Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 4083199, 14 pages
https://doi.org/10.1155/2021/4083199

Research Article

Provably Secure Client-Server Key Management Scheme in 5G Networks

Lei Yang,[1] **Yeh-Cheng Chen**,[2] **and Tsu-Yang Wu**[1]

[1]*College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China*
[2]*Department of Computer Science, University of California, Davis, CA, USA*

Correspondence should be addressed to Tsu-Yang Wu; wutsuyang@gmail.com

Received 25 August 2021; Revised 5 October 2021; Accepted 7 October 2021; Published 22 October 2021

Academic Editor: Muhammad Asghar Khan

The increasing demand for real-time data transmission in wireless mobile communication networks has promoted the maturity of mobile communication technology. Fifth-generation (5G) mobile communication technology is combined with cloud computing, high-frequency signal transmission, and other technologies and perfectly fits with the client-server architecture. 5G has been applied in many fields, such as the interconnection of smart devices, virtual reality, and cloud-based life. To provide the security and availability of the required services, we proposed a key management scheme based on the multiserver architecture of the client-server mode in 5G networks, which uses bilinear pairings and elliptic curve cryptography. Through informal security analysis and formal analysis (under the random oracle model and ProVerif tool), we demonstrated that the proposed scheme can complete mutual authentication and resist common network attacks. Furthermore, after the performance analysis of our scheme and other related schemes, it was found that this scheme has relatively low communication and computation costs and better security performance.

1. Introduction

The growth of mobile data has promoted the development of fifth-generation (5G) and sixth-generation (6G) mobile networks [1–4]. The data flow in mobile communication networks is soaring, and early mobile networks cannot meet the needs of users. The 5G network integrates 4G, WiFi, and other networks, providing richer communication modes and a better user experience. Specifically, in the 2G network era, users can only read words; in the 3G network era, users can view pictures; in the 4G network era, users can watch videos; in the 5G network era, users can engage in virtual reality interaction, cloud storage, and smart device interconnection. A content delivery network (CDN), as one of the key technologies of 5G networks, adds an intelligent virtual network based on the traditional network, which can build multiple proxy servers between the users and the source server. This requires the use of the benefits of the multiserver architecture and cloud computing technology to distribute information to users. As an extension of 5G, 6G can connect terrestrial wireless and satellite communications to achieve global coverage and the interconnection of everything. In other words, 5G/6G networks have a high transmission rate, low power consumption, low time delay, and other properties, allowing them to accommodate a large number of Internet of Things (IoT) devices and mobile users.

The increase in the electromagnetic wave frequency in 5G/6G networks results in high transmission rates, but it also leads to a reduction in the coverage distance and the deployment of more base stations. Furthermore, the deployment of base stations is related to the scope of network management, in which network security management is the fundamental guarantee for users to use a good network. In addition, as an emerging mobile communication, 5G/6G networks will involve many fields, such as mobile phones, smart homes, automatic driving, and telemedicine. In applications involving IoT [5–9], security has always been a weak link in the network. The management of stored and transmitted information is important. Once the information is disclosed, tampered with, or forged, it will lead to serious consequences. In addition, the client-server-based multiserver architecture overcomes the shortage of resources and long response time

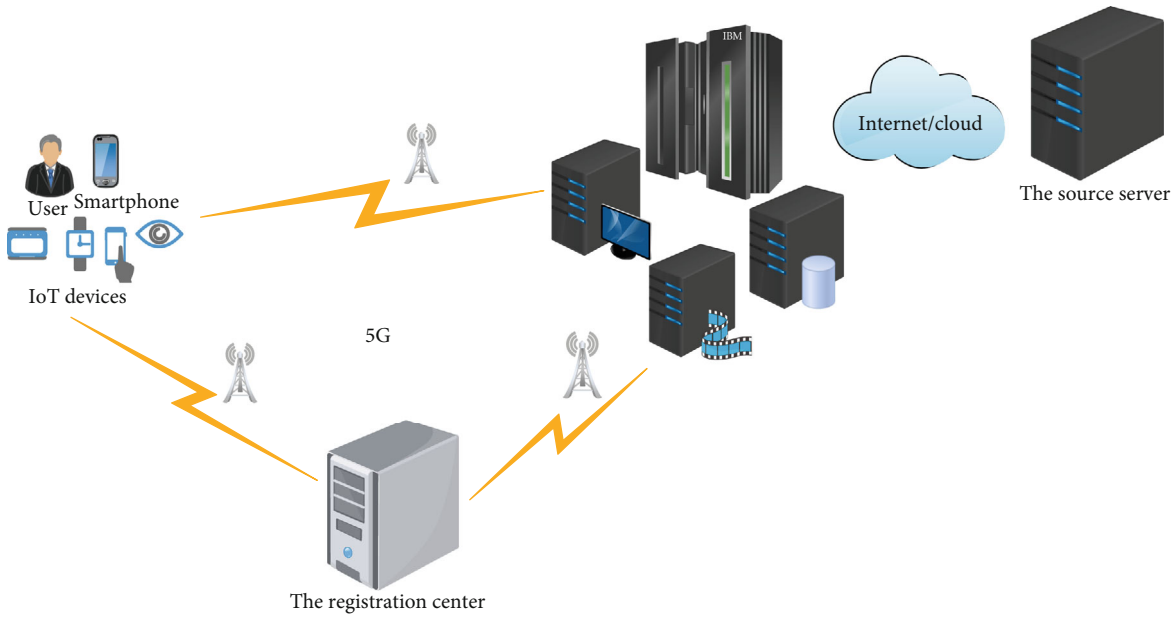

Figure 1: The client-server communication construction in 5G.

of a single server and can provide powerful network data processing capability. Therefore, to solve the security difficulties in network communication and improve user experience, secure authentication schemes for key management and storage based on the multiserver architecture in mobile networks have been proposed. The client-server communication construction in 5G is shown in Figure 1.

In Figure 1, every client and server must register with the registration center to obtain a legal identity to communicate in the network. Note that the clients include users and IoT devices. Owing to the powerful storage function of cloud computing technology, each proxy server will have the corresponding backup of the transmitted information. Therefore, in the authentication phase, a legitimate client directly authenticates and confirms the session key with a proxy server without the participation of the third-party source server.

In recent years, 5G has gradually developed into the core of mobile communication systems, and its distributed service mechanism is consistent with the multiserver architecture of the client-server mode. Therefore, some researchers have started working on mutual authentication protocols based on multiserver 5G networks. In 2019, Ying and Nayak [10] introduced an anonymous multiserver authentication scheme (MSAS) using self-certified public key cryptography in 5G and declared that their scheme was secure. Unfortunately, Haq et al. [11] found that [10] did not provide untraceability and two-factor security and could not resist offline identity, password guessing attacks, or user impersonation attacks. They then proposed an enhanced MSAS in 5G networks.

In this study, we introduced a key management scheme based on the multiserver architecture of the client-server mode in 5G networks. The scheme uses passwords, smart cards, and biometric authentication to provide users with more comprehensive security. Each IoT device has a unique media access control (MAC) address, which we enter as the biometric information of the user:

Table 1: Abbreviations.

5G	Fifth-generation
CDN	Content delivery network
IoT	Internet of Things
MSAS	Multiserver authentication scheme
MAC	Media access control
ROR	Real-Oracle-Random

(i) The research shows that our scheme can guarantee anonymity, perfect forward security, and anti-impersonation attacks, and the smart card is a stolen attack in a multiserver architecture

(ii) In our scheme, users and servers can complete mutual authentication without passing the registration center to avoid the communication load caused by excessive user traffic

(iii) In the random oracle model, we proposed a hypothesis based on the elliptic curve discrete logarithm problem. This reveals that the proposed scheme has a secure mutual authentication process. The informal security analysis and ProVerif tool proof reveal that the proposed scheme can resist common network attacks and has a secure and complete process of generating the session key

(iv) Our protocol has better performance. In a series of related schemes, the proposed scheme has relatively low communication and computation costs. This scheme is more suitable for servers in 5G communication to provide services to users

Some abbreviations used in this paper are shown in Table 1. The remainder of this paper is organized as follows.

Section 2 introduces the related work. A detailed description of the proposed scheme is provided in Section 3. Section 4 provides a formal and informal security analysis of the proposed scheme. Performance analysis of the schemes is presented in Section 5, and the study is concluded in Section 6.

2. Related Work

The earliest MSAS [12] was aimed at neural networks. Because of the time consumption of training neural networks, numerous enhanced MSASs have been proposed. In the process of remote authentication, it is not realistic to use only passwords. Using a smart card is a particularly effective resolution for user authentication and key management [13–15]. Lin et al. [16] introduced a remote MSAS without a verification table in 2003. Cao and Zhong [17] pointed out that [16] is insecure and exposed to serious user impersonation attacks. In addition, in 2004, Juang [18] pointed out that each user in [16] needs to use a large memory to store relevant parameters, which is not suitable for applications using the smart card. Moreover, [12, 16] do not generate a session key; therefore, there is a certain security risk in the communication process. Therefore, they designed an MSAS using a smart card and password. In the same year, Chang and Lee [19] thought that Juang's scheme [18] performed significant computation in smart cards; therefore, they proposed an efficient MSAS based on the smart card. In a multiserver environment, multiple servers are required to provide services to users, and providing strong anonymity is more secure for users. In 2009, Liao and Wang [20] introduced a scheme using dynamic identity and smart card authentication and concluded that their scheme met all requirements in an MS environment. However, Hsiang and Shih [21] discovered that [20] cannot resist insider attacks and spoofing attacks. To address the above security loopholes, they submitted an enhanced MSAS using smart cards and dynamic identity.

Because passwords and smart cards may be forgotten or lost, human biometrics are added to the design of key management and authentication protocols [22], such as fingerprint, face, and iris. In 2010, Li and Hwang [23] introduced a remote MSAS based on biometrics and smart cards and declared that their scheme can resist masquerade attacks, replay attacks, and smart card stolen attacks and present mutual authentication and nonrepudiation. In 2011, Li et al. [24] found that [23] had some security flaws; that is, they did not provide correct authentication and could not resist man-in-the-middle attacks and impersonation attacks. Further, they submitted an advanced remote MSAS using smart cards and biometrics. To ensure perfect forward security and reduce the computation consumption of smart cards, Yoon and Yoo [25] proposed a distributed MSAS without a verification table based on biometrics and smart cards in 2013. Liao and Hsiao [26] introduced a remote MSAS based on pairing. However, Kim et al. [27] found that [25] could not resist offline password guessing attacks; therefore, an enhanced solution was proposed to overcome this vulnerability. Hsieh and Leu [28] pointed out that [26] would be subject to tracking attacks, and there was no preverification phase. Therefore, they improved the MSAS for mobile users using a self-certified public key. In 2014, Chuang and Chen [29] proposed a lightweight MSAS that guarantees anonymity and claimed that it can resist a variety of attacks. Mishra et al. [30] showed that [29] could not resist impersonation attacks, smart card stolen attacks, and denial of service attacks. They then submitted an enhanced MSAS based on [29]. However, Lu et al. [31] proved that [30] is vulnerable to server impersonation attacks and lacks perfect forward security in 2015. Therefore, they proposed an MSAS based on three factors. In 2016, He et al. [32] introduced an anonymous MSAS using self-certified public key encryption. Li et al. [33] pointed out that [32] would be subject to offline password guessing attacks and impersonation attacks in 2019. Furthermore, they designed a secure MSAS for key management in a cloud computing environment. It is worth noting that Chuang and Tseng [34] proposed a compatible cross-species authentication and key exchange protocol and realized independent authentication and member revocation. To solve the performance problem of low-power clients, Tseng et al. [35] proposed a lightweight identity-based mutual authentication and key exchange protocol for resource-constrained devices. Some important related works are summarized in Table 2.

3. Background and Scheme

3.1. Preliminaries

3.1.1. Hash Function. In this section, we will introduce the basics of hash functions. The hash function $H(\cdot)$ takes the variable-length data block M as the input to generate a fixed-length hash value $h = H(M)$ and satisfies the following conditions:

(1) *One-Way*. Given h, it is difficult to compute M such that $h = H(M)$.

(2) *Weak Collision Resistance*. Given M_1, it is difficult to find $M_2 \neq M_1$ so that $H(M_1) = H(M_2)$.

(3) *Strong Collision Resistance*. It is difficult to find $M_1 \neq M_2$ so that $H(M_1) = H(M_2)$.

3.1.2. Bilinear Pairing. Suppose F_n^* is a finite field on an elliptic curve and n is a large prime number. G_1 is an additive cyclic subgroup with P as a generator on F_n^*, and G_2 is a multiplication group with n as order. For points $P, Q \in G_1$, the bilinear pairing [33–35] is a mapping, $e : G_1 \times G_1 \longrightarrow G_2$, which has the following properties:

(1) *Bilinear*. $e(aP, bQ) = e(P, Q)^{ab}$, where $a, b \in F_n^*$.

(2) *Nondegenerate*. $e(P, Q) \neq 1$.

(3) *Computability*. There is an efficient algorithm to compute $e(P, Q)$.

3.2. Proposed Scheme. We describe a client-server key management scheme in which the client can be a user or an IoT device. The scheme is divided into four phases:

Table 2: Cryptographic techniques and limitations.

Scheme	Cryptographic techniques	Limitations
Ying and Nayak [10]	(i) Utilized ECC (ii) Utilized a self-certified public key (iii) Based on the Diffie–Hellman problem	(i) Does not provide untraceability (ii) Does not provide two-factor security (iii) Does not resist offline identity guessing attacks (iv) Does not resist offline password guessing attacks (v) Does not resist user impersonation attacks
Yoon and Yoo [25]	(i) Based on biometrics (ii) Utilized ECC (iii) Based on a smart card	(i) Does not resist offline password guessing attacks
Liao and Hsiao [26]	(i) Based on bilinear pairings (ii) Utilized a self-certified public key (iii) Based on the Diffie–Hellman problem	(i) Does not resist tracking attacks (ii) Does not provide preverification
Chuang and Chen [29]	(i) Utilized a one-way hash function (ii) Based on biometrics (iii) Based on a smart card	(i) Does not resist impersonation attacks (ii) Does not resist smart card stolen attacks (iii) Does not resist denial of service attacks
Mishra et al. [30]	(i) Utilized a one-way hash function (ii) Based on biometrics (iii) Based on a smart card	(i) Does not resist server impersonation attacks (ii) Does not provide perfect forward security
He et al. [32]	(i) Based on bilinear pairings (ii) Utilized a self-certified public key (iii) Based on the Diffie–Hellman problem	(i) Does not resist offline password guessing attacks (ii) Does not resist impersonation attacks
Li et al. [33]	(i) Based on bilinear pairings (ii) Based on the Diffie–Hellman problem (iii) Utilized a one-way hash function	—
Chuang and Tseng [34]	(i) Based on bilinear pairings (ii) Based on the Diffie–Hellman problem (iii) Utilized a one-way hash function	—
Tseng et al. [35]	(i) Based on bilinear pairings (ii) Based on the Diffie–Hellman problem (iii) Utilized a one-way hash function	—

initialization, registration phase, time key update phase, and key management phase. The symbols and descriptions of the scheme are listed in Table 3.

3.2.1. Initialization. The RC sets up an elliptic curve and chooses the parameters. RC selects two elliptic curve groups G_1 and G_2 with the same order q. Subsequently, RC defines a bilinear pairing $e : G_1 \times G_1 \longrightarrow G_2$ and computes $g = e(P, P)$, where P is a generator of G_1. RC chooses a random number x as its private key, and $P_{pub} = x \cdot P$ is the corresponding public key. Finally, RC generates two hash functions, $HG(\cdot)$ and $H(\cdot)$, and publicizes $\{G_1, G_2, q, P, P_{pub}, g, HG(\cdot), H(\cdot)\}$.

3.2.2. Registration Phase. To join the system, both U_i and S_j must register with the RC to verify their legality. This phase includes the U_i and S_j registration.

(1) Server Registration Phase. The registration steps of S_j are shown as follows:

(1) Server S_j chooses its identity SID_{S_j} and a random number r_{sj} and computes the public key $P_{sj} = g^{r_{sj}}$. Thereafter, S_j sends $\{P_{sj}, SID_{S_j}\}$ to RC via a secure channel

(2) After receipt, RC computes $d_{sj} = x \cdot HG(SID_{S_j})$ and $P_{rsj} = (P_{sj})^x \oplus d_{sj}$ and then sends $\{P_{rsj}\}$ back to S_j

(3) S_j computes $d_{sj} = P_{rsj} \oplus (P_{pub})^{r_{sj}}$ as its private key and then checks if $e(d_{sj}, P) = ^{?} e(HG(SID_{S_j}), P_{pub})$. If not equal, S_j rejects messages. If equal, S_j accepts d_{sj} and stores $\{d_{sj}, r_{sj}\}$

(4) RC stores all servers' status and identities SID_{S_j} in a table Tab_{S_j}

(2) User Registration Phase. The registration steps of U_i are shown as follows:

(1) U_i selects identity ID_{U_i}, password PW_{U_i}, and biometric BIO_{U_i}. Note that the MAC address of an IoT device is the BIO_{U_i}. Thereafter, U_i generates a random number r_{ui} and computes $Gen(BIO_{ui}) = (\theta_{ui}, \sigma_{ui})$, $VID_{ui} = H(ID_{U_i} \| r_{ui})$, $FVID_{ui} = VID_{ui} \oplus (P_{pub})^{r_{ui}}$, $VPW_{ui} = H(PW_{U_i} \| VID_{ui} \| r_{ui})$, $FVPW_{ui} = VPW_{ui} \oplus (P_{pub})^{r_{ui}}$, $VBI_{ui} = H(\sigma_{ui} \| r_{ui})$, $FVBI_{ui} = VBI_{ui} \oplus$

Table 3: Symbols and descriptions.

Symbol	Description
U_i	User and IoT device
S_j	Server
RC	Registration center
$\mathscr{A}$	Adversary
SC	Smart card
SK_{U_i}	The session key of U_i
SK_{S_j}	The session key of S_j
x	The private key of the RC
d_{sj}	The private key of S_j
BIO_{U_i}	The biometrics of U_i
$Gen(\cdot)/Rep(\cdot)$	Fuzzy generator/reproduction function
$H(\cdot)$	Hash function
$HG(\cdot)$	Map-to-point hash function
$\|$	Connect operation
$\oplus$	Exclusive or operation
Π_U^x	The x-th communication of user U_i
Π_S^y	The y-th communication of server S_j
Π_{RC}^z	The z-th communication of registry RC
q_{hash}	The number of times to make the *Hash* query
q_{send}	The number of times to make the *Send* query
GM_m	The m-th game
$Succ_{\mathscr{A}}^{GM_m}(\xi)$	The event that $\mathscr{A}$ succeeds in the game GM_m
$\Pr\left[Succ_{\mathscr{A}}^{GM_m}(\xi)\right]$	The probability of the event $Succ_{\mathscr{A}}^{GM_m}(\xi)$

$(P_{pub})^{r_{ui}}$, and $P_{ui} = g^{r_{ui}}$ and sends $\{ID_{U_i}, FVID_{ui}, P_{ui}, FVPW_{ui}, FVBI_{ui}\}$ to RC

(2) Upon receiving, RC computes $VPW_{ui} = FVPW_{ui} \oplus (P_{ui})^x$, $VBI_{ui} = FVBI_{ui} \oplus (P_{ui})^x$, $VID_{ui} = FVID_{ui} \oplus (P_{ui})^x$, $d_{ui} = x \cdot HG(VID_{ui})$, $V_{ui} = H(VPW_{ui} \| VBI_{ui} \| VID_{ui})$, and $R_{sj} = e(d_{ui}, HG(SID_{S_j})) \oplus VBI_{ui} \oplus VPW_{ui}$. Further, RC creates table Tab_{U_i}, which includes all servers' SID_{S_j}, R_{sj}, P_{sj}, and Table $Tab_{S_{U_i}}$, which includes all users' $H(ID_{U_i})$, P_{ui}, and *status*. Subsequently, RC injects $\{Tab_{U_i}, V_{ui}\}$ to the smart card SC and sends it to U_i

(3) After receiving, U_i computes $E_{ui} = r_{ui} \oplus H(\sigma_{ui})$. Finally, U_i stores $\{E_{ui}, \theta_{ui}, P_{ui}\}$ into SC and deletes the other parameters

In other words, if a new client (including users and IoT devices) wants to join the client-server communication construction in 5G, they need to register with the RC according to the above steps in the user registration phase. After registration, the client can obtain the legal identity and corresponding information and communicate with the server.

3.2.3. Time Key Update Phase. In this phase, the RC checks the *status* of users in $Tab_{S_{U_i}}$ and dynamically distributes time keys to legitimate users. Communication in this phase occurs in public channels. The details are as follows:

(1) RC queries the *status* of the user's $H(ID_{U_i})$ in $Tab_{S_{U_i}}$. When *status* is "OK," RC selects a random number b_{ui} and adds b_{ui} into $Tab_{S_{U_i}}$. Thereafter, RC chooses a time-valid period t and computes $B_{ui} = g^{b_{ui}}$, $C_{ui} = (P_{ui})^{b_{ui}}$, and $T_{ui} = H(B_{ui} \| C_{ui} \| t)$. Further, RC sends $\{B_{ui}, T_{ui}, t\}$ to U_i via a public channel

(2) After receiving, U_i inserts their SC, inputs ID_{U_i} andBIO_{U_i}, and computes $\sigma_{ui} = Rep(BIO_{U_i}, \theta_{ui})$, $r_{ui} = E_{ui} \oplus H(\sigma_{ui})$, and $C_{ui} = (B_{ui})^{r_{ui}}$. Subsequently, U_i checks if $T_{ui} \stackrel{?}{=} H(B_{ui} \| C_{ui} \| t)$. If the equation holds, U_i accepts the time key and stores $\{C_{ui}, t\}$ into SC

3.2.4. Key Management Phase. U_i inserts SC to log into the system. Subsequently, U_i can authenticate and establish a session key using S_j. The detailed steps are as follows:

(1) U_i inputs ID_{U_i}, PW_{U_i}, and BIO_{U_i} and inserts their SC. U_i then computes $\sigma_{ui} = Rep(BIO_{ui}, \theta_{ui})$, $r_{ui} = E_{ui} \oplus H(\sigma_{ui})$, $VID_{ui} = H(ID_{U_i} \| r_{ui})$, $VPW_{ui} = H(PW_{U_i} \| VID_{ui} \| r_{ui})$, and $VBI_{ui} = H(\sigma_{ui} \| r_{ui})$ and checks if $V_{ui} \stackrel{?}{=} H(VPW_{ui} \| VBI_{ui} \| VID_{ui})$. The authorized user U_i logs into the system. Thereafter, the card reader queries Tab_{U_i} and displays all the server identities SID_{S_j}. U_i selects a server's SID_{S_j} and chooses a random number α. Further, U_i computes $Y_{ui} = g^{\alpha}$, $Q_{ui} = (P_{sj})^{r_{ui}}$, $F_{ui} = VID_{ui} \oplus H(Y_{ui} \| SID_{S_j} \| Q_{ui})$, and $V_1 = H(F_{ui} \| Y_{ui} \| P_{ui} \| VID_{ui} \| C_{ui})$ and sends $\{F_{ui}, Y_{ui}, P_{ui}, V_1\}$ to S_j

(2) Upon receiving, server S_j computes $Q_{sj} = (P_{ui})^{r_{sj}}$, $C_{ui} = (P_{ui})^{b_{ui}}$, and $VID'_{ui} = F_{ui} \oplus H(Y_{ui} \| SID_{S_j} \| Q_{sj})$ and checks if $V_1 \stackrel{?}{=} H(F_{ui} \| Y_{ui} \| P_{ui} \| VID'_{ui} \| C_{ui})$. When the equation holds, S_j generates a random number β and computes $e_{sj} = e(d_{sj}, HG(VID_{ui}))$, $Y_{sj} = g^{\beta}$, and $V_2 = H(e_{sj} \| Y_{sj} \| Y_{ui} \| VID_{ui} \| SID_{S_j} \| Q_{sj})$ and sends $\{V_2, Y_{sj}\}$ to U_i

(3) Upon receiving, U_i computes $e_{ui} = R_{sj} \oplus VPW_{ui} \oplus VBI_{ui}$. Subsequently, U_i checks if $V_2 \stackrel{?}{=} H(e_{sj} \| Y_{sj} \| Y_{ui} \| VID_{ui} \| SID_{S_j} \| Q_{sj})$ holds. When the equation holds, U_i computes $key_{ui} = (Y_{sj})^{\alpha}$ and establishes the session key $SK_{U_i} = H(key_{ui} \| Q_{ui} \| VID_{ui} \| SID_{S_j})$. Further, U_i computes $V_3 = H(SK_{U_i} \| VID_{ui} \| SID_{S_j})$ and sends it to S_j

(4) After receipt, S_j computes $key_{sj} = (key_{ui})^{\beta}$ and session key $SK_{S_j} = H(key_{sj} \| Q_{sj} \| VID_{ui} \| SID_{S_j})$. Finally,

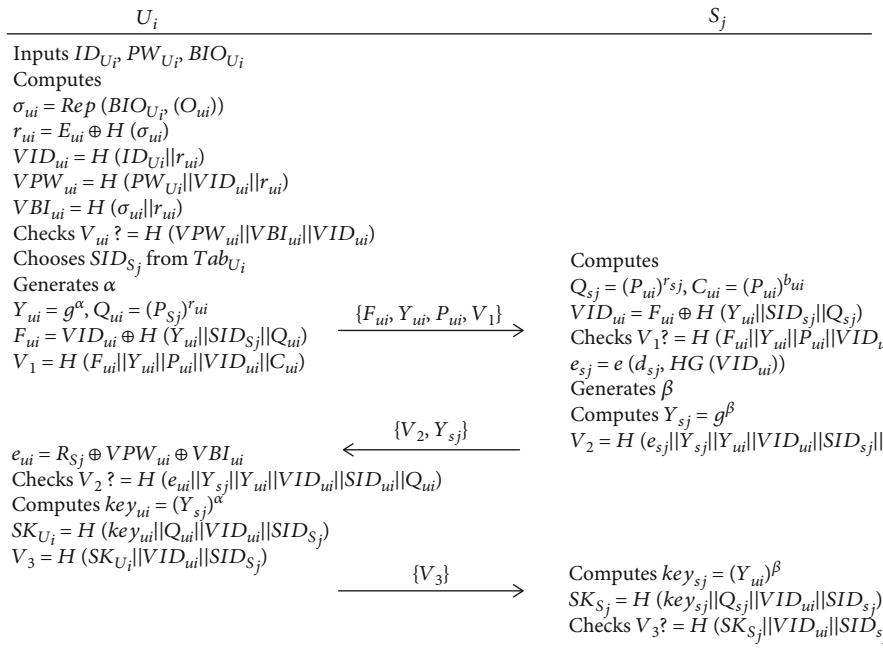

Figure 2: Key management phase.

S_j verifies whether $V_3 = ^{?} H(SK_{S_j}||VID_{ui}||SID_{S_j})$ and accepts the session key when the equation holds

The key management phase is shown in Figure 2.

4. Security Analysis

4.1. Formal Security Analysis Based on Real-Oracle-Random. We conducted a formal security analysis of the scheme under the Real-Oracle-Random (ROR) model [30, 32, 33, 36, 37]. The ROR model is used to simulate ideal hash functions. We assume that ideal hash functions are random and uniformly distributed. Suppose Π_U^x, Π_S^y, and Π_{RC}^z represent the x-th communication of user U_i, the y-th communication of server S_j, and the z-th communication of registry RC, respectively. The Π_U^x, Π_S^y, and Π_{RC}^z act to simulate the real communications of U_i, S_j, and RC. Note that $\mathcal{O} = \{\Pi_U^x, \Pi_S^y, \Pi_{RC}^z\}$.

4.1.1. Queries. Subsequently, adversary $\mathcal{A}$ verifies the security of the protocol with the following query:

(1) $Execute(\mathcal{O})$. Starting the query, $\mathcal{A}$ obtains message records of transmission in the public channel. The execution of this query is a passive attack.

(2) $Hash(string)$. Starting the query, $\mathcal{A}$ can enter a *string* and then obtain the corresponding hash value.

(3) $Send(\mathcal{O}, M)$. Starting the query, $\mathcal{A}$ sends M to $\mathcal{O}$ and receives a response from $\mathcal{O}$.

(4) $Corrupt(\mathcal{O})$. Starting the query, $\mathcal{A}$ obtains one of the private values from $\mathcal{O}$.

(5) $Test(\mathcal{O})$. Starting the query, $\mathcal{A}$ flips coin $\mathcal{C}$ and attempts to determine the correctness of the session key. There are only two results for flipping $\mathcal{C}$: $\mathcal{C}$ = 1 or $\mathcal{C} = 0$. The former means that $\mathcal{A}$ receives the session key, and the latter means that $\mathcal{A}$ receives a random string.

4.1.2. Definitions. The proposed scheme involves the discrete logarithm problem on an elliptic curve (ECDLP) over a finite field F_n^*, which is defined by the following. On the elliptic curve E, given the points H and Q of order n, and $H = aQ$, where H and Q belong to E and a to F_n^*, in the polynomial time ξ, the probability that $\mathcal{A}$ obtains a that satisfies $H = aQ$ is $Adv_{\mathcal{A}}^{ECDLP}(\xi) = \Pr[\mathcal{A}(Q, aQ) = a : a \in F_n^*, Q \in E]$. Furthermore, for a sufficiently small η, we have the following results: $Adv_{\mathcal{A}}^{ECDLP}(\xi) < \eta$.

4.1.3. Theorem. If $\mathcal{A}$ queries in polynomial time ξ, the advantage of breaking scheme S is $Adv_{\mathcal{A}}^{S}(\xi) \le 2 \max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\} + 2q_{hash}Adv_{\mathcal{A}}^{ECDLP}(\xi) + q_{hash}^2/2^l + (q_{send} + q_{hash})/2^{l-1}$, where q_{hash} is the number of times to achieve the *Hash* query, q_{send} is the number of times to achieve the *Send* query, l is the length of the password, and C' and s' are constants.

Proof. We define the game sequence $GM_m (m = 0, 1, 2, 3, 4, 5, 6)$ to prove the theorem. $Succ_{\mathcal{A}}^{GM_m}(\xi)$ is the event that $\mathcal{A}$ succeeds in the game GM_m. □

(1) GM_0. GM_0 means starting the game without queries. At this time, the advantage of $\mathcal{A}$ breaking S is

$$Adv_{\mathcal{A}}^{S}(\xi) = \left|2 \Pr\left[Succ_{\mathcal{A}}^{GM_0}(\xi)\right] - 1\right|. \quad (1)$$

(2) GM_1. GM_1 starts executing the *Execute* query. Because the *Execute* query can only receive messages $\{B_{ui}, T_{ui}, t\}$, $\{F_{ui}, Y_{ui}, P_{ui}, V_1\}$, $\{V_2, Y_{sj}\}$, and $\{V_3\}$ transmitted through the public channel,

$$\Pr\left[Succ_{\mathscr{A}}^{GM_1}(\xi)\right] = \Pr\left[Succ_{\mathscr{A}}^{GM_0}(\xi)\right]. \tag{2}$$

(3) GM_2. GM_2 begins to achieve the *Send* query. Based on Zipf's law [38], we obtain

$$\left|\Pr\left[Succ_{\mathscr{A}}^{GM_2}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_1}(\xi)\right]\right| \le \frac{q_{send}}{2^l}. \tag{3}$$

(4) GM_3. GM_3 begins to achieve the *Hash* query. Based on the birthday paradox, we obtain

$$\left|\Pr\left[Succ_{\mathscr{A}}^{GM_3}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_2}(\xi)\right]\right| \le \frac{q_{hash}^2}{2^{l+1}}. \tag{4}$$

(5) GM_4. GM_4 starts to judge the security of the session key, mainly based on the following two attacks:

(i) *Perfect Forward Security*. $\mathscr{A}$ uses Π_{RC}^z to obtain x of RC and verify whether protocol S can provide perfect forward security.

(ii) *Known Session-Specific Temporary Information Attacks*. $\mathscr{A}$ uses Π_U^x or Π_S^y or Π_{RC}^z to obtain temporary information and verify whether protocol S can resist the attack.

In the above two cases, the ECDLP must be solved to calculate $SK_{U_i} = SK_{S_j}$. For $SK_{U_i} = H(key_{ui}\|Q_{ui}\|VID_{ui}\|SID_{S_j})$, in the first case, suppose $\mathscr{A}$ can calculate $d_{sj} = x \cdot HG(SID_{S_j})$ through x, but $key_{ui} = (Y_{sj})^{\alpha}andQ_{ui} = (P_{sj})^{r_{ui}}$ is unknown, which needs to solve ECDLP twice; in the second case, suppose $\mathscr{A}$ obtains the random number α of U_i and further calculates $key_{ui} = (Y_{sj})^{\alpha}$, but Q_{ui} and r_{ui} are still unknown. The above analysis is also true for $SK_{S_j} = H(key_{sj}\|Q_{sj}\|VID_{ui}\|SID_{S_j})$. Therefore,

$$\left|\Pr\left[Succ_{\mathscr{A}}^{GM_4}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_3}(\xi)\right]\right| \le q_{hash}Adv_{\mathscr{A}}^{ECDLP}(\xi). \tag{5}$$

(6) GM_5. GM_5 uses Π_U^x to obtain the information $\{Tab_{U_i}, V_{ui}, E_{ui}, \theta_{ui}, P_{ui}, C_{ui}, t\}$ stored in SC. Subsequently, $\mathscr{A}$ uses the information to launch offline password guessing attacks or stolen smart card attacks. $\mathscr{A}$ calculates $V_{ui} = H(VPW_{ui}\|VBI_{ui}\|VID_{ui})$, where $VID_{ui} = H(ID_{U_i}\|r_{ui})$, $VPW_{ui} = H(PW_{U_i}\|VID_{ui}\|r_{ui})$, and $VBI_{ui} = H(\sigma_{ui}\|r_{ui})$. However, σ_{ui}, r_{ui}, ID_{U_i}, PW_{U_i}, and BIO_{U_i} are confidential. The probability that $\mathscr{A}$ can successfully guess the biological information of l-bit is $1/2^l$ [39], which is an extremely small value. Based on Zipf's law [38], we have

$$\begin{aligned}&\left|\Pr\left[Succ_{\mathscr{A}}^{GM_5}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_4}(\xi)\right]\right| \\ &\le \max\left\{C' \cdot q_{send}^{s'}, \frac{q_{send}}{2^l}\right\},\end{aligned} \tag{6}$$

where C' and s' are constants.

(7) GM_6. GM_6 starts to execute query $H(key_{ui}\|Q_{ui}\|VID_{ui}\|SID_{S_j})$ or $H(key_{sj}\|Q_{sj}\|VID_{ui}\|SID_{S_j})$ to verify whether protocol S can resist the key compromise impersonation attacks. At this time, the advantage of $\mathscr{A}$ breaking S is

$$\left|\Pr\left[Succ_{\mathscr{A}}^{GM_6}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_5}(\xi)\right]\right| \le \frac{q_{hash}}{2^l}. \tag{7}$$

Because the probability of success and failure of GM_6 is equal,

$$\Pr\left[Succ_{\mathscr{A}}^{GM_6}(\xi)\right] = \frac{1}{2}. \tag{8}$$

According to formulas (1)–(8), we have

$$\begin{aligned}\frac{1}{2}Adv_{\mathscr{A}}^{S}(\xi) &= \left|\Pr\left[Succ_{\mathscr{A}}^{GM_0}(\xi)\right] - \frac{1}{2}\right| \\ &= \left|\Pr\left[Succ_{\mathscr{A}}^{GM_0}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_6}(\xi)\right]\right| \\ &= \left|\Pr\left[Succ_{\mathscr{A}}^{GM_1}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_6}(\xi)\right]\right| \\ &\le \sum_{m=0}^{5}\left|\Pr\left[Succ_{\mathscr{A}}^{GM_{m+1}}(\xi)\right] - \Pr\left[Succ_{\mathscr{A}}^{GM_m}(\xi)\right]\right| \\ &= \max\left\{C' \cdot q_{send}^{s'}, \frac{q_{send}}{2^l}\right\} + q_{hash}Adv_{\mathscr{A}}^{ECDLP}(\xi) \\ &\quad + \frac{q_{hash}^2}{2^{l+1}} + \frac{(q_{send} + q_{hash})}{2^l}.\end{aligned} \tag{9}$$

Further, we have $Adv_{\mathscr{A}}^{S}(\xi) \le 2\max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\} + 2q_{hash}Adv_{\mathscr{A}}^{ECDLP}(\xi) + q_{hash}^2/2^l + (q_{send} + q_{hash})/2^{l-1}$.

4.2. Formal Security Analysis Based on ProVerif. ProVerif, which is mainly used for the automatic verification of cryptographic-related security protocols, is a formal analysis tool based on the Dolev-Yao model [40] and computational model proposed by Abadi et al. and Blanchet et al. [41, 42]. The ProVerif tool [43] is based on the equivalence theory of function reduction, definition of terms and processes, structural equivalence between extended processes, and others and is applied to the security analysis of the real environment. Furthermore, the analysis and verification with ProVerif have security properties, such as confidentiality, authentication, and logic. Our proposed protocol was analyzed using ProVerif, as follows.

```
(**********channel**********)
free ch: channel.
free sch: channel [private].
(**********shared keys**********)
free SKUi: bitstring [private].
free SKSj: bitstring [private].
free IDUi: bitstring [private].
free SIDSj: bitstring [private].
(**********constants**********)
free x: bitstring [private].
free g: bitstring.
free P: bitstring.
free Ppub : bitstring.
(**********functions & reductions & equations*********)
fun H (bitstring): bitstring.
fun HG (bitstring): bitstring.
fun mult (bitstring, bitstring): bitstring.
fun bilinearmap (bitstring, bitstring): bitstring.
fun exp (bitstring, bitstring): bitstring.
fun con (bitstring, bitstring): bitstring.
reduc forall m: bitstring, n: bitstring; getmess (con (m, n)) = m.
```

Figure 3: Definitions.

```
(**********functions & reductions & equations*********)
fun xor (bitstring,bitstring): bitstring.
equation forall m: bitstring, n: bitstring; xor (xor (m, n), n) = m.
fun Gen (bitstring): bitstring.
fun Rep (bitstring, bitstring): bitstring.

(**********queries**********)
query attacker (SKUi).
query attacker (SKSj).
query attacker (IDUi).
query attacker (SIDSj).
query inj-event (UserAuthed ()) ==> inj-event (UserStarted ()).
query inj-event (UserAcServer ()) ==> inj-event (ServerAcUser ()).
query inj-event (ServerAcSK ()) ==> inj-event (UserAcServer ()).

(********** event**********)
event UserStarted ().
event UserAuthed ().
event ServerAcUser ().
event UserAcServer ().
event ServerAcSK ().
```

Figure 4: Queries and events.

Some constants and functions were defined as shown in Figure 3. Among them, *hash*, *concatenation*, and *xor* are common operations; *Bilinearmap*() is a bilinear pairing operation; exp (*bitstring*, *bitstring*) are exponential operations, with the first *bitstring* as the base and the last *bitstri ng* as the exponent. The anonymity and consistency of the protocol were analyzed using events and queries. As shown in Figure 4, the events *UserStarted*(), *UserAuthed*(), and *UserAcServer*(), respectively, indicate that the user starts authentication, completes login, and successfully authenticates the server; events *ServerAcUser*() and *ServerAcSK*(), respectively, indicate that the server successfully authenticates the user and completes the authentication of the session key.

Figures 5(a) and 5(b) show the specific operations of each entity and describe the authentication process between the user and the server. The results of the query using ProVerif are shown in Figure 6. The first and second results confirm that the session key is secure, and our scheme can resist key compromise impersonation attacks. The third and fourth results reveal that the proposed scheme can guarantee anonymity and resist offline password guessing attacks. The $inj-event(UserAuthed()) = \Rightarrow inj-event(UserStarted())$ indicates that the user logs in after authentication. The $inj-event(UserAcServer()) = \Rightarrow inj-event(ServerAcUser())$ indicates that user authentication is performed after the server completes authentication. The $inj-event(ServerAcSK()) = \Rightarrow inj-event(UserAcServer())$ indicates that the server verifies the session key after the user completes authentication. In other words, the proposed scheme maintains consistency.

4.3. Informal Analysis

4.3.1. Insider Attacks. In the proposed scheme, $\{ID_{U_i}, FVID_{ui}, P_{ui}, FVPW_{ui}, FVBI_{ui}\}$ are sent to RC when U_i is registered, where $FVID_{ui} = VID_{ui} \oplus (P_{pub})^{r_{ui}}$, $P_{ui} = g^{r_{ui}}$, $FVPW_{ui} = VPW_{ui} \oplus (P_{pub})^{r_{ui}}$, and $FVBI_{ui} = VBI_{ui} \oplus (P_{pub})^{r_{ui}}$. In this process, U_i does not directly transmit passwords or biometrics. Insiders in RC cannot compute real PW_{ui} and BIO_{ui}. Therefore, the proposed scheme can resist insider attacks.

4.3.2. User Impersonation Attacks. Malicious adversary $\mathcal{A}$ attempts to impersonate legitimate users in communicating with S_j. (1) $\mathcal{A}$ intercepts $\{F_{ui}, Y_{ui}, P_{ui}, V_1\}$, selects α', and computes $Y'_{ui} = g^{\alpha'}$, $Q_{ui} = (P_{sj})^{r_{ui}}$, $F'_{ui} = VID_{ui} \oplus H(Y'_{ui} \| SID_{S_j} \| Q_{ui})$, and $V_1 = H(F'_{ui} \| Y'_{ui} \| P_{ui} \| VID_{ui} \| C_{ui})$, where $VID_{ui} = H(ID_{U_i} \| r_{ui})$ and $C_{ui} = (B_{ui})^{r_{ui}}$. However, ID_{U_i} and r_{ui} are confidential to $\mathcal{A}$. Therefore, when S_j verifies V_1, it will reject $\mathcal{A}$. (2) If $\mathcal{A}$ intercepts $\{V_2, Y_{sj}\}$ and forges $\{V_3\}$, it attempts to pass S_j's validation for SK_{U_i}. However, $SK_{U_i} = H(key_{ui} \| Q_{ui} \| VID_{ui} \| SID_{S_j})$, where key_{ui}, Q_{ui}, and VID_{ui} are all confidential to $\mathcal{A}$. Therefore, S_j declines to generate a session key with $\mathcal{A}$. The above analysis indicates that the proposed scheme can resist user impersonation attacks.

4.3.3. Server Impersonation Attacks. Malicious adversary $\mathcal{A}$ attempts to impersonate S_j to communicate with legitimate users. $\mathcal{A}$ intercepts $\{F_{ui}, Y_{ui}, P_{ui}, V_1\}$ and $\{V_2, Y_{sj}\}$, selects β', and computes $Y'_{sj} = g^{\beta'}$ and $V'_2 = H(e_{sj} \| Y'_{sj} \| Y_{ui} \| VID_{ui} \| SID_{S_j} \| Q_{sj})$, where $e_{sj} = e(d_{sj}, HG(VID_{ui}))$, $VID_{ui} = H(ID_{U_i} \| r_{ui})$, and $Q_{sj} = (P_{ui})^{r_{sj}}$ are unknown to $\mathcal{A}$. Therefore, $\mathcal{A}$ cannot be forged as the S_j to communicate with U_i. In other words, our scheme can resist server impersonation attacks.

4.3.4. Replay Attacks. In our scheme, whenever U_i starts a new session with S_j, new α and β will participate in the session. $Y_{ui} = g^{\alpha}$, $Y_{sj} = g^{\beta}$, $key_{ui} = (Y_{sj})^{\alpha}$, and $key_{sj} = (key_{ui})^{\beta}$ are updated in each round, where Y_{ui} and Y_{sj} are used for

```
(*************User's process************)
let ProcessUser =
new IDUi: bitstring;
new PWUi: bitstring;
new BIOUi: bitstring;
new rui: bitstring;
let (a: bitstring, b: bitstring) = Gen (BIOUi) in
let VIDui = H (con (IDUi, rui)) in
let FVIDui = xor (VIDui, exp (Ppub, rui)) in
let VPWui = H (con (PWUi, con (VIDui, rui))) in
let FVPWui = xor (VPWui, exp (Ppub, rui)) in
let VBIui = H (con (b, rui)) in
let FVBIui = xor (VBIui, exp (Ppub,rui)) in
let Pui = exp (g, rui) in
out(sch, (IDUi, FVIDui, Pui, FVPWui, FVBIui));
in (sch, (xTabUi: bitstring, xVui: bitstring));
let Eui = xor (rui, H (b)) in
! (in (ch, (xBui: bitstring, xTui: bitstring, xt: bitstring));
let Cui = exp (xBui, rui) in
event UserStarted ();
let b' = Rep (BIOUi,a) in
let rui = xor (Eui, H(b')) in
let VIDui = H (con (IDUi, rui)) in
let VPWui = H (con (PWUi, con (VIDui, rui))) in
let VBIui = H (con (b', rui)) in
let Vui' = H (con (VPWui, con (VBIui, VIDui))) in
if Vui' = xVui then event UserAuthed (); (***authentication***)
new xSIDSj: bitstring;
new xPsj: bitstring;
new A: bitstring;
let Yui = exp (g, A) in
let Qui = exp (xPsj, rui) in
let Fui = xor (VIDui, H (con (Yui, con (xSIDSj, Qui)))) in
let V1 = H (con (Fui, con (Yui, con (Pui, con (VIDui, Cui))))) in
out (ch, (Fui, Yui, Pui, V1));
in (ch, (xV2: bitstring, xYsj: bitstring));
new xRsj: bitstring;
let eui = xor (xRsj, xor (VPWui, VBIui)) in
let V2' = H (con (Qui, con (econ (xYsj, VIDui), econ (Yui, eui)))) in
if V2' = xV2 then event UserAcServer ();
let keyui = exp (xYsj, A) in
let SKUi = H (con (keyui, con (Qui, con (VIDui, xSIDSj)))) in
let V3 = H (con (SKUi, con (VIDui, xSIDSj))) in
out (ch, (V3));
0).
(*************Server's process************)
let ProcessServer =
new SIDSj: bitstring; (* the Server's identity *)
new rsj: bitstring;
let Psj = exp (g, rsj) in
out (sch, (Psj, SIDSj));
in (sch, (yPrsj: bitstring));
let dsj = xor (yPrsj, exp (Ppub, rsj)) in
if bilinearmap (dsj, P) = bilinearmap (HG(SIDSj), mult (x, P)) then
```

(a) Process

```
! (in (ch, (yFui: bitstring, yYui: bitstring, yPui: bitstring, yV1: bitstring));
new ybui: bitstring;
let Qsj = exp (yPui, rsj) in
let Cui = exp (yPui, ybui) in
let VIDui = xor (yFui, H (con (yYui, con (SIDSj, Qsj)))) in
let V1' = H (con (yFui, con (yYui, con (yPui, con (VIDui, Cui))))) in
if V1' = yV1 then event ServerAcUser ();
new B: bitstring;
let esj = bilinearmap (dsj, HG (VIDui)) in
let Ysj = exp (g, B) in
let V2 = H (con (esj, con (Ysj, con (yYui, con (VIDui, con (SIDSj, Qsj)))))) in
out (ch, (V2, Ysj));
in (ch, (yV3: bitstring));
let keysj = exp (yYui, B) in
let SKSj = H (con (keysj, con (Qsj, con (VIDui, SIDSj)))) in
let V3' = H (con (SKSj, con (VIDui, SIDSj))) in
if V3' = yV3 then event ServerAcSK ();
0).
(*************RC's process************)
let UserReg =
in (sch, (zIDUi: bitstring, zFVIDui: bitstring, zPui: bitstring, zFVPWui: bi
tstring, zFVBIui: bitstring));
let VPWui = xor (zFVPWui, exp (zPui, x)) in
let VBIui = xor (zFVBIui, exp (zPui, x)) in
let VIDui = xor (zFVIDui, exp (zPui, x)) in
let dui = mult (x, HG (VIDui)) in
let Vui = H (con (VPWui, con (VBIui, VIDui))) in
let Rsj = xor (bilinearmap (dui, HG(SIDSj)), xor (VBIui, VPWui)) in
new Tabui: bitstring;
out (sch, (Tabui, Vui));
0.
let ServerReg =
in (sch, (zPsj: bitstring, zSIDSj: bitstring));
let dsj = mult (x, HG (zSIDSj)) in
let Prsj = xor (exp (zPsj, x), dsj) in
out (sch, (Prsj));
0.
let RCAuth =
new zPui: bitstring;
new zFVPWui: bitstring;
let VPWui = xor (zFVPWui, exp (zPui,x)) in
new bui: bitstring;
new t: bitstring;
let Bui = exp (g, bui) in
let Cui = exp (zPui, bui) in
let Tui = H (con (Bui, con (Cui, t))) in
out (ch, (Bui, Cui, t));
0.
let ProcessRC = UserReg | ServerReg | RCAuth.

(*---------main----------*)
process
        (!ProcessUser | !ProcessServer | !ProcessRC)
```

(b) Process

Figure 5: Processes.

validation and key_{ui} and key_{sj} constitute the session key. Therefore, the proposed scheme can resist replay attacks.

4.3.5. User Anonymity. In the key management phase, the user passes the virtual identity $VID_{ui} = H(ID_{U_i} \| r_{ui})$ to the server, and each round of session is protected by a new random number α. Accordingly, the server verifies the user's pseudonym $VID'_{ui} = F_{ui} \oplus H(Y_{ui} \| SID_{S_j} \| Q_{sj})$. Therefore, $\mathcal{A}$ cannot extract the real identity ID_{U_i} of the user. The proposed scheme achieves user anonymity.

5. Performance Analysis

The proposed scheme is evaluated with those of [32–35] in terms of security, computation cost, communication cost, and storage cost. These five protocols all use bilinear pairing operations. Considering the practical application value, we

```
(**********results**********)
1-- RESULT not attacker (SKUi[]) is true.
2-- RESULT not attacker (SKSj[]) is true.
3-- RESULT not attacker (IDUi[]) is true.
4-- RESULT not attacker (SIDSj[]) is true.
5-- RESULT inj-event (UserAuthed) ==> inj-event (UserStarted) is true.
6-- RESULT inj-event (UserAcServer) ==> inj-event (ServerAcUser) is true.
7-- RESULT inj-event (ServerAcSK) ==> inj-event (UserAcServer) is true.
```

Figure 6: Results.

Table 4: Security comparison.

	[32]	[33]	[34]	[35]	Ours
K1	χ [33]	√	√	√	√
K2	√	√	√	√	√
K3	√	√	√	√	√
K4	χ [33]	√	√	√	√
K5	√	√	√	√	√
K6	√	√	√	√	√
K7	χ [33]	√	√	√	√
K8	χ [33]	√	√	√	√

analyzed the consumption of the login and authentication phase for computation cost and communication cost and the consumption of the registration phase for storage cost.

5.1. Security Comparison. In Table 4, we conducted a security evaluation. Let K1, K2, K3, K4, K5, K6, K7, and K8 represent user anonymity, perfect forward security, session key agreement, offline password guessing attacks, insider attacks, preverification, server impersonation attacks, and user impersonation attacks, respectively. Note that preverification means that the user needs to pass the verification of the smart card before communicating with the server. Session key agreement means that the session key needs to be established by two participants. "√" indicates that the protocol can resist attacks. "χ" symbolizes that the protocol cannot resist the attack. Table 4 shows that the scheme in [32] is subject to server impersonation attacks and user impersonation attacks, which is a significant security hazard for the entire protocol. In addition, [32] cannot provide user anonymity and cannot resist offline password guessing attacks. The schemes in [33–35] and our scheme have strong security.

5.2. Computation Cost Comparison. In Table 5, we counted and compared the number of operations and computation time in the schemes of [32–35] and ours. T_{map} denotes the time of bilinear pairing operation, T_{mtp} denotes the time of hash operation from the map to point, T_{ex} represents the time consumption of exponential operation, T_f represents the time consumption of fuzzy extraction function, T_m represents the time consumption of scalar multiplication in G_1, T_a represents the time of point addition in G_1, T_{mg} represents the time of multiplication in G_2, and T_h represents the time consumption of general hash operation. The XOR and join operations were ignored. According to [32], the approximate time consumptions of T_{map}, T_{mtp}, T_{ex}, T_m, T_a, T_{mg}, and T_h are 32.713 ms, 33.582 ms, 2.249 ms, 13.405 ms, 0.081 ms, 0.008 ms, and 0.056 ms for the end-user, respectively. Note that we assume $T_f = T_m$. On the server-side, the approximate time consumptions of T_{map}, T_{mtp}, T_{ex}, T_m, T_a, T_{mg}, and T_h were 5.427 ms, 5.493 ms, 0.339 ms, 2.165 ms, 0.013 ms, 0.001 ms, and 0.007 ms, respectively. The consumption of the proposed scheme on the user-side was slightly higher than that of [33], but on the server-side, it is lower than that of their scheme. In addition, the total computation cost of the proposed scheme was slightly higher than that of [33]. In practical applications, it can cause almost the same online experience for the user. Figure 7 more intuitively shows the comparison of computation cost between our scheme and [32–35].

5.3. Communication and Storage Cost Comparison. We chose $q = 160$ bits and $n = 512$ bits. The output length in G_1 and G_2 is 1024 bits, and the output length for general hash operation and identity is 160 bits. The specific analysis is as follows.

In He et al.'s scheme [32], the transmitted messages are $\{R_{U_i}\}$, $\{y, \alpha_{S_j}\}$, and $\{C_{U_i}\}$, where $\{R_{U_i}, y\}$ belong to G_1 and $\{\alpha_{S_j}, C_{U_i}\}$ belong to F_n^*. The communication cost was 2368 bits. The stored messages are $\{g_{U_i}, \psi_{U_i}, v_{U_i}, b_{U_i}\}$ and $\{D_{S_j}\}$, where $\{g_{U_i}, b_{U_i}, D_{S_j}\}$ belong to G_1 and $\{\psi_{U_i}, v_{U_i}\}$ belong to F_n^*. The storage cost was 3392 bits.

In Li et al.'s scheme [33], the transmitted messages are $\{F_{ui}, k_{ui}, B_{ui}, d_{tui}, t\}$, $\{D_{sj}, k_{sj}\}$, and $\{D_{U_i}\}$, where $\{k_{ui}, B_{ui}, k_{sj}\}$ belong to $G_1$and $\{F_{ui}, d_{tui}, t, D_{sj}, D_{U_i}\}$ belong to F_n^*. The communication cost is 3872 bits. The stored messages are $\{T_{ui}, G_{ui}, V_{ui}, h_1, \theta_{ui}, W_{ui}, H_i(i = 00, 01, 1, \cdots, 7), g_{pub}\}$ and $\{B_{ui}, d_{tui}, t\}$, where $\{H_i(i = 00, 01), g_{pub}, B_{ui}, d_{tui}\}$ belong to G_1 and $\{T_{ui}, G_{ui}, V_{ui}, h_1, \theta_{ui}, W_{ui}, H_i(i = 1, \cdots, 7), t\}$ belong to F_n^*. The storage cost was 7360 bits.

In Chuang and Tseng's scheme [34], the transmitted messages are $\{AID, N_i, ACAKE\}$, $\{N_B, v_B\}$, and $\{v_i\}$, where $\{AID, N_i, ACAKE, N_B\}$ belong to G_1 and $\{v_B, v_i\}$ belong to F_n^*. The communication cost is 4416 bits. The stored messages are $\{ID_i, t, Q_i, MPK_t, HLP_i\}$, $\{ID_j, t, Q_j, MPK_t\}$, and $\{ID_l, t, Q_l, MPK_t, HLP_l\}$, where $\{Q_i, MPK_t, Q_j, MPK_t, Q_l, MPK_t\}$ belong to G_1 and $\{ID_i, t, HLP_i, ID_j, t, ID_l, t, HLP_l\}$ belong to F_n^*. The storage cost was 7424 bits.

Table 5: Computation cost comparison.

	[32]	[33]	[34]	[35]	Ours
User	$2T_m + T_a + T_{ex} + 8T_h \approx 31.837$ ms	$T_f + 2T_{ex} + 10T_h \approx 18.463$ ms	$2T_{map} + 2T_{mtp} + 3T_m + 4T_h \approx 173.029$ ms	$T_{map} + T_{mtp} + 2T_m + 5T_h \approx 93.385$ ms	$T_f + 3T_{ex} + 10T_h \approx 20.712$ ms
Server	$2T_{map} + 4T_{ex} + 2T_m + 5T_h \approx 16.575$ ms	$T_{map} + T_{mtp} + 4T_{ex} + T_m + 6T_h \approx 14.483$ ms	$T_{map} + T_{mtp} + 2T_m + 5T_h \approx 15.285$ ms	$5T_{map} + 8T_{ex} + 6T_m + 2T_h \approx 42.851$ ms	$T_{map} + T_{mtp} + 4T_{ex} + 5T_h \approx 12.311$ ms
Total	48.412 ms	32.946 ms	188.314 ms	136.236 ms	33.023 ms

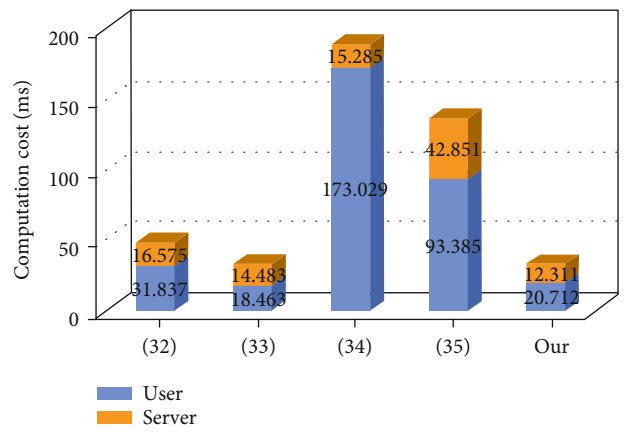

Figure 7: Computation cost comparison.

Table 6: Communication and storage cost comparison.

	Rounds	Communication cost (bits)	Storage cost (bits)
[32]	3	2368	3392
[33]	3	3872	7360
[34]	3	4416	7424
[35]	3	6944	8192
Our	3	3712	2848

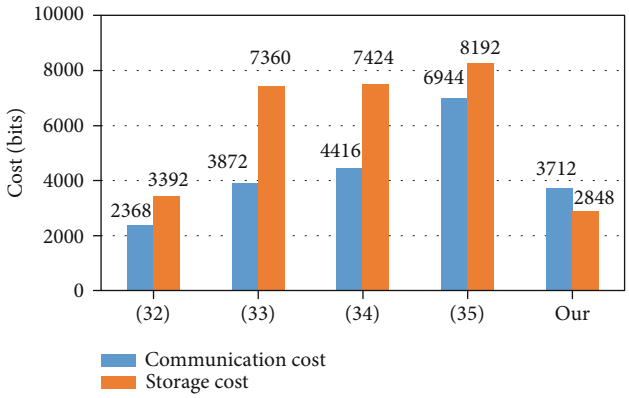

Figure 8: Communication and storage cost comparison.

In Tseng et al.'s scheme [35], the transmitted messages are $\{ID_c, QT_c, QU_c, X, n_c, Sig_c\}$, $\{Y_1, Y_2, Auth_s, n_s\}$, and $\{Auth_c\}$, where $\{X, n_c, Sig_c, Y_1, Y_2, n_s\}$ belong to G_1 and $\{ID_c, QT_c, QU_c, Auth_s Auth_c\}$ belong to F_n^*. The communication cost is 3872 bits. The stored messages are $\{SU_{c,0,1}, SU_{c,0,2}\}$, $\{XU_{c,0,1}, XU_{c,0,2}\}$, $\{SU_{s,0,1}, SU_{s,0,2}\}$, and $\{XU_{s,0,1}, XU_{s,0,2}\}$, where $\{SU_{c,0,1}, SU_{c,0,2}, XU_{c,0,1}, XU_{c,0,2}, SU_{s,0,1}, SU_{s,0,2}, XU_{s,0,1}, XU_{s,0,2}\}$ belong to G_1. The storage cost was 8192 bits.

In the proposed scheme, the transmitted messages are $\{F_{ui}, Y_{ui}, P_{ui}, V_1\}$, $\{V_2, Y_{sj}\}$, and $\{V_2\}$, where $\{Y_{ui}, P_{ui}, Y_{sj}\}$ belong to G_1 and $\{F_{ui}, V_1, V_2, V_3\}$ belong to F_n^*. The communication cost is 3712 bits. The stored messages are $\{Tab_{U_i}, V_{ui}, E_{ui}, \theta_{ui}, P_{ui}\}$ and $\{C_{ui}, t\}$, where $\{Tab_{U_i}, V_{ui}, E_{ui}, \theta_{ui}, t\}$ belong to G_1 and $\{P_{ui}, C_{ui}\}$ belong to F_n^*. The storage cost was 2848 bits.

Table 6 summarizes the communication and storage costs of the five schemes. It can be noticed that the communication cost of the proposed scheme is lower than that of [33–35] but slightly higher than that of [32]. However, [32] is subject to offline identity guessing and impersonation attacks. Furthermore, our scheme has the lowest storage cost. Figure 8 more intuitively shows the comparison of communication and storage costs between our scheme and [32–35].

6. Conclusion

Many researchers have proposed solutions for authentication in the multiserver architecture of the client-server mode, but most of them have some security vulnerabilities. In addition, the development of 5G technology has gradually matured, which can bring users a superfast online experience. Therefore, we proposed a secure key management protocol to protect user anonymity based on the multiserver architecture of the client-server mode in 5G. Through formal and informal analyses, we proved that our scheme has better security. Furthermore, the performance estimate confirms that the proposed scheme has higher advantages than similar schemes. Therefore, the proposed scheme is more suitable for the client-server mode of the multiserver in 5G.

Data Availability

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

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, and V. Stettler, "A formal analysis of 5G authentication," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1383–1396, Toronto, Canada, 2018.

[2] H. Viswanathan and P. E. Mogensen, "Communications in the 6G era," *IEEE Access*, vol. 8, pp. 57063–57074, 2020.

[3] E. K. Wang, X. Liu, C.-M. Chen, S. Kumari, M. Shojafar, and M. S. Hossain, "Voice-transfer attacking on industrial voice control systems in 5G-aided IIoT domain," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 7085–7092, 2020.

[4] P. Wang, C.-M. Chen, S. Kumari, M. Shojafar, R. Tafazolli, and Y.-N. Liu, "HDMA: hybrid D2D message authentication scheme for 5G-enabled VANETs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5071–5080, 2021.

[5] S. Kumari, M. Karuppiah, A. K. Das, X. Li, F. Wu, and N. Kumar, "A secure authentication scheme based on elliptic curve cryptography for IoT and cloud servers," *The Journal of Supercomputing*, vol. 74, no. 12, pp. 6428–6453, 2018.

[6] H. Xiong, Y. Wu, C. Jin, and S. Kumari, "Efficient and privacy-preserving authentication protocol for heterogeneous systems in IIoT," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11713–11724, 2020.

[7] H. Xiong, Y. Zhao, Y. Hou et al., "Heterogeneous signcryption with equality test for IIoT environment," *IEEE Internet of Things Journal*, 2020.

[8] M. A. Khan, I. Ullah, S. Nisar et al., "Multiaccess edge computing empowered flying ad hoc networks with secure deployment using identity-based generalized signcryption," *Mobile Information Systems*, vol. 2020, Article ID 8861947, 15 pages, 2020.

[9] M. Asghar Khan, I. Ullah, A. Alkhalifah et al., "A provable and privacy-preserving authentication scheme for UAV-enabled intelligent transportation systems," *IEEE Transactions on Industrial Informatics*, p. 1, 2021.

[10] B. Ying and A. Nayak, "Lightweight remote user authentication protocol for multi-server 5G networks using self-certified public key cryptography," *Journal of Network and Computer Applications*, vol. 131, pp. 66–74, 2019.

[11] I.-U. Haq, J. Wang, and Y. Zhu, "Secure two-factor lightweight authentication protocol using self-certified public key cryptography for multi-server 5G networks," *Journal of Network and Computer Applications*, vol. 161, article 102660, 2020.

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

[13] X. Li, J. Liao, J. Zhang, J. Niu, and S. Kumari, "A secure remote user mutual authentication scheme using smart cards," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 89–92, Beijing, China, 2014.

[14] M. Karuppiah, "Remote user authentication scheme using smart card: a review," *International Journal of Internet Protocol Technology*, vol. 9, no. 2/3, pp. 107–120, 2016.

[15] T.-Y. Wu, Z. Lee, M. S. Obaidat, S. Kumari, S. Kumar, and C.-M. Chen, "An authenticated key exchange protocol for multi-server architecture in 5G networks," *IEEE Access*, vol. 8, pp. 28096–28108, 2020.

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

[17] Xiang Cao and Sheng Zhong, "Breaking a remote user authentication scheme for multi-server architecture," *IEEE Communications Letters*, vol. 10, no. 8, pp. 580-581, 2006.

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

[19] C.-C. Chang and J.-S. Lee, "An efficient and secure multi-server password authentication scheme using smart cards," in *2004 international conference on cyberworlds*, pp. 417–422, Tokyo, Japan, 2004.

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

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

[22] Y. Luo, W.-M. Zheng, and Y.-C. Chen, "An anonymous authentication and key exchange protocol in smart grid," *Journal of Network Intelligence*, vol. 6, no. 2, pp. 206–215, 2021.

[23] C.-T. Li and M.-S. Hwang, "An efficient biometrics-based remote user authentication scheme using smart cards," *Journal of Network and Computer Applications*, vol. 33, no. 1, pp. 1–5, 2010.

[24] X. Li, J.-W. Niu, J. Ma, W.-D. Wang, and C.-L. Liu, "Cryptanalysis and improvement of a biometrics-based remote user authentication scheme using smart cards," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 73–79, 2011.

[25] E.-J. Yoon and K.-Y. Yoo, "Robust biometrics-based multi-server authentication with key agreement scheme for smart cards on elliptic curve cryptosystem," *The Journal of Supercomputing*, vol. 63, no. 1, pp. 235–255, 2013.

[26] Y.-P. Liao and C.-M. Hsiao, "A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 886–900, 2013.

[27] H. Kim, W. Jeon, K. Lee, Y. Lee, and D. Won, "Cryptanalysis and improvement of a biometrics-based multi-server authentication with key agreement scheme," in *Computational Science and Its Applications – ICCSA 2012*, pp. 391–406, Springer, 2012.

[28] W.-B. Hsieh and J.-S. Leu, "An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures," *The Journal of Supercomputing*, vol. 70, no. 1, pp. 133–148, 2014.

[29] M.-C. Chuang and M. C. Chen, "An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1411–1418, 2014.

[30] D. Mishra, A. K. Das, and S. Mukhopadhyay, "A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8129–8143, 2014.

[31] Y. Lu, L. Li, H. Peng, and Y. Yang, "A biometrics and smart cards-based authentication scheme for multi-server environments," *Security and Communication Networks*, vol. 8, no. 17, 3228 pages, 2015.

[32] D. He, S. Zeadally, N. Kumar, and W. Wu, "Efficient and anonymous mobile user authentication protocol using self-certified public key cryptography for multi-server architectures," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 2052–2064, 2016.

[33] W. Li, L. Xuelian, J. Gao, and H. Y. Wang, "Design of secure authenticated key management protocol for cloud computing environments," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2019.

[34] Y. H. Chuang and Y. M. Tseng, "CAKE: compatible authentication and key exchange protocol for a smart city in 5G networks," *Symmetry*, vol. 13, no. 4, p. 698, 2021.

[35] Y. M. Tseng, J. L. Chen, and S. S. Huang, "A lightweight leakage-resilient identity-based mutual authentication and key exchange protocol for resource-limited devices," *Computer Networks*, vol. 196, article 108246, 2021.

[36] Y.-S. Jheng, R. Tso, C.-M. Chen, and M.-E. Wu, "Password-based authenticated key exchange from lattices for client/server model," in *Advances in Computer Science and Ubiquitous Computing*, pp. 315–319, Springer, 2017.

[37] T.-Y. Wu, L. Yang, Z. Lee, C.-M. Chen, J.-S. Pan, and S. Islam, "Improved ECC-based three-factor multiserver authentication scheme," *Security and Communication Networks*, vol. 2021, Article ID 6627956, 14 pages, 2021.

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

[39] V. Odelu, A. K. Das, and A. Goswami, "A secure biometrics-based multi-server authentication protocol using smart cards," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1953–1966, 2015.

[40] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[41] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," *ACM SIGPLAN Notices*, vol. 36, no. 3, pp. 104–115, 2001.

[42] B. Blanchet, "A computationally sound mechanized prover for security protocols," *IEEE Transactions on Dependable and Secure Computing*, vol. 5, no. 4, pp. 193–207, 2008.

[43] T.-Y. Wu, Y.-Q. Lee, C.-M. Chen, Y. Tian, and N. A. Al-Nabhan, "An enhanced pairing-based authentication scheme for smart grid communications," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2021.