details of the proposed protocol. Section 5 is mainly a discussion of ProVerif, BAN logic analysis, and informal security analysis. Security and performance comparisons are given in Section 6. Finally, in Section 7, we give the conclusion of this article.

**TABLE 1. Notations and their meanings.**

| Notations | Meanings |
|---|---|
| $U_i$ | The $i$th user |
| $ID_i$ | $U_i$'s identity |
| $PW_i$ | $U_i$'s password |
| $ID_{SC}$ | Smart card's identity |
| $x$ | The secret key of CS |
| $SID_j$ | The identity (ID) of $S_j$ |
| $PID_i$ | The pseudo-ID of $U_i$ |
| $PSID_j$ | The pseudo-ID of $S_j$ |
| $SK_C, SK_S, SK_U$ | Session keys produced by $CS, S_j, U_i$ |
| $\mathcal{A}$ | The attacker |

## II. SECURITY ANALYSIS OF WU et al's PROTOCOL

### A. REVIEW OF WU et al.'s PROTOCOL

In the section, we briefly introduce Wu *et al.*'s protocol [44]. Their protocol consists of user and server registration, authentication, and password change phases. It requires the use of secure channels in the registration phases and public channels in the second and third phases. Data transmitted over a public channel can be stolen, forged, or modified. In their protocol, there exist three roles: user $U_i$, cloud server $S_j$, and control server $CS$. The notation used in this paper is presented in Table 1. Because the security analysis does not involve the password update phase, our review of Wu *et al.* consists of only the registration and the authentication phases.

#### 1) REGISTRATION

$U_i$ registers with $CS$ by executing the following steps:

1) $U_i$ selects $ID_i$, $PW_i$, and $b_i$ to compute $HP_i = h(PW_i \| b_i)$. Then, it sends $ID_i$ and $HP_i$ to $CS$ over the secure channel.
2) $CS$ generates a pseudo-identity $PID_i$ for $U_i$ and computes $D_1$, $D_2$. Then $CS$ stores $(PID_i, D_1, D_2)$ into a smart card (SC) and sends the SC to $U_i$, where $x$ is $CS$'s secret key, $ID_{SC}$ is an identity of the smart card, and $D_1 = h(PID_i \| x) \oplus H(ID_i \| HP_i)$, $D_2 = h(ID_i \| ID_{SC}) \oplus HP_i$.
3) After receiving the SC, $U_i$ computes $D_3 = b_i \oplus h(ID_i \| b_i)$ and stores it into the smart card.

$S_j$ registers with $CS$ by executing the following steps:

1) $S_j$ selects its identity $SID_j$ and sends it to $CS$. Then, $CS$ stores $SID_j$ and generates a pseudo identity $PSID_j$ for $S_j$.
2) Finally, $CS$ sends $(PSID_j, C_1)$ to $S_j$ via a secure channel, where $C_1 = h(PSID_j \| x)$.
3) On receiving the message from $CS$, $S_j$ stores this message into its database.

#### 2) AUTHENTICATION

When user $U_i$ wants to access the service of some cloud server $S_j$, $CS$ can help to establish a session key for communication. The detailed procedures are described as follows.

1) User $U_i$ inputs $ID_i$ and $PW_i$ and computes $b_i = D_3 \oplus h(ID_i \| PW_i)$ and $HP_i = h(PW_i \| b_i)$. Then, $U_i$ selects a random value $N_i$, $SID_j$ to compute $B_1 = D_1 \oplus h(ID_i \| HP_i)$, $B_2 = B_1 \oplus N_i$, $B_3 = h(PID_i \| b_i) \oplus ID_i$, $B_4 = D_2 \oplus h(b_i \| PID_i) \oplus SID_j$, $B_5 = h(b_i \| PID_i \| ID_i \| SID_j)$. Finally, $U_i$ sends $M_1 = \{PID_i, B_2, B_3, B_4, B_5\}$.
2) Upon receiving $M_1$, $S_j$ selects a random value $N_j$ and computes $B_6 = C_1 \oplus N_j$, $B_7 = h(PSID_i \| N_j) \oplus SID_j$, $B_8 = h(N_j \| SID_j \| SID_j)$. Then, $S_j$ sends $M_2 = \{M_1, PSID_j, B_6, B_7, B_8\}$ to $CS$.
3) Upon receiving $M_2$, $CS$ recovers $N_i = B_2 \oplus h(PID_i \| x)$, $ID_i = B_3 \oplus h(PID_i \| N_i)$, $SID_j^U = B_4 \oplus h(N_i \| PID_i) \oplus h(ID_i \| x \| ID_S C)$. Then, CS verifies $ID_i$, $SID_j^U$, and $B_5 = h(N_i \| PID_i \| ID_i \| SID_j^U)$. If the verifications do not hold, $CS$ reject the request.
4) $CS$ recovers $N_j = B_6 \oplus h(PSID_j \| x)$ and $SID_j^S = B_7 \oplus h(PSID_j \| N_j)$. Then, it verifies $SID_j^U = SID_j^S$ and $B_8 = h(N_j \| SID_j^S \| B_5)$.
5) $CS$ generates $N_c$, $PID_i^{new}$, $PSID_j^{new}$ and computes $B_9 = N_j \oplus N_c \oplus h(N_i \| ID_i)$, $B_{10} = h((N_j \oplus N_c) \| N_i) \oplus PID_i^{new}$, $B_{11} = h(PID_i^{new} \| x) \oplus h(N_i \| (N_j \oplus N_c))$, $SK_c = h(N_i \oplus N_j \oplus N_c)$, $B_{12} = h(PID_i^{new} \| h(PID_i^{new} \| x) \| SK_c)$, $B_{13} = N_i \oplus N_c \oplus h(N_j \| SID_j)$, $B_{14} = h((N_i \oplus N_c) \| N_j) \oplus PSID_j^{new}$, $B_{15} = h(PSID_j^{new} \| x) \oplus h(N_j \| (N_i \oplus N_c))$, $B_{16} = h(PSID_j^{new} \| h(PSID_j^{new} \| x) \| SK_c)$. Then, $CS$ sends $M_3 = \{B_9, B_{10}, \ldots, B_{16}\}$ to $S_j$.
6) Upon receiving $M_3$, $S_j$ recovers $N_i \oplus N_c = B_{13} \oplus h(N_j \| SID_j)$, $PSID_j^{new} = B_{14} \oplus h((N_i \oplus N_c) \| N_j)$, $C_1^{new} = B_{15} \oplus h(N_j \| (N_i \oplus N_c))$. Then, it computes $SK_S = h(N_i \oplus N_c \oplus N_j)$ and verifies $B_{16} = h(PSID_j^{new} \| h(C_1^{new} \| x) \| SK_S)$. If the verifications do not hold, $S_j$ terminates. Finally, $S_j$ sends $M_4 = \{B_9, B_{10}, B_{11}, B_{12}\}$ to $U_i$.
7) Upon receiving $M_4$, $U_i$ recovers $N_j \oplus N_c = B_9 \oplus h(N_i \| ID_i)$, $PID_i^{new} = B_{10} \oplus h((N_j \oplus N_c) \| N_i)$, $B_1^{new} = B_{11} \oplus h(N_i \| (N_j \oplus N_c))$. Then, it computes $SK_U = h(N_i \oplus N_j \oplus N_c)$ and verifies $B_{12} = h(PID_i^{new} \| B_1^{new} \| SK_U)$. If the verifications do not hold, $U_i$ terminates.

### B. CRYPTANALYSIS OF WU et al.'s PROTOCOL

This section discusses the cryptanalysis of Wu *et al.*'s protocol. We analyze the security and design flaws, which is described in the following subsections.

#### 1) PERFECT FORWARD SECRECY (PFS)

In this section, we demonstrate that Wu *et al.*'s protocol did not provide PFS, an important security requirement in authenticated key agreement protocols, under some assumptions.

Assume that adversary $\mathcal{A}$ can obtain $\{D_1, D_2, D_3\}$, the information of $U_i$'s SC and $CS$'s secret key $x$. Meanwhile, $\mathcal{A}$ can capture messages $\left\{PID_i, PSID_j, B_2, B_3, \ldots, B_{16}\right\}$ for each session in which $U_i$ wants to access the service of $S_j$. The established session $SK$ can be derived by $\mathcal{A}$ according to the following steps:

1) Recover $N_i = B_2 \oplus h(PID_i \| x)$
2) Recover $ID_i = B_3 \oplus h(PID_i \| N_i)$

3) Recover $N_j = B_6 \oplus h(PSID_j \| x)$
4) Recover $SID_j = B_7 \oplus h(PSID_j \| N_j)$
5) Recover $N_j \oplus N_c = B_9 \oplus H(N_i \| ID_i)$ or $N_i \oplus N_c = B_{13} \oplus h(N_j \| SID_j)$

Thus, $SK$ can be computed by $H(N_i \oplus N_j \oplus N_c) = h(N_j \oplus N_i \oplus N_c)$.

#### 2) PRIVILEGED-INSIDER ATTACKS

Assume that there is a malicious $U_i$ who tries to convince $CS$ that $S_j$ is willing to communicate with him. $U_i$ keeps two sets of $\{D_1, PID_i\}$, namely $D_1$, $PID_i$ and $D_1{}'$, $PID_i{}'$, when running two logins with other $S_j{}'$. $U_i$ now prepares his message $M_1$ faithfully using the old method. Then, this malicious $U_i$ will create the message $PSID_j, B_6, B_7, B_8$ as follows:

- This $U_i$ selects a random number $N_j$ and a timestamp $T_j$.
- The malicious $U_i$ sets $PSID_j = PID_i{}'$.
- The malicious $U_i$ sets $B_6 = D_1{}' \oplus h(ID_i \parallel HP_i) \oplus N_j = h(PID_i{}' \parallel x) \oplus N_j$.
- The malicious user sets $B_7 = h(PID_i{}' \parallel N_j) \oplus SID_j$.
- The malicious $U_i$ sets $B_8 = h(N_j \parallel SID_j \parallel B_5 \parallel T_j)$.

The malicious user sends the above-computed message along with $M_1$ to the $CS$. The latter will accept the authentication. The user and the $CS$ can complete mutual authentication (MA) and compute the session key. Some values will be updated by Wu *et al.*'s protocol after completion of the authentication.

After the malicious user and the $CS$ complete the authentication, the related information stored in the $CS$ and the $S$ may be inconsistent, and then the legitimate server cannot communicate normally. The details are as follows:

The $CS$ generates $N_c$, $PID_i^{new}$, $PSID_j^{new}$ and performs the same computations as the authentication phase above (computes $\{B_9 - B_{11}, SK_c, B_{12} - B_{14}, B_{16}\}$). Then $CS$ sends $M_3 = \{B_9, B_{10}, \ldots, B_{16}\}$ to $S_j$. The malicious user intercepts the message and then computes a new virtual identity. After such computations are completed, the virtual identity stored by the legitimate server is not the same as that stored in the control server. This causes data desynchronization, and in subsequent communications, the cloud server will be treated as an illegal individual.

#### 3) PRE-VERIFICATION IN SMART CARDS

In general, users will log in to the smart card before performing authentication. That is, when the user enters $ID$ and $PW$, the SC can verify them whether correct. However, Wu *et al.* did not provide such a process. In Wu *et al.*'s protocol, the user inputs $ID$ and $PW$, and because the smart card does not have a corresponding verification value, the smart card cannot perform any verification on the user's $PW$ and $ID$.

## III. ENHANCED PROTOCOL BASED ON WU *et al.*'s PROTOCOL

In this section, we present the details of the proposed protocol. Our protocol can resolve the above security problems.

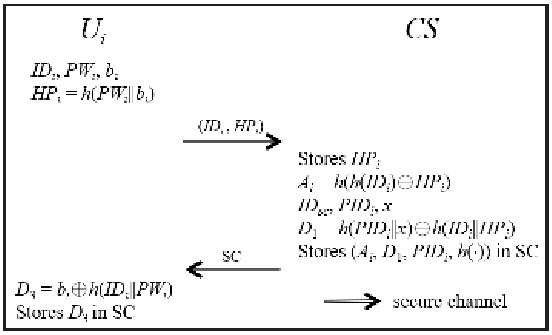

$U_i$
$CS$
$ID_i, PW_i, b_i$
$HP_i = h(PW_i \| b_i)$
$(ID_i, HP_i)$
Stores $HP_i$
$A_i \quad h(h(ID_i) \oplus HP_i)$
$ID_{gw}, PID_i, x$
$D_1 \quad h(PID_i \| x) \oplus h(ID_i \| HP_i)$
Stores $(A_i, D_1, PID_i, h(\cdot))$ in SC
SC
$D_3 = b_i \oplus h(ID_i \| PW_i)$
Stores $D_3$ in SC
secure channel

**FIGURE 2.** User registration phase.

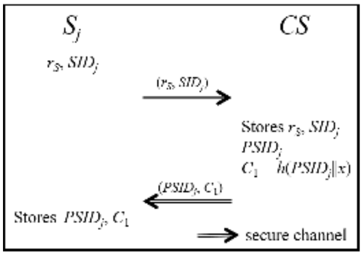


**FIGURE 3.** Cloud server registration phase.

There exist three roles: user $U_i$, cloud server $S_j$, and control server $CS$.

### A. USER AND CLOUD SERVER REGISTRATION PHASE

The user registers with $CS$ by executing the following steps. Fig. 2 demonstrates the user registration phase of the enhanced protocol.

1) $U_i$ determines $ID_i$, $PWi$, and $b_i$ to compute $HP_i = h(PW_i \| b_i)$. Then, it sends $ID_i$ and $HP_i$ via a secure channel.
2) $CS$ generates a pseudo identity $PID_i$ for $U_i$ and computes $A_i = h(h(ID_i) \oplus HP_i)$. $CS$ stores $HP_i$ into its database. Then, $CS$ computes $D_1$, stores $(PID_i, A_i, D_1, h(\cdot))$ into SC and sends it to $U_i$, where $D_1 = h(PID_i \| x) \oplus h(ID_i \| HP_i)$, $x$ is $CS$'s secret key.
3) After receiving the smart card, $U_i$ computers $D_3 = b_i \oplus h(ID_i \| PW_i)$ and stores $D_3$ into $SC$.

The cloud server registers with the control server by executing the following steps. Fig. 3 demonstrates the cloud server registration phase of the enhanced protocol.

1) $S_j$ selects a random number $r_S$ and its identity $SID_j$. It then sends $SID_j$ and $r_S$ to $CS$. Then, $CS$ stores $SID_j$ and $r_S$. $CS$ generates a pseudo identity $PSIDj$ for $Sj$.
2) $CS$ sends $(PSID_j, C_1)$ to $S_j$ via a secure channel, where $C_1 = h(PSID_j \| x)$.
3) Upon receiving this message, $S_j$ stores it into its database.

### B. AUTHENTICATION PHASE

When $U_i$ wants to access the service of some $S_j$, $CS$ can help to establish a session key. The detailed procedures are described as follows, and can also be found in Fig. 4.

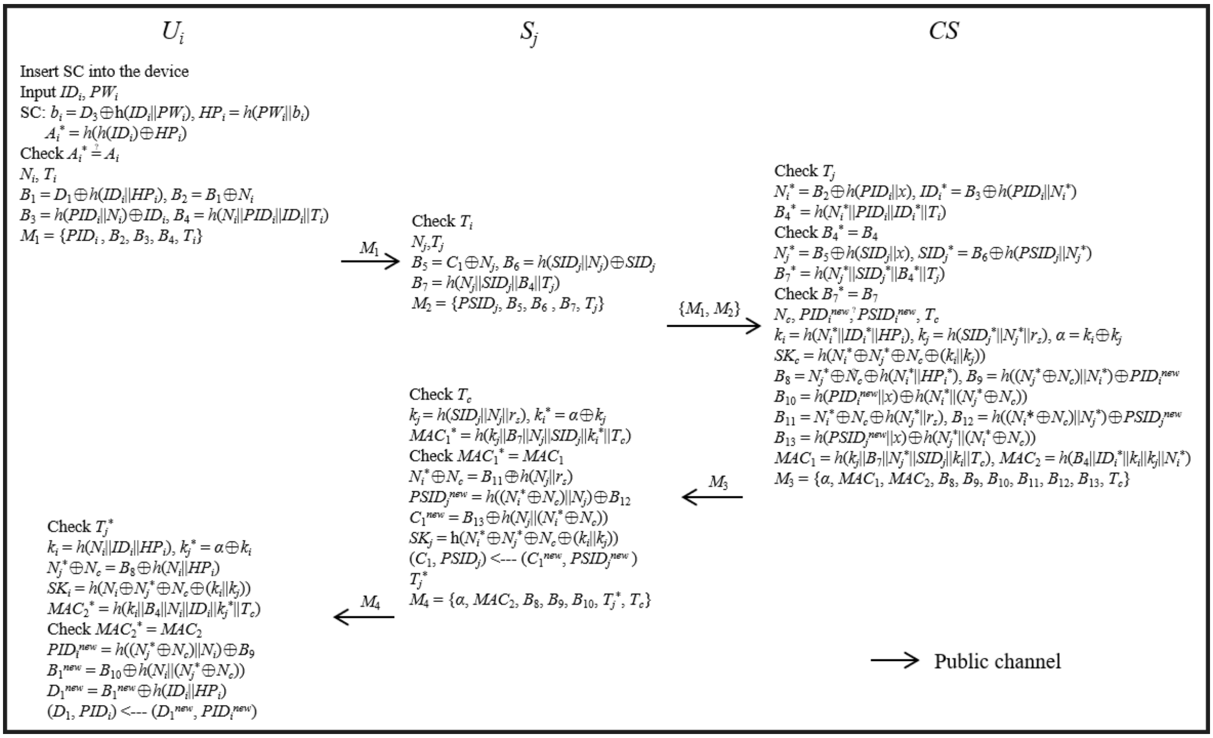

$U_i$
$S_j$
$CS$
Insert SC into the device
Input $ID_i$, $PW_i$
SC: $b_i = D_3 \oplus h(ID_i \| PW_i)$, $HP_i = h(PW_i \| b_i)$
$A_i^* = h(h(ID_i) \oplus HP_i)$
Check $A_i^* \overset{?}{=} A_i$
$N_i$, $T_i$
$B_1 = D_1 \oplus h(ID_i \| HP_i)$, $B_2 = B_1 \oplus N_i$
$B_3 = h(PID_i \| N_i) \oplus ID_i$, $B_4 = h(N_i \| PID_i \| ID_i \| T_i)$
$M_1 = \{PID_i, B_2, B_3, B_4, T_i\}$
$M_1$
Check $T_i$
$N_j, T_j$
$B_5 = C_1 \oplus N_j$, $B_6 = h(SID_j \| N_j) \oplus SID_j$
$B_7 = h(N_j \| SID_j \| B_4 \| T_j)$
$M_2 = \{PSID_j, B_5, B_6, B_7, T_j\}$
$\{M_1, M_2\}$
Check $T_j$
$N_i^* = B_2 \oplus h(PID_i \| x)$, $ID_i^* = B_3 \oplus h(PID_i \| N_i^*)$
$B_4^* = h(N_i^* \| PID_i \| ID_i^* \| T_i)$
Check $B_4^* = B_4$
$N_j^* = B_5 \oplus h(SID_j \| x)$, $SID_j^* = B_6 \oplus h(PSID_j \| N_j^*)$
$B_7^* = h(N_j^* \| SID_j^* \| B_4^* \| T_j)$
Check $B_7^* = B_7$
$N_c$, $PID_i^{new}$, $PSID_i^{new}$, $T_c$
$k_i = h(N_i^* \| ID_i^* \| HP_i)$, $k_j = h(SID_j^* \| N_j^* \| r_s)$, $\alpha = k_i \oplus k_j$
$SK_c = h(N_i^* \oplus N_j^* \oplus N_c \oplus (k_i \| k_j))$
$B_8 = N_j^* \oplus N_c \oplus h(N_i^* \| HP_i^*)$, $B_9 = h((N_j^* \oplus N_c) \| N_i^*) \oplus PID_i^{new}$
$B_{10} = h(PID_i^{new} \| x) \oplus h(N_i^* \| (N_j^* \oplus N_c))$
$B_{11} = N_i^* \oplus N_c \oplus h(N_j^* \| r_s)$, $B_{12} = h((N_i^* \oplus N_c) \| N_j^*) \oplus PSID_j^{new}$
$B_{13} = h(PSID_j^{new} \| x) \oplus h(N_j^* \| (N_i^* \oplus N_c))$
$MAC_1 = h(k_j \| B_7 \| N_j^* \| SID_j \| k_i \| T_c)$, $MAC_2 = h(B_4 \| ID_i^* \| k_i \| k_j \| N_i^*)$
$M_3 = \{\alpha, MAC_1, MAC_2, B_8, B_9, B_{10}, B_{11}, B_{12}, B_{13}, T_c\}$
$M_3$
Check $T_c$
$k_j = h(SID_j \| N_j \| r_s)$, $k_i^* = \alpha \oplus k_j$
$MAC_1^* = h(k_j \| B_7 \| N_j \| SID_j \| k_i^* \| T_c)$
Check $MAC_1^* = MAC_1$
$N_i^* \oplus N_c = B_{11} \oplus h(N_j \| r_s)$
$PSID_j^{new} = h((N_i^* \oplus N_c) \| N_j) \oplus B_{12}$
$C_1^{new} = B_{13} \oplus h(N_j \| (N_i^* \oplus N_c))$
$SK_j = h(N_i^* \oplus N_j^* \oplus N_c \oplus (k_i \| k_j))$
$(C_1, PSID_j)$ <--- $(C_1^{new}, PSID_j^{new})$
$T_j^*$
$M_4 = \{\alpha, MAC_2, B_8, B_9, B_{10}, T_j^*, T_c\}$
$M_4$
Check $T_j^*$
$k_i = h(N_i \| ID_i \| HP_i)$, $k_j^* = \alpha \oplus k_i$
$N_j^* \oplus N_c = B_8 \oplus h(N_i \| HP_i)$
$SK_i = h(N_i \oplus N_j^* \oplus N_c \oplus (k_i \| k_j))$
$MAC_2^* = h(k_i \| B_4 \| N_i \| ID_i \| k_j^* \| T_c)$
Check $MAC_2^* = MAC_2$
$PID_i^{new} = h((N_j^* \oplus N_c) \| N_i) \oplus B_9$
$B_1^{new} = B_{10} \oplus h(N_i \| (N_j^* \oplus N_c))$
$D_1^{new} = B_1^{new} \oplus h(ID_i \| HP_i)$
$(D_1, PID_i)$ <--- $(D_1^{new}, PID_i^{new})$
→ Public channel

**FIGURE 4.** The authentication phase.

1) At the beginning, $U_i$ inserts a SC into the device and enters an $ID_i$ and $PW_i$.
The $SC$ computes $b_i = D_3 \oplus h(ID_i \| PW_i)$, $HP_i = h(PW_i \| b_i)$ and $A_i^* = h(h(ID_i) \oplus HP_i)$, and confirms the user credentials by matching $A_i^*$ with $A_i$.
If they match, $SC$ selects $N_i$ and computes $B_1, B_2, B_3, B_4$:

$$\begin{aligned} B_1 &= D_1 \oplus h(ID_i \| HP_i) \\ B_2 &= B_1 \oplus N_1 \\ B_3 &= h(PID_i \| N_i) \oplus ID_i \\ B_4 &= h(N_i \| PID_i \| ID_i \| T_i) \end{aligned}$$

Finally, $U_i$ sends $M_1 = \{PID_i, B_2, B_3, B_4, T_i\}$ to $S_j$.

2) Upon receipt of message $M_1$, $S_j$ validates the timestamp $T_i$ first, and only if the timestamp is valid can the next calculation be performed.
Then, $S_j$ selects a random $N_j$, and computes $B_5, B_6, B_7$:

$$\begin{aligned} B_5 &= C_1 \oplus N_j \\ B_6 &= h(PSID_j \| N_j) \oplus SID_j \\ B_7 &= h(N_j \| B_4 \| T_j) \end{aligned}$$

Finally, $S_j$ sends $M_1$ and $M_2 = \left\{PSID_j, B_5, B_6, B_7, T_j\right\}$ to $CS$.

3) After receiving $M_1$ and $M_2$ from $S_j$, $CS$ checks the validity of the $T_j$.
$CS$ recovers $N_i^* = B_2 \oplus h(PID_i \| x)$ and $ID_i^* = B_3 \oplus h(PID_i \| N_i^*)$. Then, $CS$ verifies $ID_i^*$ and $B_4 = h(N_i^* \| PID_i \| ID_i^* \| T_i)$. If not equal, terminate.
$CS$ computes $N_j^* = B_5 \oplus h(SID_j \| x)$ and $SID_j^* = B_6 \oplus h(PSID_j \| N_j^*)$.Then, $CS$ verifies $SID_j^*$ and $B_7 = h(N_j^* \| SID_j^* \| B_4 \| T_j)$. If not equal, terminate.
After the authentication is completed, $CS$ takes a random number $N_c$, timestamp $T_c$, and the new virtual identity $(PID_i^{new}, PSID_j^{new})$, then computes:

$$\begin{aligned} k_i &= h(N_i^* \| ID_i^* \| HP_i) \\ k_j &= h(SID_j^* \| N_j^* \| r_S) \\ \alpha &= k_i \oplus k_j \\ SK_c &= h(N_i^* \oplus N_j^* \oplus N_c \oplus (k_i \| k_j)) \\ B_8 &= N_j^* \oplus N_c \oplus h(N_i^* \| HP_i) \\ B_9 &= h((N_j^* \oplus N_c) \ \| \ N_i^*) \oplus PID_i^{new} \\ B_{10} &= h(PID_i^{new} \ \| \ x) \oplus h(N_i^* \ \| \ (N_j^* \oplus N_c)) \\ B_{11} &= N_i^* \oplus N_c \oplus h(N_j^* \| r_S) \\ B_{12} &= h((N_i^* \oplus N_c) \ \| \ N_j^*) \oplus PSID_j^{new} \\ B_{13} &= h(PSID_j^{new} \ \| \ x) \oplus h(N_j^* \ \| \ (N_i^* \oplus N_c)) \\ MAC_1 &= h(k_j \| B_7 \| N_j^* \| SID_j^* \| k_i \| T_c) \\ MAC_2 &= h(k_i \| K_j \| B_4 \| N_i^* \| ID_i^* \| T_c) \end{aligned}$$

Then, $CS$ sends $M_3 = \{\alpha, MAC_1, MAC_2, B_8 - B_{13}, T_c\}$ to $S_j$.

4) Upon receipt of message $M_3$, $S_j$ validates the timestamp $T_c$ first, and only if the timestamp is valid can the next calculation be performed. Then, $S_j$ computes

$$\begin{aligned} k_j^* &= h(SID_j \| N_j \| r_S) \\ k_i^* &= \alpha \oplus k_j^* \end{aligned}$$

it verifies $MAC_1 = h(k_j^*\|B_7\|N_j\|SID_j\|k_i^*\|T_c)$. If the verification does not hold, $S_j$ terminates. Otherwise, $S_j$ authenticates $CS$. After the authentication, $S_j$ recovers

$$\begin{aligned} N_i^*\oplus N_c &= B_9\oplus h(N_j\|r_S)\\ PSID_j^{new} &= B_{12}\oplus h((N_i^*\oplus N_c)\|N_j)\\ C_1^{new} &= B_{13}\oplus h(N_j\|(N_i^*\oplus N_c))\\ SK_j &= h(N_i^*\oplus N_c\oplus N_j\oplus(k_i^*\|k_j^*)) \end{aligned}$$

Then, $(C_1, PSID_j)$ is replaced by $(C_1^{new}, PSID_j^{new})$. To verify the session key again, $S_j$ generates a nonce $b_j$, and then computes $V_1 = b_j\oplus(k_i^*\|k_j^*)$. Finally, $S_j$ sends $M_4 = \left\{\alpha, V_1, MAC_2, B_8, B_9, B_{10}, T_c, T_j^*\right\}$ to $U_i$.

5) Upon receiving $M_4$, $U_i$ first checks the timestamp $T_j^*$. Then, $U_i$ computes:

$$\begin{aligned} k_i^* &= h(N_i\|ID_i\|HP_i)\\ k_j^* &= \alpha\oplus k_i^*\\ MAC_2^* &= h(B_4\|ID_i\|k_i^*\|k_j^*\|N_i\|T_c) \end{aligned}$$

If $MAC_2^* = MAC_2$, $CS$ is authenticated by $U_i$. Otherwise, $U_i$ terminates. Then, $U_i$ computes:

$$\begin{aligned} N_j^*\oplus N_c &= B_8\oplus h(N_i\|HP_i)\\ SK_i &= h(N_j^*\oplus N_c\oplus N_i(k_i^*\|k_j^*))\\ PID_i^{new} &= B_9\oplus h(N_i\|(N_j^*\oplus N_c))\\ B_1^{new} &= B_{10}\oplus h(N_i\|(N_j^*\oplus N_c))\\ D_1^{new} &= B_1^{new}\oplus h(ID_i\|HP_i)\\ b_j^* &= V_1\oplus(k_i^*\|k_j^*)\\ MAC_3 &= h(SK_i\|b_j^*) \end{aligned}$$

Then, $U_i$ updates $(D_1, PID_i)$ to $(D_1^{new}, PID_i^{new})$ and sends $\{MAC_3\}$ to $S_j$.

Finally, $S_j$ checks $MAC_3 =?h(SK_j||b_j)$. If this is true, the session key is $SK_i = SK_j = SK_c$.

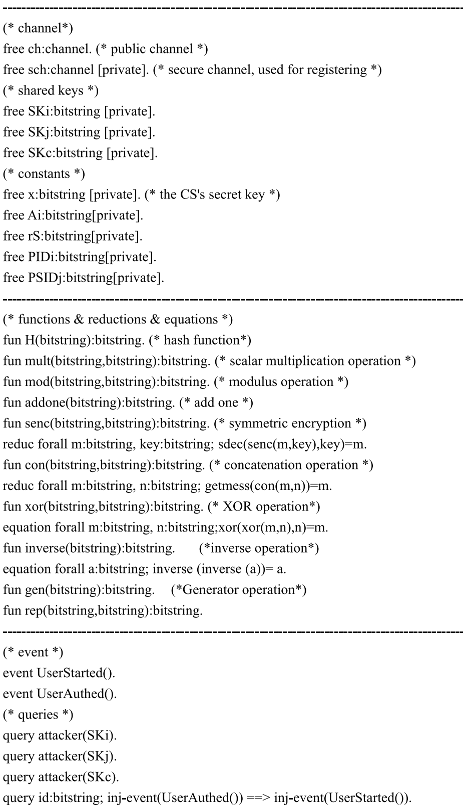
## IV. SECURITY ANALYSIS OF THE ENHANCED PROTOCOL

In this section, we use BAN logic [46]–[50], ProVerif [51], and informal security analysis to show the security of our enhanced protocol.

### A. SECURITY ANALYSIS THROUGH PROVERIF

Through user, cloud server, and control server registration and authentication process programming, we create an authentication protocol simulation. The whole process in ProVerif is:

1) A public channel *ch* is defined for login and authentication. A secure channel *sch* is used for registration of the users and cloud servers. $SK_i$, $SK_j$, and $SK_c$ are the session keys generated by $U_i$, $S_j$, and $SC$. Then, string connection operation, XOR operation, and hash function are defined. We made some queries to validate the security requirements. A process of function definition is shown in Fig. 5.
2) A process of $U_i$ is shown in Fig. 6.



```
------------------------------------------------------------------------------------------------
(* channel*)
free ch:channel. (* public channel *)
free sch:channel [private]. (* secure channel, used for registering *)
(* shared keys *)
free SKi:bitstring [private].
free SKj:bitstring [private].
free SKc:bitstring [private].
(* constants *)
free x:bitstring [private]. (* the CS's secret key *)
free Ai:bitstring[private].
free rS:bitstring[private].
free PIDi:bitstring[private].
free PSIDj:bitstring[private].
------------------------------------------------------------------------------------------------
(* functions & reductions & equations *)
fun H(bitstring):bitstring. (* hash function*)
fun mult(bitstring,bitstring):bitstring. (* scalar multiplication operation *)
fun mod(bitstring,bitstring):bitstring. (* modulus operation *)
fun addone(bitstring):bitstring. (* add one *)
fun senc(bitstring,bitstring):bitstring. (* symmetric encryption *)
reduc forall m:bitstring, key:bitstring; sdec(senc(m,key),key)=m.
fun con(bitstring,bitstring):bitstring. (* concatenation operation *)
reduc forall m:bitstring, n:bitstring; getmess(con(m,n))=m.
fun xor(bitstring,bitstring):bitstring. (* XOR operation*)
equation forall m:bitstring, n:bitstring;xor(xor(m,n),n)=m.
fun inverse(bitstring):bitstring.      (*inverse operation*)
equation forall a:bitstring; inverse (inverse (a))= a.
fun gen(bitstring):bitstring.    (*Generator operation*)
fun rep(bitstring,bitstring):bitstring.
------------------------------------------------------------------------------------------------
(* event *)
event UserStarted().
event UserAuthed().
(* queries *)
query attacker(SKi).
query attacker(SKj).
query attacker(SKc).
query id:bitstring; inj-event(UserAuthed()) ==> inj-event(UserStarted()).
```

**FIGURE 5.** **Predefinition code.**

3) A Process of $S_j$ is shown in Fig. 7.
4) A Process of $SC$ is shown in Fig. 8.
5) In Fig. 9, we state the protocol using UserAuthed() and UserStarted(), and the verification results are "RESULT not attacker(SKi[]) is true", "RESULT not attacker(SKj[]) is true", "RESULT not attacker(SKc[]) is true", and "RESULT inj-event(UserAuthed) ==> inj-event(UserStarted) is true".

Thus, we conclude that $SK_i$, $SK_j$, and $SK_c$ withstood the attacks and the enhanced protocol passed the verification by ProVerif.

### B. FORMAL SECURITY ANALYSIS USING BAN LOGIC

In this subsection, we will show that $U_i$ and $S_j$ share a key $SK$, which is calculated by the $CS$ so that when the user wants to get the server's data, this key can be used to send a request message to the server. Note that the following notations and rules for BAN logic are referred to [46]–[50].

#### 1) GOALS

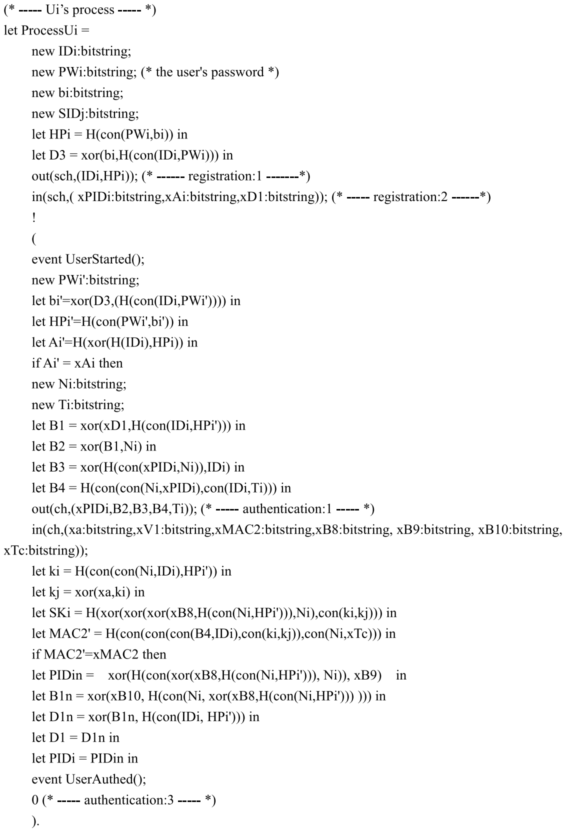
Our goals are defined as follows.

G1 $U_i \mid\equiv U_i \xleftrightarrow{SK} S_j$.

G2 $S_j \mid\equiv U_i \xleftrightarrow{SK} S_j$.

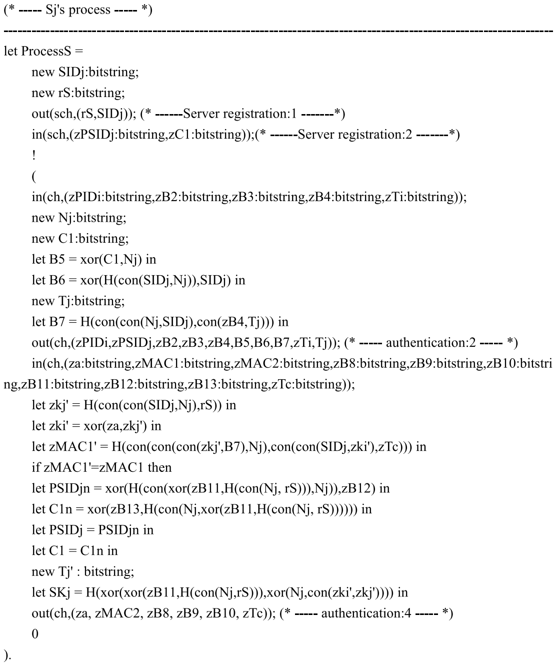
G3 $CS \mid\equiv U_i \xleftrightarrow{SK} S_j$.

G4 $U_i \mid\equiv S_j \mid\equiv U_i \xleftrightarrow{SK} S_j$.



```
(* ----- Ui's process ----- *)
let ProcessUi =
        new IDi:bitstring;
        new PWi:bitstring; (* the user's password *)
        new bi:bitstring;
        new SIDj:bitstring;
        let HPi = H(con(PWi,bi)) in
        let D3 = xor(bi,H(con(IDi,PWi))) in
        out(sch,(IDi,HPi)); (* ------ registration:1 -------*)
        in(sch,( xPIDi:bitstring,xAi:bitstring,xD1:bitstring)); (* ----- registration:2 ------*)
        !
        (
        event UserStarted();
        new PWi':bitstring;
        let bi'=xor(D3,(H(con(IDi,PWi')))) in
        let HPi'=H(con(PWi',bi')) in
        let Ai'=H(xor(H(IDi),HPi)) in
        if Ai' = xAi then
        new Ni:bitstring;
        new Ti:bitstring;
        let B1 = xor(xD1,H(con(IDi,HPi'))) in
        let B2 = xor(B1,Ni) in
        let B3 = xor(H(con(xPIDi,Ni)),IDi) in
        let B4 = H(con(con(Ni,xPIDi),con(IDi,Ti))) in
        out(ch,(xPIDi,B2,B3,B4,Ti)); (* ----- authentication:1 ----- *)
        in(ch,(xa:bitstring,xV1:bitstring,xMAC2:bitstring,xB8:bitstring, xB9:bitstring, xB10:bitstring,
xTc:bitstring));
        let ki = H(con(con(Ni,IDi),HPi')) in
        let kj = xor(xa,ki) in
        let SKi = H(xor(xor(xor(xB8,H(con(Ni,HPi'))),Ni),con(ki,kj))) in
        let MAC2' = H(con(con(con(B4,IDi),con(ki,kj)),con(Ni,xTc))) in
        if MAC2'=xMAC2 then
        let PIDin =    xor(H(con(xor(xB8,H(con(Ni,HPi'))), Ni)), xB9)    in
        let B1n = xor(xB10, H(con(Ni, xor(xB8,H(con(Ni,HPi'))) ))) in
        let D1n = xor(B1n, H(con(IDi, HPi'))) in
        let D1 = D1n in
        let PIDi = PIDin in
        event UserAuthed();
        0 (* ----- authentication:3 ----- *)
        ).
```

FIGURE 6. The process of $U_i$.



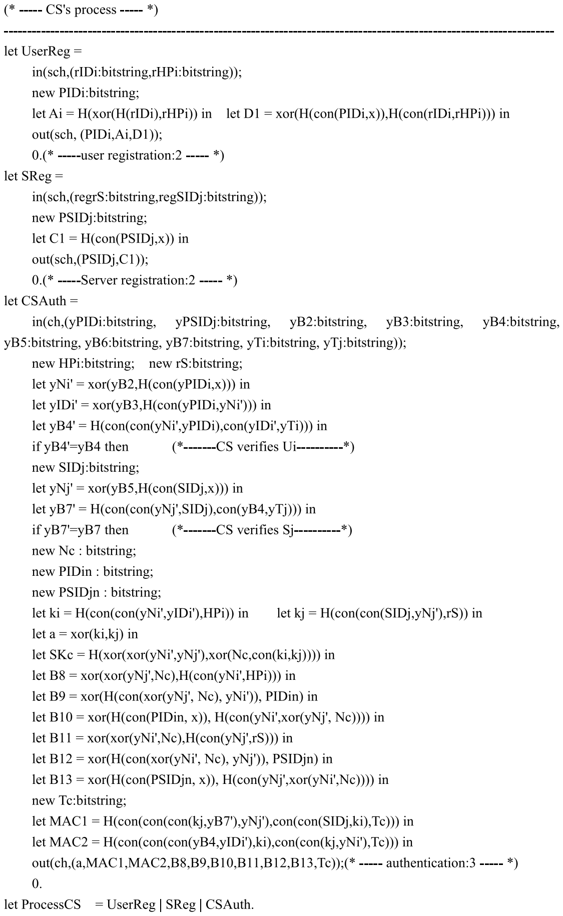
```
(* ----- Sj's process ----- *)
------------------------------------------------------------------------------------------------------------------
let ProcessS =
        new SIDj:bitstring;
        new rS:bitstring;
        out(sch,(rS,SIDj)); (* ------Server registration:1 -------*)
        in(sch,(zPSIDj:bitstring,zC1:bitstring));(* ------Server registration:2 -------*)
        !
        (
        in(ch,(zPIDi:bitstring,zB2:bitstring,zB3:bitstring,zB4:bitstring,zTi:bitstring));
        new Nj:bitstring;
        new C1:bitstring;
        let B5 = xor(C1,Nj) in
        let B6 = xor(H(con(SIDj,Nj)),SIDj) in
        new Tj:bitstring;
        let B7 = H(con(con(Nj,SIDj),con(zB4,Tj))) in
        out(ch,(zPIDi,zPSIDj,zB2,zB3,zB4,B5,B6,B7,zTi,Tj)); (* ----- authentication:2 ----- *)
        in(ch,(za:bitstring,zMAC1:bitstring,zMAC2:bitstring,zB8:bitstring,zB9:bitstring,zB10:bitstri
ng,zB11:bitstring,zB12:bitstring,zB13:bitstring,zTc:bitstring));
        let zkj' = H(con(con(SIDj,Nj),rS)) in
        let zki' = xor(za,zkj') in
        let zMAC1' = H(con(con(con(zkj',B7),Nj),con(con(SIDj,zki'),zTc))) in
        if zMAC1'=zMAC1 then
        let PSIDjn = xor(H(con(xor(zB11,H(con(Nj, rS))),Nj)),zB12) in
        let C1n = xor(zB13,H(con(Nj,xor(zB11,H(con(Nj, rS)))))) in
        let PSIDj = PSIDjn in
        let C1 = C1n in
        new Tj' : bitstring;
        let SKj = H(xor(xor(zB11,H(con(Nj,rS))),xor(Nj,con(zki',zkj')))) in
        out(ch,(za, zMAC2, zB8, zB9, zB10, zTc)); (* ----- authentication:4 ----- *)
        0
).
```

FIGURE 7. The process of $S_i$.

G5 $S_j \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.

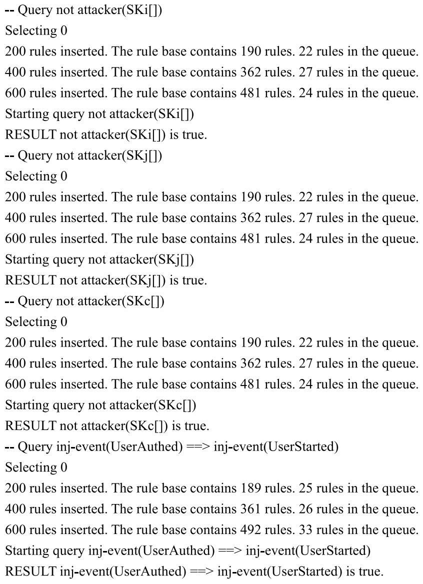
G6 $CS \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.

G7 $CS \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.



```
(* ----- CS's process ----- *)
------------------------------------------------------------------------------------------------------------------
let UserReg =
        in(sch,(rIDi:bitstring,rHPi:bitstring));
        new PIDi:bitstring;
        let Ai = H(xor(H(rIDi),rHPi)) in    let D1 = xor(H(con(PIDi,x)),H(con(rIDi,rHPi))) in
        out(sch, (PIDi,Ai,D1));
        0.(* -----user registration:2 ----- *)
let SReg =
        in(sch,(regrS:bitstring,regSIDj:bitstring));
        new PSIDj:bitstring;
        let C1 = H(con(PSIDj,x)) in
        out(sch,(PSIDj,C1));
        0.(* -----Server registration:2 ----- *)
let CSAuth =
        in(ch,(yPIDi:bitstring,    yPSIDj:bitstring,    yB2:bitstring,    yB3:bitstring,    yB4:bitstring,
yB5:bitstring, yB6:bitstring, yB7:bitstring, yTi:bitstring, yTj:bitstring));
        new HPi:bitstring;    new rS:bitstring;
        let yNi' = xor(yB2,H(con(yPIDi,x))) in
        let yIDi' = xor(yB3,H(con(yPIDi,yNi'))) in
        let yB4' = H(con(con(yNi',yPIDi),con(yIDi',yTi))) in
        if yB4'=yB4 then            (*-------CS verifies Ui----------*)
        new SIDj:bitstring;
        let yNj' = xor(yB5,H(con(SIDj,x))) in
        let yB7' = H(con(con(yNj',SIDj),con(yB4,yTj))) in
        if yB7'=yB7 then            (*-------CS verifies Sj----------*)
        new Nc : bitstring;
        new PIDin : bitstring;
        new PSIDjn : bitstring;
        let ki = H(con(con(yNi',yIDi'),HPi)) in        let kj = H(con(con(SIDj,yNj'),rS)) in
        let a = xor(ki,kj) in
        let SKc = H(xor(xor(yNi',yNj'),xor(Nc,con(ki,kj)))) in
        let B8 = xor(xor(yNj',Nc),H(con(yNi',HPi))) in
        let B9 = xor(H(con(xor(yNj', Nc), yNi')), PIDin) in
        let B10 = xor(H(con(PIDin, x)), H(con(yNi',xor(yNj', Nc)))) in
        let B11 = xor(xor(yNi',Nc),H(con(yNj',rS))) in
        let B12 = xor(H(con(xor(yNi', Nc), yNj')), PSIDjn) in
        let B13 = xor(H(con(PSIDjn, x)), H(con(yNj',xor(yNi',Nc)))) in
        new Tc:bitstring;
        let MAC1 = H(con(con(con(kj,yB7'),yNj'),con(con(SIDj,ki),Tc))) in
        let MAC2 = H(con(con(con(yB4,yIDi'),ki),con(con(kj,yNi'),Tc))) in
        out(ch,(a,MAC1,MAC2,B8,B9,B10,B11,B12,B13,Tc));(* ----- authentication:3 ----- *)
        0.
let ProcessCS    = UserReg | SReg | CSAuth.
```

FIGURE 8. The process of $CS$.



```
-- Query not attacker(SKi[])
Selecting 0
200 rules inserted. The rule base contains 190 rules. 22 rules in the queue.
400 rules inserted. The rule base contains 362 rules. 27 rules in the queue.
600 rules inserted. The rule base contains 481 rules. 24 rules in the queue.
Starting query not attacker(SKi[])
RESULT not attacker(SKi[]) is true.
-- Query not attacker(SKj[])
Selecting 0
200 rules inserted. The rule base contains 190 rules. 22 rules in the queue.
400 rules inserted. The rule base contains 362 rules. 27 rules in the queue.
600 rules inserted. The rule base contains 481 rules. 24 rules in the queue.
Starting query not attacker(SKj[])
RESULT not attacker(SKj[]) is true.
-- Query not attacker(SKc[])
Selecting 0
200 rules inserted. The rule base contains 190 rules. 22 rules in the queue.
400 rules inserted. The rule base contains 362 rules. 27 rules in the queue.
600 rules inserted. The rule base contains 481 rules. 24 rules in the queue.
Starting query not attacker(SKc[])
RESULT not attacker(SKc[]) is true.
-- Query inj-event(UserAuthed) ==> inj-event(UserStarted)
Selecting 0
200 rules inserted. The rule base contains 189 rules. 25 rules in the queue.
400 rules inserted. The rule base contains 361 rules. 26 rules in the queue.
600 rules inserted. The rule base contains 492 rules. 33 rules in the queue.
Starting query inj-event(UserAuthed) ==> inj-event(UserStarted)
RESULT inj-event(UserAuthed) ==> inj-event(UserStarted) is true.
```

FIGURE 9. Verification result.

### 2) IDEALIZE THE COMMUNICATION MESSAGES

M1 $U_i \rightarrow S_j$: $\{PID_i, B_2, B_3, B_4, T_i\}$.

M2 $U_i \rightarrow CS$: $\{PID_i, B_2, B_3, B_4\}$.

M3 $S_j \rightarrow CS$: $\{PSID_j, B_5, B_6, B_7, T_j, PID_i, B_2, B_3, B_4\}$.

M4 $CS \rightarrow U_i$: $\{\alpha, MAC_2, B_8, T_c\}$.

M5 $CS \rightarrow S_j$: $\{\alpha, MAC_1, B_9, T_c, MAC_2, B_8\}$.

M6 $S_j \rightarrow U_i$: $\{\alpha, MAC_2, B_8, T_c, T_j, V_1\}$.

### 3) INITIAL STATE ASSUMPTIONS

A1 $U_i \mid\equiv \sharp(N_i)$.

A2 $S_j \mid\equiv \sharp(N_j)$.

A3 $CS \mid\equiv \sharp(N_c)$.

A4 $CS \mid\equiv U_i \overset{x}{\rightleftharpoons} CS$.

A5 $CS \mid\equiv \sharp(PID_i)$.

A6 $CS \mid\equiv \sharp(PSID_i)$.

A7 $CS \mid\equiv U_i \mid\Longrightarrow N_i$.

A8 $CS \mid\equiv S_j \mid\Longrightarrow N_j$.

A9 $CS \mid\equiv U_i \mid\Longrightarrow ID_i$.

A10 $CS \mid\equiv S_j \mid\Longrightarrow ID_j$.

A11 $CS \mid\equiv \sharp(ID_i)$.

A12 $CS \mid\equiv \sharp(SID_j)$.

A13 $U_i \mid\equiv U_i \overset{HP_i}{\rightleftharpoons} CS$.

A14 $CS \mid\equiv U_i \overset{HP_i}{\rightleftharpoons} CS$.

A15 $CS \mid\equiv S_j \overset{x}{\rightleftharpoons} CS$.

A16 $CS \mid\equiv S_j \overset{r_S}{\rightleftharpoons} CS$.

A17 $S_j \mid\equiv S_j \overset{r_S}{\rightleftharpoons} CS$.

A18 $U_i \mid\equiv U_i \overset{k_i}{\rightleftharpoons} CS$.

A19 $U_i \mid\equiv CS \mid\Longrightarrow k_j$.

A20 $S_j \mid\equiv CS \mid\Longrightarrow k_i$.

A21 $U_i \mid\equiv \sharp(N_j \oplus N_c)$.

A22 $U_i \mid\equiv CS \mid\Longrightarrow (N_j \oplus N_c)$.

A23 $S_j \mid\equiv S \overset{k_j}{\rightleftharpoons} CS$.

A24 $S_j \mid\equiv \sharp(N_i \oplus N_c)$.

A25 $S_j \mid\equiv CS \mid\Longrightarrow (N_i \oplus N_c)$.

A26 $S_j \mid\equiv S_j \overset{x}{\rightleftharpoons} CS$.

A27 $U_i \mid\equiv U_i \overset{x}{\rightleftharpoons} CS$.

A28 $S_j \mid\equiv \sharp(PID_j)$.

A29 $S_j \mid\equiv U_i \mid\Longrightarrow N_i$.

A30 $CS \mid\equiv \sharp(N_i)$.

A31 $CS \mid\equiv \sharp(N_j)$.

### 4) MAIN PROOFS USING BAN RULES AND ASSUMPTIONS

According to M1 and using the seeing rule, we get

**S1**: $S_j \triangleleft \{PID_i, B_2 : \langle N_i, PID_i\rangle_x; B_3, B_4, T_i\}$.

Using S1, we get

**S2**: $S_j \triangleleft \{\langle N_i, PID_i\rangle_x\}$.

Using A26, A27, we get

**S3**: $S_j \mid\equiv S_j \overset{x}{\rightleftharpoons} U_i$.

Using S2, S3, and the message-meaning (M-M) rule, we get

**S4**:$S_j \mid\equiv U_i \mid\sim (N_i, PID_i)$.

Using A28, S4, the freshness rule, and the nonce-verification (N-V) rule, we get

**S5**:$S_j \mid\equiv U_i \mid\equiv (N_i, PID_i)$.

Applying this for each component, we get

**S6**:$S_j \mid\equiv U_i \mid\equiv N_i$.

Using A29, S6, and the jurisdiction rule, we get

**S7**: $S_j \mid\equiv N_i$.

According to the message M2 and using the seeing rule, we get

**S8**: $CS \triangleleft \{PID_i, B_2 : \langle N_i, PID_i\rangle_x; B_3 : \langle ID_i\rangle_{h(PID_i \| N_i)}; B_4, T_j\}$.

Using the seeing rule for components we get

**S9**: $CS \triangleleft \{\langle N_i, PID_i\rangle_x\}$.

Using A4, S9, and the M-M rule, we get

**S10**: $CS \mid\equiv U_i \mid\sim (N_i, PID_i)$.

Using A5, S3, the freshness rule, and the N-V rule, we get

**S11**: $CS \mid\equiv U_i \mid\equiv (N_i, PID_i)$.

Using S11 and the belief rule, we get

**S12**: $CS \mid\equiv U_i \mid\equiv (N_i)$.

**S13**: $CS \mid\equiv U_i \mid\equiv (PID_i)$.

Using A7, S12, and the jurisdiction rule, we get

**S14**: $CS \mid\equiv N_i$.

According to S8 and using the seeing rule, we get

**S15**: $CS \triangleleft \{\langle ID_i\rangle_{h(PID_i \| N_i)}\}$.

Using A5, S14, and the M-M rule, we get

**S16**: $CS \mid\equiv U_i \mid\sim ID_i$.

Using A11, S16, and the N-V rule, we get

**S17**: $CS \mid\equiv U_i \mid\equiv ID_i$.

Using A9, S17, and the jurisdiction rule, we get

**S18**: $CS \mid\equiv ID_i$.

Using A14, S14, S18, and the belief rule, we get

**S19**: $CS \mid\equiv (ID_i, N_i, HP_i)$.

Because $K_i = h(N_i \parallel ID_i \parallel HP_i)$, we can get

**S20**: $CS \mid\equiv k_i$.

According to message M3 and using the seeing rule, we get

**S21**: $CS \triangleleft \{PSID_j, B_5 : \left\langle N_j, PSID_i\right\rangle_x; B_6 : \left\langle SID_j\right\rangle_{h(PSID_j \| N_j)}; B_7, T_j\}$.

Using the seeing rule for components we get

**S22**: $CS \triangleleft \{\left\langle N_j, PSID_i\right\rangle_x\}$.

Using A15, S22, and the message-meaning rule, we get

**S23**: $CS \mid\equiv S_j \mid\sim (N_j, PSID_j)$.

Using A6, S23, the freshness rule, and the N-V rule, we get

**S24**: $CS \mid\equiv S_j \mid\equiv (N_j, PSID_i)$.

Using the belief rule for components we get

**S25**: $CS \mid\equiv S_j \mid\equiv (N_j)$.

**S26**: $CS \mid\equiv S_j \mid\equiv (PSID_j)$.

Using A8, S25, and the jurisdiction rule, we get

**S26**: $CS \mid\equiv N_j$.

According to the S21 and using the seeing rule, we get

**S27**: $CS \triangleleft \{\left\langle SID_j\right\rangle_{h(PSID_j \| N_j)}\}$.

Using S26, $CS \triangleleft PSID_j$, and the M-M rule, we get

**S28**: $CS \mid\equiv S_j \mid\sim SID_j$.

Using A12, S28, and the N-V rule, we get

**S29**: $CS \mid\equiv S_j \mid\equiv SID_j$.

Using A10, S29, and the jurisdiction rule, we get

**S30**: $CS \mid\equiv SID_j$.

Using A16, S30, S26, and the belief rule, we get

**S31**: $CS \mid\equiv (SID_j, N_j, r_S)$.

Because $K_j = h(N_j \parallel SID_j \parallel r_S)$, we can get

**S32**: $CS \mid\equiv k_j$.

Using A3, S14, S20, S26, S32, and the belief rule, we get

**S33**: $CS \mid\equiv U_i \overset{SK}{\longleftrightarrow} S$.(**G3**) and

Using A30, S33, and the session key (SK) rule, we get

**S34**: $CS \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G6**)

Using A31, S33, and the SK rule, we obtain

**S35**: $CS \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G7**)

According to message M4 and using the seeing rule, we get

**S36**: $U_i \triangleleft \{\alpha : \langle k_j \rangle_{k_i}; MAC_2 : \langle B_4, ID_i, k_j, N_i, T_c \rangle_{k_i}; B_8 : \langle N_j \oplus N_c \rangle_{h(N_i \| HP_i)}; T_c\}$.

Using the seeing rule for components we get

**S37**: $U_i \triangleleft \{\langle B_4, ID_i, k_j, N_i, T_c \rangle_{k_i}\}$.

Using A18, S37, and the M-M rule, we get

**S38**: $U_i \mid\equiv CS \mid\sim (B_4, ID_i, k_j, N_i, T_c)$.

Using A1, S38, the freshness rule, and the N-V rule, we get

**S39**: $U_i \mid\equiv CS \mid\equiv (B_4, ID_i, k_j, N_i, T_c)$.

Using the belief rule for components we get

**S40**: $U_i \mid\equiv CS \mid\equiv k_j$.

Using A19, S40, and the N-V rule, we get

**S41**: $U_i \mid\equiv k_j$.

According to S36 and using the seeing rule, we get

**S42**: $U_i \triangleleft \{\langle N_j \oplus N_c \rangle_{h(N_i \| HP_i)}\}$.

Using A1, A13, S14, and the M-M rule, we get

**S43**: $U_i \mid\equiv CS \mid\sim (N_j \oplus N_c)$.

Using A21, S43, and the N-V rule, we get

**S44**: $U_i \mid\equiv CS \mid\equiv (N_j \oplus N_c)$.

Using A22, S44, and the jurisdiction rule, we get

**S45**: $U_i \mid\equiv (N_j \oplus N_c)$.

Using A1, A18, S41, S45, and the belief rule, we get

**S46**: $U_i \mid\equiv (N_i, N_j \oplus N_c, k_i.k_j)$.

**S47**: $U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G1**)

Using A1, S47, and the SK rule, we get

**S48**: $U_i \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G4**)

According to message M5 and using the seeing rule, we get

**S49**: $S_j \triangleleft \{\alpha : \langle k_i \rangle_{k_j}; MAC_1 : \langle B_7, SID_j, k_i, N_j, T_c \rangle_{k_j}; B_9 : \langle N_i \oplus N_c \rangle_{h(N_j \| r_S)}; T_c\}$.

Using seeing rule for components we get

**S50**: $S_j \triangleleft \{\langle B_7, SID_j, k_i, N_j, T_c \rangle_{k_j}\}$.

Using A23, S50, and the M-M rule, we get

**S51**: $S_j \mid\equiv CS \mid\sim (B_7, SID_j, k_i, N_j, T_c)$.

Using A2, S51, the freshness rule, and the N-V rule, we get

**S52**: $S_j \mid\equiv CS \mid\equiv (B_7, SID_j, k_i, N_j, T_c)$.

Using the belief rule for components we get

**S53**: $S_j \mid\equiv CS \mid\equiv k_i$.

Using A20, S53, and the N-V rule, we get

**S54**: $S_j \mid\equiv k_i$.

According to S49 and using the seeing rule, we get

**S55**: $S_j \triangleleft \{\langle N_i \oplus N_c \rangle_{h(N_j \| r_S)}\}$.

Using A2, A17, S32, and the M-M rule, we get

**S56**: $S_j \mid\equiv CS \mid\sim (N_i \oplus N_c)$.

Using A24, S56, and the N-V rule, we get

**S57**: $S_j \mid\equiv CS \mid\equiv (N_i \oplus N_c)$.

Using A25, S57, and the jurisdiction rule, we get

**S58**: $S_j \mid\equiv (N_i \oplus N_c)$.

Using A2, A23, S54, S58, and the belief rule, we get

**S59**: $S_j \mid\equiv (N_j, N_i \oplus N_c, k_i.k_j)$.

**S60**: $S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G2**)

Using A2, S60, and the SK rule, we get

**S61**: $S_i \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.(**G5**)

### C. INFORMAL SECURITY ANALYSIS

#### 1) PERFECT FORWARD SECRECY (PFS)

PFS is a feature of key agreement protocol, and the feature is becoming increasingly important in the protocol. PFS requires that if the long-term key is revealed to $\mathcal{A}$, $\mathcal{A}$ still cannot compute the $SK$ between $U_i$, $S_j$, and $CS$, which is secure.

Assume that $\mathcal{A}$ wants to compute session key $SK_i = SK_j = SK_c$, by $SK = H(N_j \oplus N_c \oplus N_i(k_i \| k_j))$. The attacker starts computing the session key after obtaining the smart card, information about the public channel, and $x$.

First attacker can compute $N_i = B_2 \oplus H(PID_i \| x)$ and $N_j = B_5 \oplus H(SID_j \| x)$. Then $\mathcal{A}$ needs to compute another random number $N_c$ ($N_c = N_j \oplus B_8 \oplus H(N_i \| HP_i)$, $N_c = N_i \oplus B_9 \oplus H(N_j \| r_S)$ ). However, these two parameters $HP_i$, $r_S$ are not available to $\mathcal{A}$. That is, the attacker cannot compute $SK$. The modified protocol can provide PFS.

#### 2) PRIVILEGED-INSIDER ATTACKS (PIA)

Let assume there is a malicious $U_i$ who tries to convince $CS$ that $S_j$ is willing to communicate with him. $U_i$ keeps two sets of $\{D_1, PID_i\}$, namely $D_1$, $PID_i$ and $D_1{}'$, $PID_i{}'$, when running two logins with other $S_j{}'$. $U_i$ now prepares his message $M_1$ faithfully using the old method. Then, this malicious $U_i$ will create the message $PSID_j$, $B_5$, $B_6$, $B_7$ as follows:

- This $U_i$ selects a random number $N_j$ and a timestamp $T_j$.
- The malicious $U_i$ sets $PSID_j = PID_i{}'$.
- The malicious $U_i$ sets $B_5 = D_1{}' \oplus h(ID_i \parallel HP_i) \oplus N_j = h(PID_i{}' \parallel x) \oplus N_j$
- The malicious $U_i$ sets $B_6 = h(PID_i{}' \parallel N_j) \oplus SID_j$
- The malicious $U_i$ sets $B_7 = h(N_j \parallel SID_j \parallel B_4 \parallel T_j)$

The malicious user sends the above-computed message along with $M_1$ to the $CS$. The $CS$ computes $k_i = H(N_i \| ID_i HP_i)$ and $k_j = H(SID_j \| N_j \| r_S)$ for mutual authentication. However, $r_S$ is a secret value between the $S$ and $CS$, and the user cannot get this value. There is no way to complete mutual authentication, and our proposed protocol can protect against malicious users.

#### 3) STOLEN SMART CARD (SSC) ATTACKS

Assuming SC is stolen, the $\mathcal{A}$ can extract $(PID_i, A_i, D_1, D_3, h(\cdot))$. However, we know that $N_i, N_j, N_c, k_i, k_j$ are needed to calculate the session key, where $N_i = B_2 \oplus h(PID_i \| x)$, $N_j^* = B_5 \oplus h(SID_j \| x)$, $k_i = h(N_i^* \| ID_i^* \| HP_i)$, $k_j = h(SID_j \| N_j \| r_S)$. Therefore, the $\mathcal{A}$ cannot learn any information after obtaining SC, which means that the proposed protocol can resist SSC attacks.

#### 4) OFF-LINE PASSWORD GUESSING (OPG) ATTACKS

Assume that $\mathcal{A}$ stole $U_i$'s SC and wants to guess $PW_i$ by comparing the parameter $A_i = h(h(ID_i) \oplus HP_i)$, $HP_i$ computed by $HP_i = h(PW_i \parallel b_i)$. In other words, the attacker needs to guess the $ID_i$, $PW_i$, and $b_i$ together, which is impossible, so our protocol can resist OPK attacks. Similarly, the identity cannot be guessed.

#### 5) MUTUAL AUTHENTICATION (MA)

MA requires that entities across the entire network environment can authenticate each other as legitimate and secure. In our proposed protocol, the authentication values include $\{B_4, B_7, MAC_1, MAC_2\}$, and these values are calculated using the secret $\{x, HP_i, r_S\}$. These secrets are assigned during the registration phase. This scheme can provide MA. The establishment of the session key is the reason for the user to perform the authentication protocol. The successful establishment of the session key can ensure the security of the subsequent communication. After the mutual authentication is completed, the user computes the verification value $\{MAC_3 = h(SK_i \parallel b_j)\}$ and sends this value to the server. If the verification $MAC_3 =?h(SK_j||b_j)$ holds, it verifies that $SK_i = SK_j$. Hence, this protocol can complete MA and session key verification.

#### 6) REPLAY ATTACKS

In our protocol, there are random numbers and timestamps in every transmitted message, where $\mathcal{A}$ cannot obtain the random number $N_i, N_j, N_c$ from the public channel. After each message is received, the timestamp T is validated. Subsequent calculations are performed only if the timestamp is valid. As a result, $\mathcal{A}$ cannot replay the messages without a valid timestamp and the random number, hence, our protocol can resist replay attack.

#### 7) KNOWN SESSION-SPECIFIC TEMPORARY INFORMATION (KSSTI) ATTACKS

Assume that the temporary information $N_i$ is obtained by $\mathcal{A}$. The session key is not only computed by random values; it also contains private information $(HP_i, r_S)$. There is no way for $\mathcal{A}$ to compute additional values, so this protocol can resist KSSTI attacks.

#### 8) NO KEY CONTROL PROPERTY

Neither party can control the key negotiation process to compute $SK$ separately, where $SK_i = SK_j = SK_c = h(N_i \oplus N_j \oplus N_c(k_i \| k_j))$. The details are as follows:

- $N_i$, $N_j$, and $N_c$ are random numbers independently selected by each entity.
- If $U_i$ does not know $k_j$, which is contributed by $S_j$, $U_i$ cannot compute $SK_i$. Similarly, $S_j$ cannot compute $SK_j$ without the value $k_i$ from $U_i$.

#### 9) USER ANONYMITY

In our scheme, the pseudo-identity $PID_i$ is used instead of the original $ID_i$. The pseudo-identities are updated after each communication. Additionally, all messages transmitted on a public channel $\{M_1, M_2, M_3, M_4, MAC_3\}$ are refreshed using the random numbers $\{N_i, N_j, N_c\}$. Because the hash function is a one way function, there is no way to calculate $ID_i$ by $MAC_3 = h(k_i \parallel B_4 \parallel N_i \parallel ID_i \parallel k_j \parallel T_c)$. Then because $N_i$ is secret, the attacker has no way to compute $ID_i$ by $ID_i = B_3 \oplus h(PID_i \parallel N_i)$. Hence, $\mathcal{A}$ cannot extract $ID_i$ from exchanged messages.

**TABLE 2. Comparisons of security.**

| | (a) | (b) | (c) | (d) | (e) | (f) | (g) | (h) | (i) |
|---|---|---|---|---|---|---|---|---|---|
| Irshad et al. [43] | N | Y | Y | Y | - | - | - | N | Y |
| Amin et al. [45] | N | Y | Y | N | - | - | - | - | - |
| Wu et al. [44] | Y | Y | Y | Y | N | Y | N | N | Y |
| Our | Y | Y | Y | Y | Y | Y | Y | Y | Y |

The protocol is secured against (a) user anonymity, (b) mutual authentication, (c) no key control property, (d) OPG attacks, (e) SSC attacks, (f) KSSTI attacks, (g) perfect forward secrecy, (h) PI attacks, and (i) replay attacks. The "-" denotes that the protocol does not use the related factor. The "Y" denotes that this protocol can resist the attack. The "N" denotes that the protocol has suffered the attack.

## V. SECURITY PERFORMANCE COMPARISONS

This section is used to compare the security and performance of our protocol with related protocols, such as Wu *et al.* [44], Amin *et al.* [45], and Irshad *et al.* [43]. Due to the smaller number of actual uses, we did not calculate the registration phase when comparing.

### A. SECURITY COMPARISONS

Table 2 shows the comparisons of our research with some of the latest lightweight authentication schemes in terms of safety performance. Obviously, our protocol is superior to all protocols.

### B. PERFORMANCE COMPARISONS

There are two operations in our scheme: hash function and XOR. Compared to the hash operation, the XOR operation cost is negligible. This paper ignores the XOR operation in its performance analysis. We use the symbols $t_h$ and $t_c$ to represent the time of the hash function and the time of the Chebyshev chaotic map, respectively. Through [44], we know that the time cost of one hash function is 0.005174 ms, and the time cost of one Chebyshev chaotic map is 127.042 ms ($t_h \approx 0.005174\ ms$, $t_c \approx 127.042\ ms$).

Table 3 depicts the results of the computational costs of the different protocols (Irshad *et al.* [43], Amin *et al.* [45], and Wu *et al.* [44]). The comparison scheme is a three-party key agreement and identity authentication protocol, so the calculation cost of each party is listed. It can be clearly seen that the cost of Irshad *et al.*'s scheme is relatively high, and this scheme is not safe. Amin *et al.*'s scheme is the least expensive, but their solution is vulnerable to OPG attacks and KSSTI attacks, and it does not guarantee UA and PFS. Similarly, Wu *et al.*'s scheme has cost a few hash operations relative to our protocol, but their protocols have many security issues, such as PFS, malicious user attacks, and SSC attacks. Therefore, the security assessment in Table 2 indicates that the proposed protocol is not affected by the attacks and weaknesses suffered by earlier schemes.

**TABLE 3. Efficiency comparison.**

| | scheme | | | |
|---|---|---|---|---|
| | Irshad et al. [43] | Amin et al. [45] | Wu et al [44] | our |
| $U_i$ (ms) | $3t_c + 4t_h \approx 381.14619$ | $9t_h \approx 0.04656$ | $11t_h \approx 0.05691$ | $13t_h \approx 0.06726$ |
| $S_j$ (ms) | $2t_c + 4t_h \approx 254.10469$ | $4t_h \approx 0.02069$ | $6t_h \approx 0.03104$ | $8t_h \approx 0.04139$ |
| $CS$ (ms) | $t_c + 6t_h \approx 127.07304$ | $10t_h \approx 0.05174$ | $19t_h \approx 0.09831$ | $19t_h \approx 0.09831$ |
| Total (ms) | $6t_c + 14t_h \approx 762.32392$ | $23t_h \approx 0.11899$ | $36t_h \approx 0.18626$ | $40t_h \approx 0.20696$ |
| Security | User anonymity, Privileged-insider attacks from [44] | User anonymity, Off-line password guessing attacks from [44] | Stolen smart card attacks, Perfect forward secrecy, Privileged-insider attacks | Provably secure |

In Table 3, our protocol is only a few $t_h$ and $t_{xor}$ more than Wu *et al.*'s protocol. In practice, these operations are trivial, and the solution has security problems.

## VI. CONCLUSION

In this paper, we first review the definition and importance of 5G and IoT. Then, we reviewed the authentication protocol of Wu *et al.* and proved that their protocol have some security issues, such as perfect forward secrecy and privileged-insider attacks. To address these security weaknesses, we propose an enhanced protocol based on M-S architecture in a 5G network environment. Through formal security analysis, we show that our protocol can resist such various attacks. Finally, the comparison of security and performance shows that the protocol improved in this paper has better performance and higher security.

## ACKNOWLEDGMENT

The authors would like to thank the handled editor and reviewers for their great support and valuable suggestions.

## REFERENCES

[1] S. Li, L. Da Xu, and S. Zhao, "5G Internet of Things: A survey," *J. Ind. Inf. Integr.*, vol. 10, pp. 1–9, Jun. 2018.

[2] M. Obaidat and N. Boudriga, *Security of E-systems and Computer Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[3] H. Xiong, Y. Zhao, L. Peng, H. Zhang, and K.-H. Yeh, "Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing," *Future Gener. Comput. Syst.*, vol. 97, pp. 453–461, Aug. 2019.

[4] T.-Y. Wu, C.-M. Chen, K.-H. Wang, C. Meng, and E. K. Wang, "A provably secure certificateless public key encryption with keyword search," *J. Chin. Inst. Eng.*, vol. 42, no. 1, pp. 20–28, Jan. 2019.

[5] L. Ni, F. Tian, Q. Ni, Y. Yan, and J. Zhang, "An anonymous entropy-based location privacy protection scheme in mobile social networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 93, 2019.

[6] J.-S. Pan, C.-Y. Lee, A. Sghaier, M. Zeghid, and J. Xie, "Novel systolization of subquadratic space complexity multipliers based on toeplitz matrix–vector product approach," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 7, pp. 1614–1622, Jul. 2019.

[7] X. Wang, S.-J. Ji, Y.-Q. Liang, H.-F. Leung, and D. K. Chiu, "An unsupervised strategy for defending against multifarious reputation attacks," *Appl. Intell.*, vol. 49, no. 12, pp. 4189–4210, Dec. 2019.

[8] M. S. Obaidat, I. Traore, and I. Woungang, *Biometric-Based Physical and Cybersecurity Systems*. Cham, Switzerland: Springer, 2019.

[9] J. M.-T. Wu, J. C.-W. Lin, and A. Tamrakar, "High-utility itemset mining with effective pruning strategies," *ACM Trans. Knowl. Discov. Data*, vol. 13, no. 6, pp. 1–22, Nov. 2019.

[10] Z. Zhao, C. Li, X. Zhang, F. Chiclana, and E. H. Viedma, "An incremental method to detect communities in dynamic evolving social networks," *Knowl.-Based Syst.*, vol. 163, pp. 404–415, Jan. 2019.

[11] Z. Meng and J.-S. Pan, "HARD-DE: Hierarchical archive based mutation strategy with depth information of evolution for the enhancement of differential evolution on numerical optimization," *IEEE Access*, vol. 7, pp. 12832–12854, 2019.

[12] Z. Meng, J.-S. Pan, and K.-K. Tseng, "PaDE: An enhanced differential evolution algorithm with novel control parameter adaptation schemes for numerical optimization," *Knowl.-Based Syst.*, vol. 168, pp. 80–99, Mar. 2019.

[13] J.-S. Pan, P. Hu, and S.-C. Chu, "Novel parallel heterogeneous metaheuristic and its communication strategies for the prediction of wind power," *Processes*, vol. 7, no. 11, p. 845, Nov. 2019.

[14] J.-S. Pan, L. Kong, T.-W. Sung, P.-W. Tsai, and V. Snášel, "$\alpha$-fraction first strategy for hierarchical model in wireless sensor networks," *J. Internet Technol.*, vol. 19, no. 6, pp. 1717–1726, 2018.

[15] J. Wang, X. Gu, W. Liu, A. K. Sangaiah, and H.-J. Kim, "An empower hamilton loop based data collection algorithm with mobile agent for WSNs," *Hum.-Centric Comput. Inf. Sci.*, vol. 9, no. 1, p. 18, Dec. 2019.

[16] J. Wang, Y. Gao, K. Wang, A. K. Sangaiah, and S.-J. Lim, "An affinity propagation-based self-adaptive clustering method for wireless sensor networks," *Sensors*, vol. 19, no. 11, p. 2579, Jun. 2019.

[17] B. Ying and A. Nayak, "Lightweight remote user authentication protocol for multi-server 5G networks using self-certified public key cryptography," *J. Netw. Comput. Appl.*, vol. 131, pp. 66–74, Apr. 2019.

[18] E. Borcoci, T. Ambarus, J. Bruneau-Queyreix, D. Negru, and J. M. Batalla, "Optimization of multi-server video content streaming in 5G environment," in *Proc. 8th Int. Conf. Evolving Internet*, Barcelona, Spain, 2016.

[19] L. Lamport, "Password authentication with insecure communication," *Commun. ACM*, vol. 24, no. 11, pp. 770–772, Nov. 1981.

[20] T.-Y. Hwang, "Password authentication using public-key encryption," in *Proc. Int. Carnahan Conf. Secur. Technol.*, 1983, pp. 35–38.

[21] S.-P. Shieh, W.-H. Yang, and H.-M. Sun, "An authentication protocol without trusted third party," *IEEE Commun. Lett.*, vol. 1, no. 3, pp. 87–89, May 1997.

[22] M. Sandirigama, A. Shimizu, and M. T. Noda, "Simple and secure password authentication protocol (SAS)," *IEICE Trans. Commun.*, vol. E83-B, no. 6, pp. 1363–1365, 2000.

[23] C.-C. Chang and T.-C. Wu, "Remote password authentication with smart cards," *IEICE Proc. E, Comput. Digit. Techn.*, vol. 138, no. 3, pp. 165–168, 1991.

[24] M.-S. Hwang and L.-H. Li, "A new remote user authentication scheme using smart cards," *IEEE Trans. Consum. Electron.*, vol. 46, no. 1, pp. 28–30, Feb. 2000.

[25] C.-C. Chang and K.-F. Hwang, "Some forgery attacks on a remote user authentication scheme using smart cards," *Informatica*, vol. 14, no. 3, pp. 289–294, 2003.

[26] S.-W. Lee, H.-S. Kim, and K.-Y. Yoo, "Improvement of Chien's remote user authentication scheme using smart cards," *Comput. Standards Interfaces*, vol. 27, no. 2, pp. 181–183, 2005.

[27] B.-L. Chen, W.-C. Kuo, and L.-C. Wuu, "Robust smart-card-based remote user password authentication scheme," *Int. J. Commun. Syst.*, vol. 27, no. 2, pp. 377–389, Feb. 2014.

[28] X. Li, J. Niu, M. Khurram Khan, and J. Liao, "An enhanced smart card based remote user password authentication scheme," *J. Netw. Comput. Appl.*, vol. 36, no. 5, pp. 1365–1371, Sep. 2013.

[29] L.-H. Li, L.-C. Lin, and M.-S. Hwang, "A remote password authentication scheme for multiserver architecture using neural networks," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1498–1504, Nov. 2001.

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

[31] X. Cao and S. Zhong, "Breaking a remote user authentication scheme for multi-server architecture," *IEEE Commun. Lett.*, vol. 10, no. 8, pp. 580–581, Aug. 2006.

[32] C.-C. Chang and J.-S. Lee, "An efficient and secure multi-server password authentication scheme using smart cards," in *Proc. Int. Conf. Cyberworlds*, Dec. 2004, pp. 417–422.

[33] S. Ghosh, A. Majumder, J. Goswami, A. Kumar, S. P. Mohanty, and B. K. Bhattacharyya, "Swing-pay: One card meets all user payment and identity needs: A digital card module using NFC and biometric authentication for peer-to-peer payment," *IEEE Consum. Electron. Mag.*, vol. 6, no. 1, pp. 82–93, Jan. 2017.

[34] X. Duan and X. Wang, "Fast authentication in 5G HetNet through SDN enabled weighted secure-context-information transfer," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[35] C.-C. Lee, T.-H. Lin, and R.-X. Chang, "A secure dynamic ID based remote user authentication scheme for multi-server environment using smart cards," *Expert Syst. Appl.*, vol. 38, no. 11, pp. 13863–13870, 2011.

[36] X. Li, Y. Xiong, J. Ma, and W. Wang, "An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards," *J. Netw. Comput. Appl.*, vol. 35, no. 2, pp. 763–769, Mar. 2012.

[37] K. Xue, P. Hong, and C. Ma, "A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture," *J. Comput. Syst. Sci.*, vol. 80, no. 1, pp. 195–206, Feb. 2014.

[38] S. Jangirala, S. Mukhopadhyay, and A. K. Das, "A multi-server environment with secure and efficient remote user authentication scheme based on dynamic ID using smart cards," *Wireless Pers. Commun.*, vol. 95, no. 3, pp. 2735–2767, Aug. 2017.

[39] S. Shunmuganathan, R. D. Saravanan, and Y. Palanichamy, "Secure and efficient smart-card-based remote user authentication scheme for multi-server environment," *Can. J. Elect. Comput. Eng.*, vol. 38, no. 1, pp. 20–30, 2015.

[40] H. Zhu, "Flexible and password-authenticated key agreement scheme based on chaotic maps for multiple servers to server architecture," *Wireless Pers. Commun.*, vol. 82, no. 3, pp. 1697–1718, Jun. 2015.

[41] J.-L. Tsai and N.-W. Lo, "A privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE Syst. J.*, vol. 9, no. 3, pp. 805–815, Sep. 2015.

[42] H. Xiong and Z. Qin, "Revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1442–1455, Jul. 2015.

[43] A. Irshad, H. F. Ahmad, B. A. Alzahrani, M. Sher, and S. A. Chaudhry, "An efficient and anonymous chaotic map based authenticated key agreement for multi-server architecture," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 12, pp. 5572–5595, 2016.

[44] F. Wu, X. Li, L. Xu, A. K. Sangaiah, and J. J. Rodrigues, "Authentication protocol for distributed cloud computing: An explanation of the security situations for Internet-of-Things-enabled devices," *IEEE Consum. Electron. Mag.*, vol. 7, no. 6, pp. 38–44, Nov. 2018.

[45] R. Amin, N. Kumar, G. Biswas, R. Iqbal, and V. Chang, "A light weight authentication protocol for IoT-enabled devices in distributed cloud computing environment," *Future Gener. Comput. Syst.*, vol. 78, pp. 1005–1019, Jan. 2018.

[46] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," *Proc. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 426, no. 1871, pp. 233–271, 1989.

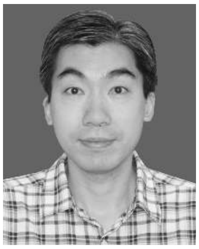
[47] R. Amin and G. Biswas, "A secure light weight scheme for user authentication and key agreement in multi-gateway based wireless sensor networks," *Ad Hoc Netw.*, vol. 36, pp. 58–80, Jan. 2016.

[48] C.-M. Chen, L. Xu, K.-H. Wang, S. Liu, and T.-Y. Wu, "Cryptanalysis and improvements on three-party-authenticated key agreement protocols based on chaotic maps," *J. Internet Technol.*, vol. 19, no. 3, pp. 679–687, 2018.

[49] C.-T. Li, T.-Y. Wu, and C.-M. Chen, "A provably secure group key agreement scheme with privacy preservation for online social networks using extended chaotic maps," *IEEE Access*, vol. 6, pp. 66742–66753, 2018.

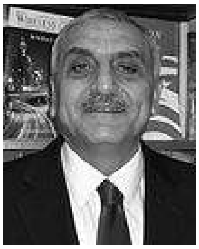
[50] C.-M. Chen, B. Xiang, Y. Liu, and K.-H. Wang, "A secure authentication protocol for Internet of vehicles," *IEEE Access*, vol. 7, pp. 12047–12057, 2019.

[51] C.-M. Chen, K.-H. Wang, K.-H. Yeh, B. Xiang, and T.-Y. Wu, "Attacks and solutions on a three-party password-based authenticated key exchange protocol for wireless communications," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 8, pp. 3133–3142, Aug. 2019.



**TSU-YANG WU** received the Ph.D. degree from the Department of Mathematics, National Changhua University of Education, Taiwan. He was an Assistant Professor with the Harbin Institute of Technology, Shenzhen Campus. He is currently an Associate Professor with the College of Computer Science and Engineering, Shandong University of Science and Technology, China. He serves as the Executive Editor for the *Journal of Network Intelligence* (JNI) and as an Associate Editor in the *Data Science and Pattern Recognition* (DSPR). His research interests include cryptography and network security.

**ZHIYUAN LEE** is currently pursuing the master's degree with the College of Computer Science and Engineering, Shandong University of Science and Technology, China. His research interest include applied cryptography.



**MOHAMMAD S. OBAIDAT** (Life Fellow, IEEE) received the Ph.D. degree in computer engineering, with a minor in computer science, from the Electrical and Computer Engineering (ECE) Department, The Ohio State University, Columbus, OH, USA. He is currently an internationally known academic/researcher/scientist/ scholar. He has received extensive research funding and published to date (2018) over 850 refereed technical articles–about half of them are journal articles, over 65 books, and over 65 book chapters. He is Editor-in-Chief of three scholarly journals and an editor of many other international journals. He is the founding Editor-in-Chief of the *Wiley Security and Privacy Journal*.

**SARU KUMARI** received the Ph.D. degree in mathematics from CCS University, Meerut, India, in 2012. She has published more than 141 research articles in reputed International journals and conferences, including 122 publications in SCI-Indexed Journals. She is on the Editorial board of the *AEÜ - International Journal of Electronics and Communications* (Elsevier) (SCI); the *International Journal of Communication Systems* (Wiley) SCI-E; the *Telecommunication Systems* (Springer) SCI; the *Human Centric Computing and Information Sciences* (Springer) SCI-E; the *Transactions on Emerging Telecommunications Technologies* (Wiley) (SCI-E); the *Information Technology and Control*, Kaunas University of Technology, Lithuania (SCI-E); the *KSII Transactions on Internet and Information Systems* (SCI-E), published from Taiwan; the *Information Security: A Global Perspective, Taylor & Francis* (ESCI, Scopus); the *International Journal of Wireless Information Networks* (ESCI, Scopus), Springer; the *Journal of Reliable Intelligent Environments* (Springer) (ESCI, Scopus); the *Security and Privacy* (Wiley); the *Iran Journal of Computer Science* (Springer); and the *Azerbaijan Journal of High Performance Computing*, published by Azerbaijan State Oil and Industry University, Azerbaijan. She is the Technical Program Committee Member for more than a dozen of International conferences. She is a reviewer of more than 50 reputed Journals, including the *SCI-Indexed Journals of IEEE*, Elsevier, Springer, and Wiley. Her current research interests include information security and applied cryptography. She has served as the Guest Editor of the Special Issue Big-Data and IoT in e-Healthcare for Computers and Electrical Engineering (Elsevier) (SCI-E), Elsevier.

**SACHIN KUMAR** received the Ph.D. degree in computer science from CCS University, Meerut, in 2007. He has been working as a Professor with the Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College (AKGEC), Ghaziabad, since October 2011. Prior to joining AKGEC, he has worked with the Raj Kumar Goel Institute of Technology (RKGIT) Ghaziabad, Krishna Institute of Engineering Technology (KIET), Ghaziabad, and CCS University, Meerut. He has more than 18 years of academic experience. He has guided four Ph.D. students and ten M.Tech. students. He has published/presented several articles in journals/conferences of repute. He is the author/coauthor of three books of computer science.

**CHIEN-MING CHEN** is currently an Associate Professor with the School of Computer Science, Harbin Institute of Technology (Shenzhen), Shenzhen, China. He has published more than 70 reputed international journal, including 53 publications in SCI-indexed journals. His current research interests include network security, the mobile internet, wireless sensor networks, and cryptography. He is currently an Associate Editor of IEEE Access and an Executive Editor of the *International Journal of Information and Computer Security*.