PAPER

SLARS: Secure Lightweight Authentication for Roaming Service in Smart City

Hakjun LEE[†a)], *Nonmember*

SUMMARY Smart cities aim to improve the quality of life of citizens and efficiency of city operations through utilization of 5G communication technology. Based on various technologies such as IoT, cloud computing, artificial intelligence, and big data, they provide smart services in terms of urban planning, development, and management for solving problems such as fine dust, traffic congestion and safety, energy efficiency, water shortage, and an aging population. However, as smart city has an open network structure, an adversary can easily try to gain illegal access and perform denial of service and sniffing attacks that can threaten the safety and privacy of citizens. In smart cities, the global mobility network (GLOMONET) supports mobile services between heterogeneous networks of mobile devices such as autonomous vehicles and drones. Recently, Chen et al. proposed a user authentication scheme for GLOMONET in smart cities. Nevertheless, we found some weaknesses in the scheme proposed by them. In this study, we propose a secure lightweight authentication for roaming services in a smart city, called SLARS, to enhance security. We proved that SLARS is more secure and efficient than the related authentication scheme for GLOMONET through security and performance analysis. Our analysis results show that SLARS satisfies all security requirements in GLOMONET and saves 72.7% of computation time compared to that of Chen et al.'s scheme.

key words: *heterogeneous networks, mobile communication, multi-factor authentication, authentication protocols, roaming*

1. Introduction

With the emergence of IoT technology, a smart city aims to improve the efficiency of city operation and the quality of life of its citizens through the integration of 5G communication technology with the city's physical infrastructure [1]. A smart city is a new model that utilizes IoT, cloud computing, artificial intelligence, and big data to provide smart services for urban planning, development, and management, aiming to address various challenges such as fine dust, traffic congestion, safety, energy efficiency, water shortage, and an aging population [2]–[4].

To efficiently respond to various events occurring in a city, the scale of smart cities is growing by linking data between city services and regions [5]. When transmitting data, such as images, energy, traffic, health, and payment, generated by various devices in the city, it is necessary to encrypt sensitive information. However, a smart city's open network structure, linked with a cyber-physical system (CPS), can be targeted by adversaries. Illegal access, denial of service, and sniffing attacks can threaten the safety and privacy of citizens [6].

As the demand for mobile services in wireless networks increases in smart cities, authentication schemes for global mobility network (GLOMONET) that support the mobility of terminals, such as mobile devices, autonomous vehicles, and drones, have been proposed [7]–[9]. Global roaming is a service that allows users to move across heterogeneous networks in a smart city. A mobile user (MU) can then access the foreign network. In a mobile network, users can freely move to a foreign network managed by a foreign agent (FA) with the help of a home network and then access smart city services. In other words, an FA performs a mutual authentication using a home agent (HA) to verify the legitimacy of the user. The scenario of GLOMONET in a smart city is shown in Fig. 1. When a security breach occurs in the CPS of a smart city, it can cause paralysis of the city administration, financial damages, and even prove fatal for the public, industry, and citizens. Therefore, reliable security must be applied across data creation, collection, storage, analysis, sharing, and deletion stages. Thus, the authentication of citizens or IoT devices, the subjects of data generation, is essential.

Recently, various roaming authentication schemes have been proposed to support GLOMONET. Zhu [10] first proposed a user-authentication scheme for roaming in a wireless network. Using a temporary certificate, MU and FA establish a session key. However, Lee et al. [11] reported that the scheme proposed by Zhu et al. [10] was vulnerable to forgery attacks and did not achieve perfect backward secrecy and mutual authentication. Subsequently, Lee et al. [11] presented a new enhancement using hashes, XOR, and symmetric-key cryptography. It was more efficient than the scheme proposed by Zhu et al. [10]. Wu et al. [12] found that Lee et al.'s [11] scheme did not achieve perfect backward secrecy and anonymity and they proposed a new authentication scheme. However, Xu et al. [13] and Lee et al. [14] reported that the scheme proposed by Wu et al. [13] did not guarantee anonymity. Kang et al. [15] proposed an enhancement to improve the security of WSN. However, Karuppia and Saravana [16] found that Kang et al.'s scheme [15] was vulnerable to user impersonation and off-line password guessing attacks, and did not ensure user anonymity and perfect forward secrecy. Karuppia and Saravana [16] proposed a user authentication scheme in GLOMONET for roaming services by improving Kang et al.'s scheme [15]. In addition, they argued that the results of the comparative analysis show that their proposed scheme is more secure than

---

Manuscript received January 5, 2024.
Manuscript revised March 19, 2024.
Manuscript publicized April 16, 2024.

[†]Department of Computer Engineering at Kyungnam University, South Korea.
a) E-mail: hakjun@kyungnam.ac.kr

DOI: 10.23919/transcom.2024EBP3004

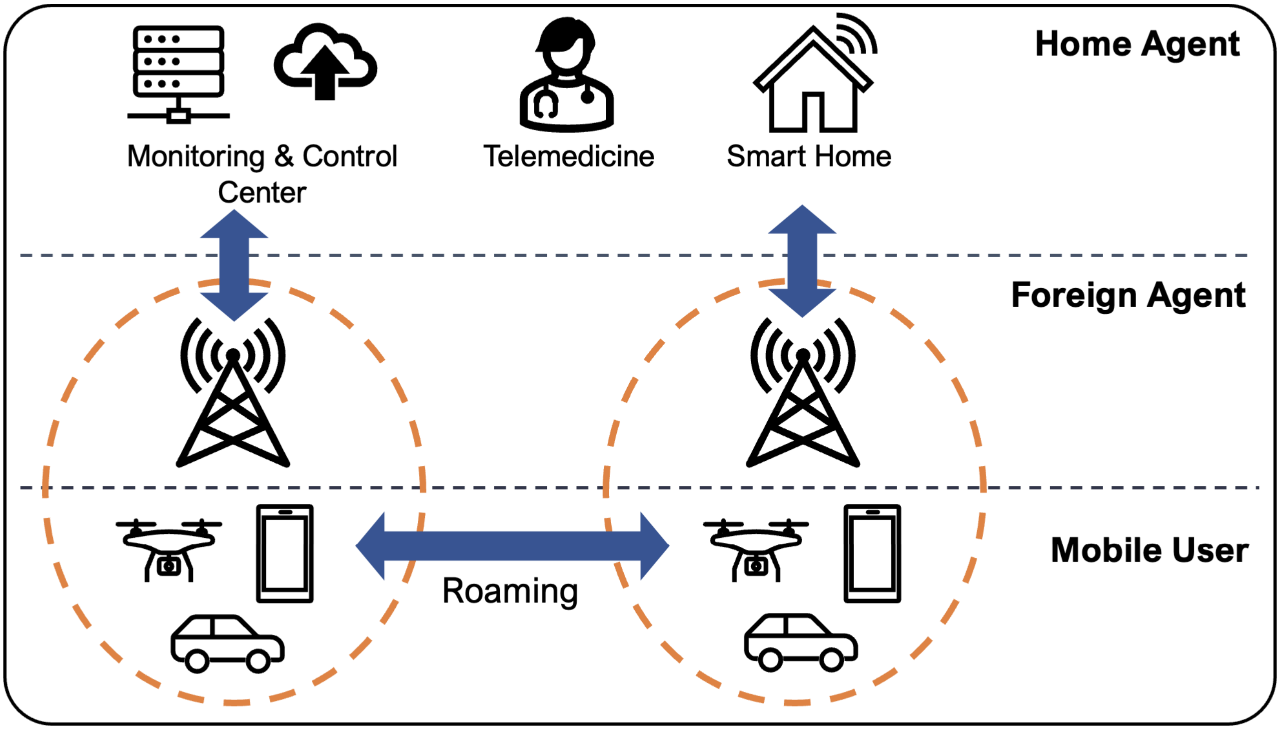


Fig. 1 Scenario of GLOMONET in smart city.

other related-schemes. However, Li et al. [17] observed that the scheme proposed by Karuppia and Saravana [16] still had several weaknesses such as imperfect forward secrecy, session key leakage, and no session key updates.

As the concept of a smart city is being discussed along with Industry 4.0, user authentication methods in smart cities are also being studied. For GLOMONET in smart cities, Li et al. [17] improved the scheme of Karuppia and Saravana [16] and presented a new user authentication and key agreement scheme to guarantee user anonymity using elliptic curve cryptography (ECC). Although they achieved a significant improvement in efficiency compared with the schemes of Kang et al. [15] and Karuppia and Saravana [16], Xie and Hwang [18] reported that Li et al.'s scheme [17] was vulnerable to impersonation attacks and did not protect against offline password-guessing attacks. Xie and Hwang [18] proposed a new authentication scheme to enhance the security of GLOMONET in smart cities. Recently, Chen et al. [19] have reported that the scheme proposed by Xie et al. [18] does not support session key updates and locally verifies the identity of the user. To supplement this and increase performance efficiency, they proposed a new ECC-based authentication method. Nevertheless, we found that the scheme of Chen et al. [19] is still weak. Therefore, in this study, we propose a new authentication method to achieve safe user roaming services in smart cities.

The contributions of this study are as follows.

1. We first define the security requirements for GLOMONET. Then, we conduct a security analysis of Chen et al.'s scheme [19] and report that it is vulnerable to FA impersonation and known-session attack.
2. To enhance the security and efficiency of GLOMONET in smart cities, we propose a secure lightweight authentication for roaming service in smart cities called SLARS. We apply three-factor authentication to strengthen the privacy of mobile users and use only XOR and hash functions to improve performance efficiency.
3. Finally, we present the results of security and performance analysis of SLARS with the related authentication scheme for GLOMONET.

The remainder of this paper is organized as follows. Section 2 provides preliminary knowledge on this study and introduces the background of the topic. In Sects. 3 and 4, we review and cryptoanalysis of Chen et al.'s scheme, respectively. In Sect. 5, the proposed authentication scheme is presented. Section 6 provides a security analysis of the proposed scheme using a random oracle model, ProVerif, and BAN logic. In Sect. 7, we compare the performance of the proposed scheme with those of other schemes. Finally, in Sect. 8, the conclusions of this study are presented.

2. Preliminaries

In this section, we explain the network model, bio-hash function, and adversarial model. The notations used in this study are shown in Table 1.

2.1 Network Model and Authentication Process

The schemes for GLOMONET presented in this study are

Table 1 Notations.

Symbols	Description
MU_i, FA_j, HA_k	Mobile user, Foreign agent, Home agent
ID_U, ID_F, ID_H	Identity of MU_i, FA_j, HA_k
PW_U	Password of MU_i
BIO_U	Biometric of MU_i
x	Secret key of HA_k
r_x	Random numbers
K_{FH}	Pre-shared key between FA_j and HA_k
$h(\cdot)$	Hash function
$H(\cdot)$	bio-Hash function
SK_U	Session key of MU_i
SK_F	Session key of FA_j
$\|$	Concatenation
$\oplus$	XOR Operation
P	Point on elliptic curve

based on the following network model.

1. MU_i sends an authentication request to FA_j to use the roaming service.
2. After receiving the request, FA_j sends the request for verifying the legitimacy of MU_i's request to HA_k.
3. HA_k checks the request received from FA_j, authenticates MU_i, and responds to FA_j.
4. FA_j sends a response to MU_i, and then MU_i and FA_j mutually establish a session key.

2.2 Elliptic Curve Cryptography

ECC is based on the logarithm problems expressed in the point addition and multiplication of elliptic curves [20], [21]. An elliptic curve is given by $E_p(a,b) : y^2 = x^3 + ax + b \bmod p$ over a finite field F_p, where p is the prime order and $a, b \in F_p$ such that $p > 3$ and $4a^3 + 27b^2 \neq 0 \bmod p$. The point multiplication over $E_p(a,b)$ is defined through a repetitive addition as $P + P + \cdots + P$ (a times) $=aP$, where P is a point on $E_p(a,b)$ and $a \in F_p^*$ is a random integer. The security of ECC relies on the following assumption:

1. Elliptic curve discrete logarithm problem (ECDLP): Given $P, aP \in E_p(a,b)$, it is computationally infeasible to find a within polynomial time.
2. Elliptic curve computational Diffie-Hellman problem (ECCDHP): Given $aP, bP \in E_p(a,b)$, it is computationally infeasible to find abP in polynomial time.

2.3 Bio-Hash Function

Two-factor user authentication using pins, passwords, and tokens that can be forgotten or stolen is vulnerable to device theft and impersonation attacks. Recently, many researchers have applied biohases for three-factor user authentication [22]–[24]. In biometrics, the imprint biometric characteristics can vary because of external reasons such as dry or cracked skin or dust on the imprint sensor, causing a false rejection. To solve this problem, Jin et al. [29] proposed a two-factor authentication method based on the dot product between tokenized pseudorandom numbers and user-specific fingerprint characteristics. They created a set of user-specific compact codes called biohash codes. It uses a pseudo-random number of user-specific tokens to randomly map biometric features to binary strings.

2.4 Adversarial Model

In this study, we considered the following adversarial model [25]–[27]:

1. By controlling a public channel, an attacker can eavesdrop on messages between MU_i, FA_j and HA_k, and then modify or replay them to impersonate participants.
2. An attacker can perform a side-channel attack to extract information stored in the smart card.
3. An attacker can discover sensitive information by combining eavesdropping and the extracted messages.

3. Review of Chen et al.'s Scheme

Chen et al.'s scheme consists of MU registration, mutual authentication, password change, and session key update phases. In this section, we describe these schemes in detail.

3.1 MU Registration Phase

In this step, MU_i registers his/her identity with HA_k, and then HA_k issues secret parameters to be used for mutual authentication. The detailed registration process is as follows:

1. MU_i selects ID_U, PW_U, and random $r \in Z_q^*$, computes $FID_U = h(ID_U || r)$, and sends the registration request with $< ID_U, FID_U >$ to HA_k.
2. HA_k select a random $n_H \in Z_q^*$, computes $R_U = h(FID_U || x)$, and $DID_U = h(x) \oplus \{FID_U || n_H\}$, stores $< DID_U, R_U, h(\cdot), P, ID_H >$ in the smart card, and sends it to MU_i.
3. After receiving the smart card, MU_i computes $r^* = h(ID_U || PW_U) \oplus r$, $V_U = h(ID_U || PW_U || r)$, and $R_U^* = R_U \oplus r$, and finally stores $< r^*, V_U, R_U^*, DID_U, h(\cdot), P, ID_H >$ in the smart card.

3.2 Mutual Authentication Phase

In this step, the pre-registered MU_i performs mutual authentication to establish a session key with FA_j with the help of HA_k. The detailed process is as follows:

1. MU_i inputs ID_U and PW_U, computes $r = h(ID_U || PW_U) \oplus r^*$ and checks $V_U = h(ID_U || PW_U || r)$. If they are not equal, MU_i terminates this process; otherwise, MU_i computes $FID_U = h(ID_U || r)$, $R_U = R_U^* \oplus r$, $a = f(ID_U || T_{seed}) \in Z_q^*$, $Q_1 = aP$, and $Q_2 = h(DID_U || R_U || Q_1 || FID_U || ID_F || ID_H)$, and sends the message $M_1 =< ID_F, ID_H, DID_U, Q_1, Q_2 >$ to FA_j, where T_{seed} is seed to generate a random number and $f()$ is a number-generating function.
2. After receiving the message, FA_j selects a random $b \in Z_q^*$, computes $W_1 = bP$ and $W_2 =$

$h(M_1||W_1||K_{FH}||ID_F||ID_H)$, and sends the message $M_2 =< M_1, W_1, W_2 >$ to HA_k.

3. HA_k first checks $W_2 = h(M_1||W_1||K_{FH}||ID_F||ID_H)$. If they are not equal, HA_k terminates the process; otherwise, HA_k retrieves FID_U and n_H by calculating $\{FID_U, n_H\} = h(x) \oplus DID_U$ and computes $R_U = h(FID_U||x)$. HA_k then checks $Q_2 = h(DID_U||R_U||Q_1||FID_U||ID_F||ID_H)$. If they are not equal, HA_k terminates the process; otherwise, HA_k selects a random $n_H^{new} \in Z_q^*$, computes $DID_U^{new} = h(x) \oplus \{FID_U||n_H^{new}\}$, $E_1 = h(R_U) \oplus DID_U^{new}$, $E_2 = h(R_U||Q_1||W_1||DID_U^{new})$, and $E_3 = h(K_{FH}||Q_1||W_1||ID_F||ID_H||E_1||E_2)$, and sends the massage $M_3 =< Q_1, E_1, E_2, E_3 >$ to FA_j.
4. FA_j checks $E_3 = h(K_{FH}||Q_1||W_1||ID_F||ID_H||E_1||E_2)$. If they are not equal, FA_j terminates the process; otherwise, FA_j computes $SK_F = h(bQ_1)$ and $B_3 = h(SK_F||E_1||E_2)$, and sends the message $M_4 =< E_1, E_2, W_1, B_3 >$ to MU_i.
5. MU_i computes $DID_U^{new} = E_1 \oplus h(R_U)$ and $SK_U = h(aW_1)$, $SK_{ij} = h_1(PK_i||ID_j||K_3||K_4)$, and checks $B_3 = h(SK_U||E_1||E_2)$. If they are equal, MU_i and FA_j successfully establish the session key SK.

3.3 Password Change Phase

MU_i, who wants to change the password performs the following procedure locally:

1. MU_i inputs ID_U and PW_U, computes $r = h(ID_U||PW_U) \oplus r^*$ and checks $V_U = h(ID_U||PW_U|| r)$. If they are not equal, MU_i terminates the process; otherwise, MU_i selects a new password PW_U^{new}.
2. MU_i computes $r^{new*} = h(ID_U||PW_U^{new}) \oplus r$ and $V_U^{new} = h(ID_U||PW_U^{new}||r)$, and replace r^* and V_U with r^{new*} and V_U^{new}, respectively.

3.4 Session Key Update Phase

If MU_i is within the coverage of FA_j for a long period, the session key should be updated regularly. In this step, the process of updating an existing session key with a new one is explained.

1. MU_i selects a random $a_U \in Z_q^*$ and computes $H_1 = a_U P$, $H_2 = h(H_1||SK_U^{i-1})$ where SK^{i-1} is the $i-1$th session key. Then, MU_i sends $M_1 =< H_1, H_2 >$ to FA_j.
2. FA_j checks $H_2 = h(H_1||SK_F^{i-1})$. If they are equal, FA_j selects a random $b_F \in Z_q^*$, computes $J_1 = b_F P$, $SK_F^i = h(b_F H_1)$, and $J_2 = h(SK_F^i||SK_F^{i-1})$, and sends $M_2 =< J_1, J_2 >$ to MU_i.
3. After receiving the message, MU_i computes $SK_U^i = h(a_U J_1)$ and checks $J_2 = h(SK_U^i||SK_U^{i-1})$. If they are equal, then MU_i and FA_j successfully update the session key.

4. Cryptoanalysis of Chen et al.'s Scheme

This section describes in detail how Chen et al.'s scheme [19] is vulnerable to FA impersonation and ephemeral key known session-specific temporary information attacks.

4.1 FA Impersonation Attack

An attacker masquerading as FA_j can perform an impersonation attack in the following manner:

1. First, the attacker obtains M_2 and M_3 by eavesdropping the public messages.
2. The attacker generates a random number b^A, and uses Q_1, E_1, and E_2 included in M_3 to compute $SK^A = h(b^A Q_1) = h(b^A aP)$, $W_1^A = b^A P$, and $B_3^A = h(SK^A||E_1||E_2)$.
3. The attacker generates $M_4 =< E_1, E_2, W_1^A, B_3 >$ and transmits them to MU.
4. According to the authentication process, MU calculates $SKu = h(aW_1^A) = h(ab^A P)$ and then verifies the correctness of B_3. Finally, MU agrees with the session key of the attacker.

Therefore, Chen et al.'s scheme is prone to FA_j impersonation attack.

4.2 Key Known Session-Specific Temporary Information Attack

If the attacker determines the random values constituting the session key in some way, he/she can attempt to calculate it. In Chen et al.'s scheme, the session key was $SK = h(abP)$. Here, P is a public parameter in the ECC system. Therefore, if the attacker knows the random values a_U and b_F, he/she can easily compute the session key and pretend to be MU or FA_j. Therefore, Chen et al.'s scheme is prone to known session-specific temporary information attacks.

5. Proposed Scheme

In this section, we describe the proposed SLARS. It consists of MU registration, mutual authentication, password change, and session-key update phases.

5.1 MU Registration Phase

In the registration phase of SLARS, MU_i registers with HA_k and is issued encrypted secret parameters of HA_k to be used later to prove its identity. The detailed process, which is illustrated in Fig. 2, is as follows:

1. MU_i selects ID_U, PW_U, BIO_U and random $r \in Z_q^*$, computes $FID_U = h(ID_U||r)$ and $PB_U = h(PW_U||H(BIO_U))$, and sends the registration request with $< ID_U, FID_U >$ to HA_k.
2. HA_k selects a random $n_H \in Z_q^*$, computes $R_U =$

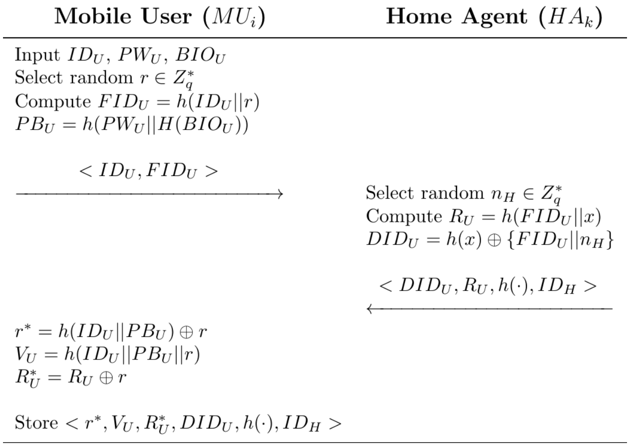


Fig. 2 MU registration phase of SLARS.

$h(FID_U||x)$, and $DID_U = h(x) \oplus \{FID_U||n_H\}$, stores $< DID_U, R_U, h(\cdot), ID_H >$ in the smart card, and sends it to MU_i.

3. After receiving the smart card, MU_i computes $r^* = h(ID_U||PB_U) \oplus r$, $V_U = h(ID_U||PB_U||r)$, and $R_U^* = R_U \oplus r$, and finally stores $< r^*, V_U, R_U^*, DID_U, h(\cdot), ID_H >$ in the smart card.

5.2 Mutual Authentication Phase

In this step, MU_i performs mutual authentication to establish a session key with FA_j with the help of HA_k. The detailed process, as depicted in Fig. 3, is as follows.

1. MU_i inputs ID_U, PW_U and BIO_U, computes $PB_U = h(PW_U||H(BIO_U))$ and $r = h(ID_U||PW_U) \oplus r^*$, and checks $V_U = h(ID_U||PB_U||r)$. If they are not equal, MU_i terminates this process; otherwise, MU_i selects $a_U \in Z_q^*$, computes $FID_U = h(ID_U||r)$, $R_U = R_U^* \oplus r$, $Q_1 = R_U \oplus a_U$, and $Q_2 = h(DID_U||R_U||a_U||FID_U||ID_F||ID_H)$, and sends the message $M_1 =< ID_F, ID_H, DID_U, Q_1, Q_2 >$ to FA_j.
2. After receiving the message M_1, FA_j selects a random $b_F \in Z_q^*$, computes $W_1 = h(M_1||K_{FH}||ID_F||ID_H) \oplus b_F$ and $W_2 = h(W_1||b_F)$, and sends the message $M_2 =< M_1, W_1, W_2 >$ to HA_k.
3. After receiving the message M_2, HA_k computes $b = h(M_1||K_{FH}||ID_F||ID_H) \oplus b_F$ and checks $W_2 = h(W_1||b_F)$. If they are not equal, HA_k terminates the process; otherwise, HA_k retrieves FID_U and n_H by calculating $\{FID_U, n_H\} = h(x) \oplus DID_U$, and then computes $R_U = h(FID_U||x)$, and $a_U = R_U \oplus Q_1$. HA_k then checks $Q_2 = h(DID_U||R_U||a_U||FID_U||ID_F||ID_H)$. If they are not equal, HA_k terminates the process; otherwise, HA_k selects a random $n_H^{new} \in Z_q^*$, computes $DID_U^{new} = h(x) \oplus \{FID_U||n_H^{new}\}$, $E_1 = h(DID_U^{new}||FID_U)$, $E_2 = h(R_U) \oplus DID_U^{new}$, $E_3 = h(K_{FH}||b_F) \oplus E_1$, $E_4 = a_U \oplus b_F$, and $E_5 = h(K_{FH}||b_F||E_1)$, and sends $M_3 =< E_2, E_3, E_4, E_5 >$ to FA_j.
4. FA_j computes $a_U = E_4 \oplus b_F$ and $E_1 = h(K_{FH}||b_F) \oplus E_3$, and checks $E_5 = h(K_{FH}||b_F||E_1)$. If they are not equal, FA_j terminates the process; otherwise, FA_j computes $SK_F = h(E_1||a_U||b_F)$, $W_3 = b_F \oplus h(E_1||a_U)$, and $W_4 = h(SK_F||M_1||E_2)$. Then it sends the message $M_2 =< E_2, W_3, W_4 >$ to MU_i.
5. MU_i computes $DID_U^{new} = E_2 \oplus h(R_U)$, $b_F = W_3 \oplus h(h(DID_U^{new}||FID_U)||a_U)$ and $SK_U = h(h(DID_U^{new}|| FID_U)||a_U||b_F)$, and checks $W_4 = h(SK_U||M_1||E_2)$. If they are equal, MU_i and FA_j successfully establish the session key SK.

5.3 Password Change Phase

In this phase, MU_i locally changes the password using the following procedure:

1. MU_i inputs ID_U, PW_U and BIO_U, computes $PB_U = h(PW_U||H(BIO_U))$ and $r = h(ID_U||PB_U) \oplus r^*$, and checks $V_U = h(ID_U||PB_U||r)$. If they are not equal, MU_i terminates the process; otherwise, MU_i selects a new password PW_U^{new}.
2. MU_i computes $r^{new*} = h(ID_U||PB_U^{new}) \oplus r$ and $V_U^{new} = h(ID_U||PB_U^{new}||r)$, and replaces r^* and V_U with r^{new*} and V_U^{new}.

5.4 Session Key Update Phase

This phase is performed for the session key update when MU_i remains in FA_j for a long time, as shown in Fig. 4.

1. MU_i selects a random $a_U \in Z_q^*$ and computes $H_1 = a_U \oplus h(SK_U^{i-1})$, $H_2 = h(H_1||SK_U^{i-1}||a_U)$ where SK^{i-1} is the $i-1$th session key. Then, MU_i sends $M_1 =< H_1, H_2 >$ to FA_j.
2. FA_j computes $a_U = H_1 \oplus h(SK_U^{i-1})$ and checks $H_2 = h(H_1||SK_F^{i-1}||a_U)$. If they are same, FA_j selects a random $b_F \in Z_q^*$, computes $J_1 = b_F \oplus a_U$, $SK_F^i = h(SK_U^{i-1}||a_U||b_F)$, and $J_2 = h(SK_F^i||SK_F^{i-1})$, and sends $M_2 =< J_1, J_2 >$ to MU_i.
3. After receiving the message, MU_i computes $b_F = J_1 \oplus a_U$ and $SK_U^i = h(SK_U^{i-1}||a_U||b_F)$. MU_i finally checks $J_2 = h(SK_U^i||SK_U^{i-1})$. If they are equal, then MU_i and FA_j successfully update the session key.

6. Security Analysis of the Proposed Scheme

6.1 Informal Security Analysis

In this section, we conduct an informal security analysis to demonstrate that SLARS is secure against various known attacks.

6.1.1 User Anonymity

In the proposed scheme, the real ID_U of MU_i is included

Mobile User (MU_i)	**Foreign agent** (FA_j)	**Home Agent** (HA_k)
Input ID_U, PW_U, BIO_U $PB_U = h(PW_U \Vert H(BIO_U))$ $r = h(ID_U \Vert PB_U) \oplus r^*$ Check $V_U = h(ID_U \Vert PB_U \Vert r)$ Select $a_U \in Z_q^*$ $FID_U = h(ID_U \Vert r)$ $R_U = R_U^* \oplus r = h(FID_U \Vert x)$ $Q_1 = R_U \oplus a_U$ $Q_2 = h(DID_U \Vert R_U \Vert a_U \Vert FID_U \Vert ID_F \Vert ID_H)$		
$\xrightarrow{M_1 =< ID_F, ID_H, DID_U, Q_1, Q_2 >}$	Select random $b_F \in Z_q^*$ $W_1 = h(M_1 \Vert K_{FH} \Vert ID_F \Vert ID_H) \oplus b_F$ $W_2 = h(W_1 \Vert b_F)$	
	$\xrightarrow{M_2 =< M_1, W_1, W_2 >}$	
		$b_F = h(M_1 \Vert K_{FH} \Vert ID_F \Vert ID_H) \oplus W_1$ Check $W_2 = h(W_1 \Vert b_F)$ $\{FID_U, n_H\} = h(x) \oplus DID_U$ $R_U = h(FID_U \Vert x)$ $a_U = R_U \oplus Q_1$ Check $Q_2 = h(DID_U \Vert R_U \Vert a_U \Vert FID_U \Vert ID_F \Vert ID_H)$ Select random $n_H^{new} \in Z_q^*$ $DID_U^{new} = h(x) \oplus (FID_U \Vert n_H^{new})$ $E_1 = h(DID_U^{new} \Vert FID_U)$ $E_2 = h(R_U) \oplus DID_U^{new}$ $E_3 = h(K_{FH} \Vert b_F) \oplus E_1$ $E_4 = a_U \oplus b_F$ $E_5 = h(K_{FH} \Vert b_F \Vert E_1)$
		$\xleftarrow{M_3 =< E_2, E_3, E_4, E_5 >}$
	$a_U = E_4 \oplus b_F$ $E_1 = h(K_{FH} \Vert b_F) \oplus E_3$ Check $E_5 = h(K_{FH} \Vert b_F \Vert E_1)$ $SK_F = h(E_1 \Vert a_U \Vert b_F)$ $W_3 = b_F \oplus h(E_1 \Vert a_U)$ $W_4 = h(SK_F \Vert M_1 \Vert E_2)$	
	$\xleftarrow{M_4 =< E_2, W_3, W_4 >}$	
$DID_U^{new} = E_2 \oplus h(R_U)$ $b_F = W_3 \oplus h(h(DID_U^{new} \Vert FID_U) \Vert a_U)$ $SK_U = h(h(DID_U^{new} \Vert FID_U) \Vert a_U \Vert b_F)$ Check $W_4 = h(SK_U \Vert M_1 \Vert E_2)$		

Fig. 3 Mutual authentication phase of SLARS.

in DID_U and transmitted through a public channel. ID_U is protected by a hashed secret key x, as well as random numbers r and n_H. It is difficult for an attacker to find these three values directly, which makes it impossible to determine the ID_U. Moreover, $DID_U^{new} = h(x) \oplus \{FID_U \Vert n_H^{new}\}$ is updated for each session. Therefore, the proposed scheme guarantees user anonymity.

6.1.2 Untraceability

In the proposed scheme, all values included in $M_1 - M_4$ transmitted through the public channel contain random numbers. In addition, these random numbers are changed every session and are never reused. Therefore, since the attacker cannot track the activities of the communication participants, the proposed scheme satisfies untraceability.

6.1.3 Resistance to Stolen-Mobile Device Attack

An adversary can potentially steal MU_i's mobile device and access $< r^*, V_U, R_U^*, DID_U, h(\cdot), ID_H >$ through a side-channel attack. However, the disclosure of ID_U and PW_U

Table 2 Comparison of security requirements.

Security property	Karuppia and Saravana [16]	Li et al. [17]	Xie et al. [18]	Chen et al. [19]	**Proposed**
User anonymity	✓	✓	✓	✓	✓
Untraceability	✓	✓	✓	✓	✓
Resistance to stolen-mobile device attack	✓	✗	✓	✓	✓
Resistance to replay attack	✓	✓	✓	✓	✓
Mutual authentication	✓	✓	✓	✓	✓
Forward secrecy	✗	✓	✓	✓	✓
Session key agreement and verification	✗	✓	✗	✓	✓
Resistance to user impersonation attack	✓	✗	✓	✓	✓
Resistance to FA impersonation attack	✓	✓	✓	✗	✓
Local user verification	✓	✓	✗	✓	✓
Resistance to stolen-verifier attack	✓	✓	✓	✓	✓
Resistance to known session-specific temporary in- formation attack	✓	✓	✓	✗	✓

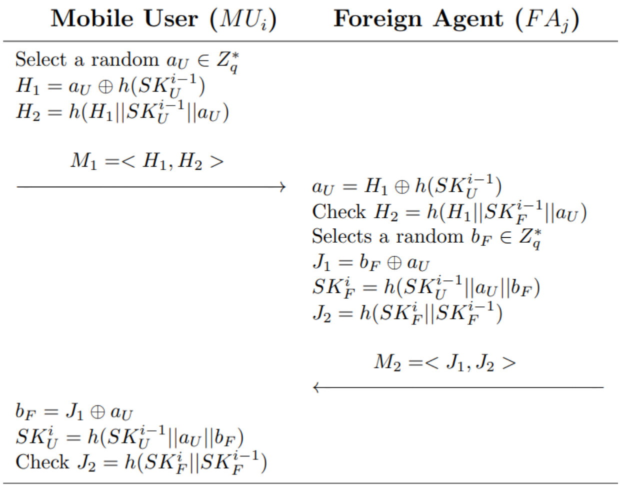


Fig. 4 Session key update phase of SLARS.

relies on XOR or biohash functions that are fortified by the values of r and BIO_U, rendering it practically unfeasible for the adversary to ascertain this information. Moreover, the generation of an authentication request message, essential for impersonating MU_i, necessitates access to FID_U, which remains shielded by the secret key x of HA_k. Consequently, even if an attacker acquires MU_i's mobile device, he/she cannot obtain sensitive information or disguise it as MU_i. Therefore, the proposed scheme is secure against a stolen-mobile device attack.

6.1.4 Resistance to Replay Attack

To maliciously generate a session key and mimic either MU_i or FA_j by replaying messages M_1 and M_4, an adversary requires a random number a_U or b_F. The acquisition of a_U necessitates access to R_U and FID_U, while obtaining b_F requires knowledge of K_{FH}. However, the adversary is unable to deduce these values from publicly transmitted messages. Consequently, the proposed scheme effectively mitigates the risks associated with potential replay attacks.

6.1.5 Mutual Authentication

In the proposed scheme, MU_i and FA_j establish a session key using HA_k. After HA_k receives DID_U from message M_1, HA_k retrieves FID_U and n_H by calculating $\{FID_U, n_H\} = h(x) \oplus DID_U$ using its own secret key and then verifies the identity of MU_i by checking $Q_2 = h(DID_U||R_U||a_U||FID_U||ID_F||ID_H)$. FA_j is authenticated to HA_k using W_1, which is created using K_{FH}. After HA_k successfully authenticates FA_j and MU_i, HA_k can obtain the random numbers a_U and b_F. Then, HA_k transmits E_4 to FA_j, and FA_j obtains the random $a_U = E_4 \oplus b_F$ and $E_1 = h(K_{FH}||b_F) \oplus E_3 = h(DID_U^{new}||FID_U)$ that only MU_i can calculate. MU_i obtains b_F from $W_3 \oplus h(h(DID_U^{new}||FID_U)||a_U)$. Finally, MU_i and FA_j compute the session key $SK = h(h(DID_U^{new}||FID_U)||a_U||b_F)$ using the parameters obtained through mutual authentication. Thus, the proposed scheme provides mutual authentication.

6.1.6 Forward Secrecy

In the proposed scheme, the random numbers a_U and b_F included in the session key are not reused for every session. In other words, they are updated with new values. Therefore, an attacker is unable to infer any relevance from the past, present, and future session keys. Therefore, the proposed scheme satisfies the forward secrecy requirement.

6.1.7 Session Key Agreement and Verification

In the proposed scheme, MU_i and FA_j generate session key $SK = h(h(DID_U^{new}||FID_U)||a_U||b_F)$ through mutual authentication. The FA_j transmits $W_4 = h(SK_F||M_1||E_2)$ to MU_i. Then, MU_i checks whether W_4 is legitimate by using the generated session key. Through this process, MU_i verifies that the same session key has been exchanged between them. Therefore, the proposed scheme satisfies the session key agreement.

6.1.8 Resistance to User Impersonation Attack

In the proposed scheme, an attacker can attempt to impersonate a MU_i. Nevertheless, our scheme ensures resistance against stolen-mobile device attacks. Furthermore, the proposed scheme guarantees mutual authentication and verifies the validity of the session key. Therefore, the proposed scheme is secure against a user impersonation attack.

6.1.9 Resistance to FA Impersonation Attack

In the proposed scheme, an attacker can create a random b_A and attempt to impersonate FA_j. However, without K_{FH} shared by FA_j and HA_k, a valid message cannot be generated and the random number a_U of MU_i cannot be determined. Therefore, the proposed scheme is safe from FA impersonation attacks.

6.1.10 Local User Verification

In the proposed scheme, the identity of MU_i is verified before a mutual authentication request message is generated. MU_i can generate the same value as V_U stored in the smart card by correctly entering ID_U, PW_U, and BIO_U. Therefore, the proposed scheme can block malicious access locally.

6.1.11 Resistance to Stolen-Verifier Attack

In the proposed scheme, HA_k does not maintain a verifier table because it does not store any sensitive information for authenticating MU_i and FA_j. Therefore, an attacker cannot perform a stolen-verifier attack.

6.1.12 Resistance to Known Session-Specific Temporary Information Attack

If an attacker knows the random a_U and b_F constituting the session key, he/she can try to determine the session key. However, to calculate the session key, E_1 together with two random numbers is required. Because an attacker cannot determine FID_U, the proposed scheme is secure from known session-specific temporary information attacks.

6.2 Formal Analysis Using ProVerif

ProVerif is an automated verification tool that formally performs security analysis of cryptographic protocols based on various cryptographic primitives, such as RSA, ECC, bilinear pairing, and hash functions [28]. ProVerif verifies whether confidentiality and authentication properties are satisfied through proofs of conformance, reachability, and equivalence of protocols. In various network environments, ProVerif is used to check the security properties of authentication protocols. In this subsection, we present the entire ProVerif code for performing a formal analysis of SLARS and its results.

In Table 3, we define the public and secure channels between the participants, predefined constants, secret keys, session keys, cryptographic functions, and communication events for each node.

We define the simulation codes for MU_i, FA_j, and HA_k in the registration phase and the mutual authentication phases in Tables 4, 5, and 6, respectively.

In Table 7, we define the simulation code for attacker's capabilities to reveal session keys SK_U and SK_F. Then, it verifies that the internodal relationships of the proposed scheme follow the accurate procedure.

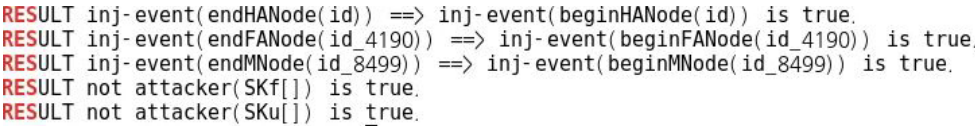
Table 3 ProVerif for definitions.

```
(*......chanels......*)
free cha:channel [private].
free chb:channel.
free chc:channel.

(*......constants......*)
free IDu:bitstring [private].
free IDf:bitstring.
free IDh:bitstring.
free PWu:bitstring [private].
free BIOu:bitstring [private].

(*......secret key......*)
free x:bitstring [private].
free Kfh:bitstring [private].

(*......shared key......*)
free SKf:bitstring [private].
free SKu:bitstring [private].

(*......functions......*)
fun cont(bitstring,bitstring) : bitstring.
fun XOR(bitstring,bitstring):bitstring.
fun h(bitstring):bitstring.
fun H(bitstring):bitstring.
equation forall p:bitstring, q:bitstring; XOR(XOR(p,q),q)=p.

(*......events......*)
event beginHANode(bitstring).
event endHANode(bitstring).
event beginFANode(bitstring).
event endFANode(bitstring).
event beginMUode(bitstring).
event endMUode(bitstring).
```

Table 4 ProVerif code to simulate MU.

```
(*......MU......*)
let pMUode=
new r:bitstring;
let PBu = h(cont(PWu,H(BIOu))) in
let FIDu = h(cont(IDu,r)) in
out(cha,(IDu,FIDu));
in(cha,(XDIDu:bitstring,XRu:bitstring,XIDh:bitstring));
let r’=XOR(r, h(cont(IDu,PBu))) in
let Vu=h(cont(IDu,cont(PBu,r))) in
let Ru’=XOR(XRu,r) in
event beginMUode(IDu);
new a:bitstring;
let PBu’=h(cont(PWu,H(BIOu))) in
let r”=XOR(r’, h(cont(IDu,PBu))) in
let Vu’=h(cont(IDu,cont(PBu,r”))) in
if Vu=Vu’ then
let XRu’=XOR(Ru’,r) in
let Q1=XOR(XRu’,a) in
let Q2=h(cont(XDIDu,cont(XRu’,cont(a,cont(FIDu,cont(IDf,IDh)))))) in
let M1=cont(IDf,cont(IDh,cont(XDIDu,cont(Q1,Q2)))) in
out(chc,(M1));
in(chc,XM4:bitstring);
let (XXE2:bitstring,XXW3:bitstring,XXW4:bitstring)= XM4 in
let DIDunew’=XOR(XXE2,h(XRu’)) in
let b”= XOR(XXW3,h(cont(a,h(cont(DIDunew’,FIDu)))))in
let SKu=h(cont(h(cont(DIDunew’,FIDu)),cont(a,b”))) in
let W4’=h(cont(SKu,cont(M1,XXE2))) in
if(XXW4 = W4’) then event endMUode(IDu).
```

Figure 5 shows the simulation results when all authentication parameters, queries, and events are accurate, each participant achieves mutual authentication, and the session key is securely generated between MU_i and FA_j.



```
RESULT inj-event(endHANode(id)) ==> inj-event(beginHANode(id)) is true.
RESULT inj-event(endFANode(id_4190)) ==> inj-event(beginFANode(id_4190)) is true.
RESULT inj-event(endMNode(id_8499)) ==> inj-event(beginMNode(id_8499)) is true.
RESULT not attacker(SKf[]) is true.
RESULT not attacker(SKu[]) is true.
```

Fig. 5 Simulation results.

Table 5 ProVerif code to simulate FA.

```
(*......FA......*)
let pFAgent=
event beginFANode(IDf);
in(chc, (XM1:bitstring));
let (XXM1:bitstring) = XM1 in
new b:bitstring;
let W1=h(cont(XM1,cont(Kfh,cont(IDf,IDh)))) in
let W2=h(cont(W1,b)) in
out(chb,(XXM1,W1,W2));
in(chb,(XM3:bitstring));
let (XE2:bitstring,XE3:bitstring,XE4:bitstring,XE5:bitstring)=XM3 in
let a'=XOR(XE4,b) in
let E1'=XOR(XE3,b) in
let E5'=h(cont(Kfh,cont(b,E1'))) in
if E5'=XE5 then
let SKf=h(cont(E1',cont(a',b))) in
let W3=XOR(b,h(cont(E1',a'))) in
let W4=h(cont(SKf,cont(XM1,XE2))) in
let M4 = cont(XE2,cont(W3,W4)) in
out(chc,(M4));
event endFANode(IDf).
```

Table 6 ProVerif code to simulate HA.

```
(*......HA......*)
let pHAgent=
in(cha,(XIDu:bitstring, XFIDu:bitstring));
new nH:bitstring;
let Ru=h(cont(XFIDu,x)) in
let DIDu=XOR(h(x),cont(XFIDu,nH)) in
out(cha,(DIDu,Ru,IDh));
event beginHANode(IDh);
in(cha,(XM3:bitstring, XW1:bitstring, XW2:bitstring));
let (XIDf:bitstring,XIDh:bitstring,XXDIDu:bitstring,XQ1:bitstring,
XQ2:bitstring)=XM3 in
let Xb=XOR(XW1,h(cont(XQ1,cont(Kfh,cont(XIDf,XIDh))))) in
if XW2=h(cont(XW1,Xb)) then
let (XXFIDu:bitstring,XnH:bitstring)=XOR(h(x),XXDIDu) in
let XXRu=h(cont(XXFIDu,x)) in
let Xa=XOR(XXRu,XQ1) in
let XQ2'=h(cont(XXDIDu,cont(XXRu,cont(Xa,cont(XXFIDu,
cont(XIDf,XIDh)))))) in
if XQ2=XQ2' then
new nHnew:bitstring;
let DIDunew=XOR(h(x),cont(XXFIDu,nHnew)) in
let E1=h(cont(DIDunew,XXFIDu)) in
let E2=XOR(h(XXRu),DIDunew) in
let E3=XOR(Xb,E1) in
let E4=XOR(Xa,Xb) in
let E5=h(cont(Kfh,cont(Xb,E1))) in
let M3=cont(E2,cont(E3,cont(E4,E5))) in
out(chb, (M3));
event endHANode(IDh).
```

Table 7 ProVerif code to simulate attacker's capabilities.

```
(*......quries......*)
query attacker(SKu).
query attacker(SKf).
query id:bitstring; inj-event(endMUode(id)) ==¿ inj-event(beginMUode(id)).
query id:bitstring; inj-event(endFANode(id)) ==¿ inj-event(beginFANode(id)).
query id:bitstring; inj-event(endHANode(id)) ==¿ inj-event(beginHANode(id)).
set traceDisplay=long.
process
((!pMUode)—(!pFAgent)—(!pHAgent))
```

6.3 Authentication Proof Using BAN Logic

BAN logic [29] is a well-known formal logic for analyzing the security of cryptographic protocols. This subsection validates the legitimacy of session keys distributed to MU_i and FA_j. The basic notation for BAN logic is as follows:

- $P \triangleleft C$: Participant P sees condition C.
- $P \mid\equiv C$: C is believed by P
- $\sharp(C)$: It makes a fresh C.
- $P \mid\sim C$: P expresses C.
- $P_1 \overset{K}{\longleftrightarrow} P_2$: Two participants P_1 and P_2 share a secret key K.
- $P \Rightarrow C$: C is handled by P.
- $(C)_K$: Perform the cryptographic operation on C using K.

BAN logic also offers the following five logic rules:

- Rule 1 (Message-meaning rule): $\frac{P_1 \mid\equiv P_1 \overset{K}{\leftrightarrow} P_2, P_1 \triangleleft <C>_K}{P_1 \mid\equiv P_2 \mid\sim C}$, and if P_1 trusts that the key K is shared with P_2, P_1 sees C combined with K, and then P_1 trusts P_2 once said C.
- Rule 2 (Nonce-verification rule): $\frac{P_1 \mid\equiv \#(C), P_1 \mid\equiv P_2 \mid\sim C}{P_1 \mid\equiv P_2 \mid\equiv C}$: if P_1 trusts that C's freshness and P_1 trusts P_2 once said C, then P_1 trusts that P_2 trusts C.
- Rule 3 (Believe rule): $\frac{P \mid\equiv C, P \mid\equiv M}{P \mid\equiv (C,M)}$: if P trusts C and M, (C,M) are also trusted by P.
- Rule 4 (Freshness-conjuncatenation rule): $\frac{P \mid\equiv \#(C)}{P \mid\equiv \#(C,M)}$: if the freshness of C is trusted by P, then P can trust the freshness of the full condition.
- Rule 5 (Jurisdiction rule): $\frac{P_1 \mid\equiv P_2 \mid\Rightarrow C, P_1 \mid\equiv P_2 \mid\equiv C}{P_1 \mid\equiv C}$: if P_1 trusts that P_2 has jurisdiction over C, and P_1 trusts that P_2 trusts condition C, then P_1 also trusts C.

Through our analysis, the following four goals can be satisfied:

- Goal 1: $MU \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$
- Goal 2: $FA \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$
- Goal 3: $MU \mid\equiv FA \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$
- Goal 4: $FA \mid\equiv MU \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$

Next, all transmitted messages can be transmuted into an idealized form as follows:

- Using $M_1 =< ID_F, ID_H, DID_U, Q_1, Q_2 >$, $MU \rightarrow FA$: $DID_U = h(x) \oplus \{FID_U || n_H\}$, $Q_1 = R_U \oplus a_U$, $Q_2 = h(DID_U || R_U || a_U || FID_U || ID_F || ID_H)$. This is reduced to $MSG_1 : (ID_F, ID_H, FID_U, R_U, a_U)_x$.
- Using $M_2 =< M_1, W_1, W_2 >$, $FA \rightarrow HA$: $W_1 = h(M_1 || K_{FH} || ID_F || ID_H) \oplus b_F$, $W_2 = h(W_1 || b_F)$. This is reduced to $MSG_2 : (M_1, ID_F, ID_H, b_F)_{K_{FH}}$.

- Using $M_3 =< E_2, E_3, E_4, E_5 >$, $HA \rightarrow FA$: $E_1 = h(DID_U^{new}||FID_U)$, $E_2 = h(R_U) \oplus DID_U^{new}$, $E_3 = b_F \oplus E_1$, $E_4 = a_U \oplus b_F$, $E_5 = h(K_{FH}||b_F||E_1)$. This is reduced to $MSG_3 : (FID_U, R_U, a_U, b_F)_{K_{FH}}$.
- Using $M_4 =< E_2, W_3, W_4 >$, $FA \rightarrow MU$: $W_3 = b_F \oplus h(E_1||a_U)$ and $W_4 = h(SK_F||M_1||E_2)$. This is reduced to $MSG_4 : (FID_U, R_U, a_U, b_F)_x$.

- A1: $MU \mid\equiv \sharp(a_U)$
- A2: $FA \mid\equiv \sharp(b_F)$
- A3: $FA \mid\equiv (FA \overset{K_{FH}}{\longleftrightarrow} HA)$
- A4: $HA \mid\equiv (FA \overset{K_{FH}}{\longleftrightarrow} HA)$
- A5: $MU \mid\equiv (MU \overset{x}{\longleftrightarrow} HA)$
- A6: $HA \mid\equiv (MU \overset{x}{\longleftrightarrow} HA)$
- A7: $FA \mid\equiv (MU \overset{a_U}{\longleftrightarrow} FA)$
- A8: $MU \mid\equiv (MU \overset{b_F}{\longleftrightarrow} FA)$
- A9: $MU \mid\equiv FA \Rightarrow (MU \overset{SK}{\longleftrightarrow} FA)$
- A10: $FA \mid\equiv MU \Rightarrow (MU \overset{SK}{\longleftrightarrow} FA)$

We then describe our main proof using predefined information, including five logic rules, four messages, and ten assumptions.

- From M_1, we obtain V1: $FA \lhd (ID_F, ID_H, FID_U, R_U)_{a_U}$
- Based on Assumption A5 and Rule 1, we derive V2: $FA \mid\equiv MU \mid\sim (ID_F, ID_H, FID_U, R_U)_{a_U}$
- Based on Assumption A1 and Rule 4, we derive V3 as follows: $FA \mid\equiv \sharp(ID_F, ID_H, FID_U, R_U)_{a_U}$
- Based on V1, V2 and Rule 2, we derive V4: $FA \mid\equiv MU \mid\equiv (ID_F, ID_H, FID_U, R_U)_{a_U}$
- From M_2, we obtain V5: $HA \lhd (M_1, ID_F, ID_H, a_U, b_F)_{K_{FH}}$
- Based on Assumption A5 and Rule 1, we derive V6: $HA \mid\equiv FA \mid\sim (M_1, ID_F, ID_H, a_U, b_F)_{K_{FH}}$
- Based on Assumption A1 and Rule 4, we derive V7 as follows: $HA \mid\equiv \sharp(M_1, ID_F, ID_H, a_U, b_F)_{K_{FH}}$
- Based on V1, V2 and Rule 2, we derive V8: $HA \mid\equiv FA \mid\equiv (M_1, ID_F, ID_H, a_U, b_F)_{K_{FH}}$
- From M_3, we obtain V9: $FA \lhd (DID_U^{new}, FID_U, a_U, b_F)_{K_{FH}}$
- Based on Assumption A5 and Rule 1, we derive V10: $FA \mid\equiv HA \mid\sim (DID_U^{new}, FID_U, a_U, b_F)_{K_{FH}}$
- Based on Assumption A1 and Rule 4, we derive V11 as follows: $FA \mid\equiv \sharp(DID_U^{new}, FID_U, a_U, b_F)_{K_{FH}}$
- Based on V1, V2 and Rule 2, we derive V12: $FA \mid\equiv HA \mid\equiv (DID_U^{new}, FID_U, a_U, b)_{K_{FH}}$
- From M_4, we obtain V13: $MU \lhd (DID_U^{new}, FID_U)_{b_F}$
- Based on Assumption A5 and Rule 1, we derive V14: $MU \mid\equiv FA \mid\sim (DID_U^{new}, FID_U)_{b_F}$
- Based on Assumption A1 and Rule 4, we derive V15 as follows: $MU \mid\equiv \sharp(DID_U^{new}, FID_U)_{b_F}$
- Based on V1, V2 and Rule 2, we derive V16: $MU \mid\equiv FA \mid\equiv (DID_U^{new}, FID_U)_{b_F}$
- Based on V8, $SK_F = h(E_1||a_U||b_F)$, and $E_1 = h(DID_U^{new}||FID_U)$, we derive V17 as follows: $MU \mid\equiv (MU \overset{SK_F}{\longleftrightarrow} FA)$ (Goal 1)
- Based on V6 and $SK_U = h(h(DID_U^{new}||FID_U)||a_U||b_F)$, we derive V18: $FA \mid\equiv (MU \overset{SK_U}{\longleftrightarrow} FA)$ (Goal 2)

Table 8 Execution time of cryptographic operation (ms).

Operation	**Description**	FA_j **and** HA_k	MU_i
T_m	Scalar multiplication	9.661	15.44
T_e	Modular exponentiation	1.434	1.927
T_s	Symmetric encryption	0.355	0.396
T_h	Hash function	0.082	1.147

- Based on A9, V8 and Rule 5, we derive V19 as follows: $MU \mid\equiv FA \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$ (Goal 3)
- Based on A10, V9 and Rule 5, we derive V20 as follows: $FA \mid\equiv MU \mid\equiv (MU \overset{SK}{\longleftrightarrow} FA)$ (Goal 4)

Thus, MU_i and FA_j achieve mutual authentication and the session key SK is securely shared between them by achieving Goals 1, 2, 3, and 4.

7. Performance Analysis

In this section, we conduct a performance analysis of SLARS in terms of computational and communication costs and compare the results with those from related studies based on the same communication mechanism. We consider 320-bit elliptic multiplication, 1024-bit-modular exponentiation with a large integer n, 128-bit advanced encryption standard (AES), and 256-bit hash function.

We measured the computation time for each cryptographic primitive by assuming the computing power of each participant as follows; the results are summarized in Table 8.

1. MU_i: Galaxy Note 20 Device, AP; Octa-Core Processor ※3.09GHz + 2.42GHz*3 + 1.8Ghz*4, 8G memory, OS; Android 11, and Android Studio and Software Development Kits tools using the Java Pairing-Based Cryptography Library (JPBC) Library 2.0.0.
2. FA_j and HA_k : CPU; Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz Quad-Core, 16G memory, OS; Win10 64 bit, and Eclipse IDE using the JPBC Library 2.0.0.

Table 9 and Fig. 6 present the performance comparison results for the schemes of Karuppia and Saravana [16], Li et al. [17], Xie et al. [18], Chen et al. [19], and our SLARS in terms of the computational cost of MU_i, FA_j and HA_k in the mutual authentication phase. As listed in the table, the computational costs of the schemes proposed by Karuppia and Saravana [16], Li et al. [17], Xie et al. [18], Chen et al. [19], and SLARS were $22T_h + 3T_e + 1T_s \approx 17.06ms$, $17T_h + 6T_m \approx 85.22ms$, $20T_h + 7T_m \approx 96.19ms$, $19T_h + 4T_m \approx 60.28ms$, and $32T_h \approx 16.47ms$, respectively. This shows that SLARS has the lowest computational cost compared to the related schemes.

To compare the communication costs of the mutual authentication phases, we assumed that the lengths of identity, timestamp, and random number are 128, 32, and 64 bits, respectively. As shown in Table 10 and Fig. 7, the total communication costs of the schemes of Karuppia and Saravana [16], Li et al. [17], Xie et al. [18], Chen et al. [19], and SLARS were 4768, 4208, 4256, 5364 and 4416 bits, respectively.

As we can see, the communication cost of SLARS is

Table 9 Comparison of the computational cost.

Scheme	Karuppia and Saravana [16]	Li et al. [17]	Xie et al. [18]	Chen et al. [19]	**Proposed**
MU_i	$10T_h + 2T_e$	$8T_h + 3T_m$	$9T_h + 3T_m$	$8T_h + 2T_m$	$13T_h$
FA_j	$3T_h$	$4T_h + 2T_m$	$4T_h + 2T_m$	$4T_h + 2T_m$	$8T_h$
HA_k	$9T_h + 3T_e + 1T_s$	$5T_h + 1T_m$	$7T_h + 2T_m$	$7T_h$	$11T_h$
Total	$22T_h + 3T_e + 1T_s$	$17T_h + 6T_m$	$20T_h + 7T_m$	$19T_h + 4T_m$	$32T_h$
Time	$\approx 17.06ms$	$\approx 85.22ms$	$\approx 96.19ms$	$\approx 60.28ms$	$\approx 16.47ms$

Table 10 Comparison of computational cost (bits).

Scheme	Karuppia and Saravana [16]	Li et al. [17]	Xie et al. [18]	Chen et al. [19]	**Proposed**
MU_i	1696	1104	1128	1192	1088
FA_j	2272	2592	2616	3000	2432
HA_k	800	512	512	1192	896
Total	4768	4208	4256	5384	4416

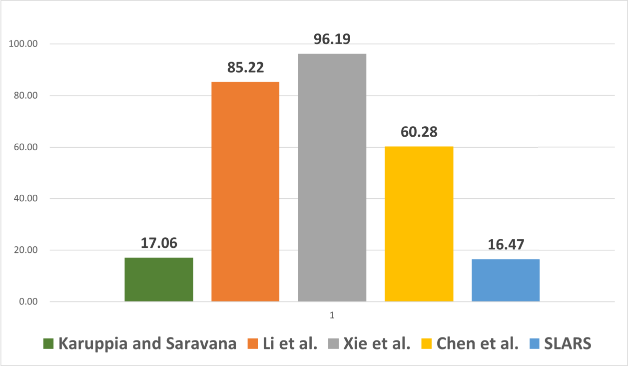


Fig. 6 Comparison of the computational cost.

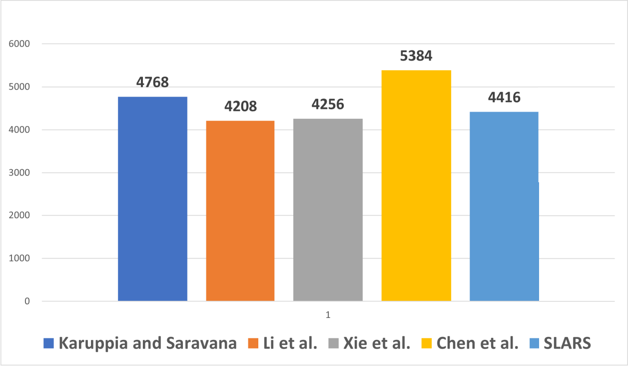


Fig. 7 Comparison of communication cost (bits).

slightly higher than that of Li et al. and Xie et al., whereas the computation cost is the lowest. This is an improvement of approximately 73% when compared with the scheme of Chen et al, the most recent study.

8. Conclusions

In this study, we demonstrated weaknesses in the recently proposed user authentication schemes for enhancing security in GLOMONET. We reported a vulnerability in Chen et al.'s scheme: an adversary can pretend to be an FA by information attack and obtain the session key for a known session-specific temporary information attack. To address vulnerabilities, we proposed SLARS. Further, we performed informal and formal analyses using BAN logic and Proverif to verify the safety of SLARS against various known attacks. In addition, our proposed scheme, which was designed using only XOR and hash functions, improved the efficiency and significantly reduced the computational cost when compared to those of other schemes using ECC and exponential operations as cryptographic primitives for authentication. Consequently, SLARS was proven to be more efficient and secure than other related schemes for roaming services in smart cities.

References

[1] G.F. Huseien and K.W. Shah, "A review on 5G technology for smart energy management and smart buildings in singapore," Energy and AI, vol.7, p.100116, 2022.

[2] S.E. Bibri, "Data-driven environmental solutions for smart sustainable cities: Strategies and pathways for energy efficiency and pollution reduction," Euro-Mediterr. J. Environ. Integr., vol.5, 66, 2020.

[3] D. Jiang, "The construction of smart city information system based on the internet of things and cloud computing," Computer Communications, vol.150, pp.158–166, 2020.

[4] B. Kim, M. Yoo, K.C. Park, K.R. Lee, and J.H. Kim, "A value of civic voices for smart city: A big data analysis of civic queries posed by seoul citizens," Cities, vol.108, p.102941, 2021.

[5] M.D. Lytras and A. Visvizi, "Who uses smart city services and what to make of it: Toward interdisciplinary smart cities research," Sustainability, vol.10, no.6, p.1998, 2018.

[6] S. Rani, A. Kataria, M. Chauhan, P. Rattan, R. Kumar, and A.K. Sivaraman, "Security and privacy challenges in the deployment of cyber-physical systems in smart city applications: State-of-art work," Materials Today: Proceedings, vol.62, pp.4671–4676, 2022.

[7] M. Ghahramani, R. Javidan, and M. Shojafar, "A secure biometric-based authentication protocol for global mobility networks in smart cities," J. Supercomput., vol.76, no.11, pp.8729–8755, 2020.

[8] X. Li, J. Niu, S. Kumari, F. Wu, and K.K.R. Choo, "A robust biometrics based three-factor authentication scheme for global mobility networks in smart city," Future Generation Computer Systems, vol.83, pp.607–618, 2018.

[9] F. Wu, X. Li, L. Xu, S. Kumari, and A.K. Sangaiah, "A novel mutual authentication scheme with formal proof for smart healthcare systems under global mobility networks notion," Computers & Electrical Engineering, vol.68, pp.107–118, 2018.

[10] J. Zhu and J. Ma, "A new authentication scheme with anonymity for wireless environments," IEEE Trans. Consum. Electron., vol.50, no.1, pp.231–235, 2004.

[11] C.C. Lee, M.S. Hwang, and I.E. Liao, "Security enhancement on a new authentication scheme with anonymity for wireless environments," IEEE Trans. Ind. Electron., vol.53, no.5, pp.1683–1687, 2006.

[12] C.C. Wu, W.B. Lee, and W.J. Tsaur, "A secure authentication scheme with anonymity for wireless communications," IEEE Commun. Lett., vol.12, no.10, pp.722–723, 2008.

[13] J. Xu and D. Feng, "Security flaws in authentication protocols with anonymity for wireless environments," ETRI J., vol.31, no.4, pp.460–462, 2009.

[14] J.S. Lee, J.H. Chang, and D.H. Lee, "Security flaw of authentication scheme with anonymity for wireless communications," IEEE Commun. Lett., vol.13, no.5, pp.292–293, 2009.

[15] M. Kang, H.S. Rhee, and J.Y. Choi, "Improved user authentication scheme with user anonymity for wireless communications," IEICE Trans. Fundamentals, vol.E94-A, no.2, pp.860–864, Feb. 2011.

[16] M. Karuppiah and R. Saravanan, "A secure authentication scheme with user anonymity for roaming service in global mobility networks," Wireless Pers. Commun., vol.84, no.3, pp.2055–2078, 2015.

[17] X. Li, A.K. Sangaiah, S. Kumari, F. Wu, J. Shen, and M.K. Khan, "An efficient authentication and key agreement scheme with user anonymity for roaming service in smart city," Pers. Ubiquit. Comput., vol.21, no.5, pp.791–805, 2017.

[18] Q. Xie and L. Hwang, "Security enhancement of an anonymous roaming authentication scheme with two-factor security in smart city," Neurocomputing, vol.347, pp.131–138, 2019.

[19] R. Chen, Y. Mou, and M. Zhang, "An improved anonymous dos-resistant authentication protocol in smart city," Wireless Netw., vol.28, pp.745–763, 2022.

[20] N. Koblitz, "Elliptic curve cryptosystems," Mathematics of computation, vol.48, no.177, pp.203–209, 1987.

[21] H. Lee, D. Kang, Y. Lee, and D. Won, "Secure three-factor anonymous user authentication scheme for cloud computing environment," Wireless Communications and Mobile Computing, vol.2021, pp.1–20, 2021.

[22] Y. Park, K. Park, and Y. Park, "Secure user authentication scheme with novel server mutual verification for multiserver environments," Int. J. Commun. Syst., vol.32, no.7, p.e3929, 2019.

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

[24] L. Zhang, Y. Zhang, S. Tang, and H. Luo, "Privacy protection for e-health systems by means of dynamic authentication and three-factor key agreement," IEEE Trans. Ind. Electron., vol.65, no.3, pp.2795–2805, 2017.

[25] P.K. Roy and A. Bhattacharya, "An anonymity-preserving mobile user authentication protocol for global roaming services," Computer Networks, vol.221, p.109532, 2023.

[26] S. Khatoon, T.Y. Chen, and C.C. Lee, "An improved user authentication and key agreement scheme for roaming service in ubiquitous network," Ann. Telecommun., vol.77, pp.621–640, 2022.

[27] M. Indushree, M. Raj, V.K. Mishra, R. Shashidhara, A.K. Das, and V. Bhat, "Mobile-chain: Secure blockchain based decentralized authentication system for global roaming in mobility networks," Computer Communications, vol.200, pp.1–16, 2023.

[28] B. Blanchet, B. Smyth, V. Cheval, and M. Sylvestre, "ProVerif 2.00: Automatic cryptographic protocol verifier, user manual and tutorial," Version from, pp.05–16, 2018.

[29] M. Burrows, M. Abadi, and R.M. Needham, "A logic of authentication," Proc. Royal Society of London. A. Mathematical and Physical Sciences, vol.426, no.1871, pp.233–271, 1989.

Hakjun Lee received the B.S. degree in Software Engineering from the Korea National University of Transportation, South Korea, in 2015. He received the M.S. and Ph.D. in Electrical and Computer Engineering at Sungkyunkwan University in 2018 and 2022, respectively. He is also currently an Assistant Professor in the Department of Computer Engineering at Kyungnam University, South Korea. His current research interests include Cryptography, Authentication Protocols, and Blockchain.