Hindawi
Security and Communication Networks
Volume 2022, Article ID 3658402, 20 pages
https://doi.org/10.1155/2022/3658402

Research Article

Secure Multifactor Authentication and Access Control Mechanism for Electronic Bill Service in a 5G Cloud-Fog Hybrid Architecture

Zhenyang Guo, Yueyu Zhang, Jin Cao, Xiongpeng Ren, Xingwen Zhao, and Hui Li

School of Cyber Engineering, State Key Laboratory of Integrated Service Network, Xidian University, Xi'an, China

Correspondence should be addressed to Yueyu Zhang; yyzhang@xidian.edu.cn

Received 12 January 2022; Revised 16 February 2022; Accepted 11 March 2022; Published 12 May 2022

Academic Editor: AnMin Fu

The electronic bill service was greatly encouraged by electronic commerce and paperless bills. However, the massive authentication and authorization access requests from users in the electronic bill services cannot be efficiently processed by a conventional client/server-based scheme during the 5G era. This paper proposes a secure multifactor authentication and access control scheme tailored for electronic bill service in a 5G cloud-fog hybrid architecture. The proposed scheme can realize many security properties, including mutual authentication, privacy protection, batch authentication, authorization and revocation of authority, and resistance to multiple protocol attacks. Moreover, the scheme has the desired performance compared to similar schemes. Finally, we used BAN logic and Scyther to analyze and verify the proposed protocol, and the result shows that the proposed scheme is stable and can achieve the previous security goals.

1. Introduction

1.1. Background and Motivation. With the continuous enlargement of the e-commerce and the mobile communication technology such as the fifth generation (5G), electronic bill have also become a requisite part of people's quotidian lives. At the same time, some countries have issued regulations to promote the application of paperless receipt systems. Existing paper invoices have problems such as low efficiency, high cost, high management risk, poor user experience, difficulty to keep, and are not conducive to resource protection. The implementation of electronic bill will help further reduce invoice costs and invoice claims. Electronic bill, convenient for the storage and retrieval of financial and taxation departments, can promote environmental protection and reduce the time for users to issue invoices [1, 2]. Additionally, benefited from the low latency and wide coverage of 5G, 5G can support more and more new type applications and provide better services for them. For electronic bill service, 5G cloud-fog hybrid architecture, as a multiserver architecture can make it more reliable and scalable [3–5].

However, electronic bill services are still facing many new security and performance challenges.

First, the electronic bill service system is constructed by the state taxation department. To promote the evolution of paper bill to electronic bill, the national tax authorities should complete the functions of the electronic bill service system promptly and improve the reliability, availability, and security of the electronic bill service system to meet the growing needs of citizens for electronic bill.

Second, due to the openness of the Internet, when users access the electronic bill service system, the data transmitted by the user may be intercepted, tampered, replayed, and modified by attackers [6]. Therefore, the electronic bill service system must establish an efficient and secure authentication mechanism to prevent illegal users from malicious attacks and protect the system security. Moreover, in the era of 5G, the network structure comes into the cloud-fog hybrid structure. The calculation requirements of the cloud server can be

reduced by the fog server. In addition, the electronic bill service is a multiserver, cross-domain network service [6, 7], and thus, the electronic bill service also needs to consider its network environment when designing an authentication mechanism.

Finally, in order to prevent malicious users, devices, and service systems [8] from illegally accessing and calling electronic bill services to obtain illegal benefits, the electronic bill service system should perform authority control and authority identification for different users, devices, and service systems.

In summary, studying a secure authentication and access control scheme for electronic bill service in the 5G cloud-fog hybrid architecture is required.

1.2. Threat Model. This paper mainly considers an authentication scenario based on proxy signature with four entities: user, proxy signer, original signer, and verifier. We define that the electronic bill device is the proxy signer, the electronic bill authentication server is the original signer, and the electronic business server is the verifier. For convenience, we abbreviate the electronic bill service device as the Device, the electronic bill authentication server as the *Service*, and the electronic business server as the Server.

First, the user registers to the Service. The Device also applies for proxy signature authority from the Service. After registration, the user chooses a Device to bind. Once the user binds to the Device, the Device will obtain the user's authentication vector. Before the user accesses the Server, the user needs to sign in and get the TOKEN from the Device. The Server once receives the TOKEN from the user, and it can verify the proxy signature by the TOKEN.

We define that Device constitutes the fog layer, and Service and Server work in the cloud layer. Since users can communicate with fog nodes through a local area network (LAN) and with cloud nodes through wide area network (WAN) [9, 10], the Dolev–Yao model [11] in this paper is suitable.

(i) Adversary $\mathscr{A}$ can obtain and attempt to modify, intercept, or delete information on the channel.

(ii) Adversary $\mathscr{A}$ can know how the protocol is transmitted and obtain public parameters.

(iii) Adversary $\mathscr{A}$ cannot obtain the user password, biometric feature, and user private key simultaneously.

1.3. Proposed Approach. In this paper, we propose a multifactor authentication scheme, which effectively overcomes the limitations of previous methods. We take advantage of the proxy signature mechanism, which effectively disperses the authentication pressure of the authentication center, reduces the cost of authentication signaling, and protects the privacy of the user's authority.

To verify the security of our protocol, we used informal analysis and formal verification including Burrows–Abadi–Needham (BAN) logic and the official Scyther simulation. Moreover, we compared the previous scheme [12–15] with ours in several aspects, such as calculation cost, storage cost, and communication cost.

1.4. Key Contributions and Results. First, we design a new cloud-fog hybrid electronic bill services structure in 5G network shown as Figure 1. Subsequently, we propose a revocable anonymous proxy signature scheme in which the Service allows Devices as a proxy signer to give the user's TOKEN as a proxy signature. We also propose a multifactor authentication for the user. Moreover, our scheme provides a multifactor update phase and proxy signer privacy revocation phase.

The contributions are described in detail as follows: (1) A new electronic bill service structure is designed based on the cloud-fog hybrid in 5G network. (2) A multifactor user identity authentication scheme is proposed based on biometrics, passwords, public and private keys, and one-time passwords. (3) By adopting the proxy signature mechanism, the computational overhead of our scheme is lower than that of the similar schemes. (4) The proposed scheme can realize the privacy protection of user biometrics, passwords, public and private keys, and one-time passwords. (5) The electronic bill business server can deal with TOKEN in batch processing. (6) The BAN logic and Scyther tool have been employed to analyze and verify the security of our scheme.

The chapters of this article consist as follows: we investigate the related work in Section 2. The system model and secure goal of our scheme are presented in Section 3. Section 4 offers the process of our scheme. The security and performance analysis is given in Sections 5 and 6, respectively. Finally, we summarize the paper in Section 7.

2. Related Work

Recently, there has been an increasing number of researches on electronic bill service systems in academia. Still, less studies on the authentication and authority control schemes of electronic bill service systems since the electronic bill service system has the characteristics of multiple servers in parallel, multiple data domains, coexisting with management domains. The electronic bill system is a typical multiserver architecture system. This section gives some related multiserver authentication (MSA) and cloud-fog hybrid architecture protocols proposed by academic researchers.

In 2014, by combining the biometrics and elliptic curve cryptosystem (ECC), He et al. [16] gave the first MSA scheme. In the same year, the scheme in [16] indicated that it is vulnerable to some attacks such as simulation attacks by Odelu et al. [17]. The scheme in [17] also presented a new protocol based on biometric, smart cards, and ECC. The schemes in [18, 19] also proposed a new authentication and key agreement (AKA) protocol. Moreover, the scheme in [19] pointed out that the scheme in [18] is vulnerable to smart card theft attacks.

In 2015, for the telemedicine environment, Barman et al. [20] successfully solved the multiserver identity authentication problem by proposing a user authentication scheme

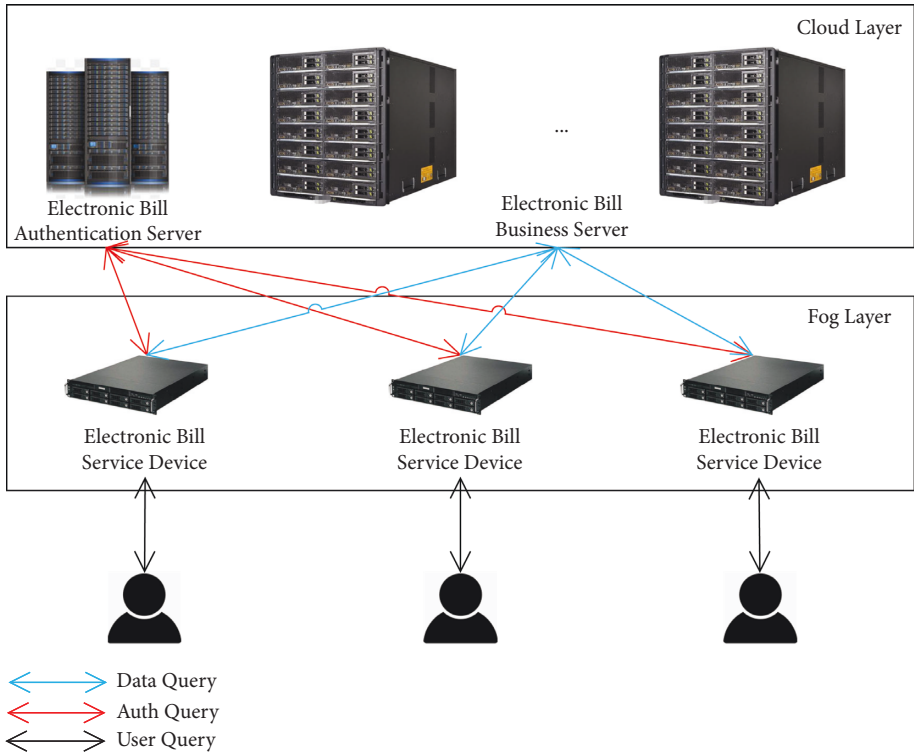

Figure 1: Electronic bill service security authentication and authority control system.

with low computational overhead. However, this solution incurs several security issues including poor scalability, vulnerability to privilege attacks, and lack of access control. Lin et al. [21] further analyzed the scheme proposed by the scheme in [18] and pointed out that it is prone to server spoofing attacks and cannot protect user anonymity and session keys. Chuang and Chen enhanced the scheme in [18] by proposing a new AKA scheme of multiserver with robust privacy protection.

In 2016, Moon et al. [22] proposed a three-factor authentication scheme. In [22], users can sign in to the service at the multiserver scenario by their smart cards, password, and biometrics. This scheme makes up for the problems of the scheme proposed by [23] that cannot resist camouflage attacks and password guessing attacks. However, it has still problems such as lack of forward security and denial of service attacks.

In 2017, Irshad et al. [24] analyzed the schemes in [25–27] and pointed out that the abovementioned schemes are easy to incur different forms of attacks. Subsequently, an improved protocol is proposed on the abovementioned work. Reddy et al. [28] also analyzed the scheme in [27] and indicated that the scheme in [27] is vulnerable to multiple attacks.

In 2018, aiming at the mobile cloud computing environment, Chatterjee et al. [29] proposed an identity authentication protocol that has a small computational and communication overhead. The registry does not take part in the login, identity verification, and key establishment stages in [29]. However, this solution has the problems that it cannot resist counterfeiting attacks, does not have user untraceability, and cannot guarantee multifactor security.

In 2019, Qiao et al. [30] proposed an extended AKA scheme for a multiserver environment with robust anonymity in a telecare medicine information system (TMIS) and improved about 20% efficiency compared with the schemes in [31, 32]. Limbasiya and Sahay [33] studied the remote authentication scheme for multiserver systems and presented a protocol. The scheme in [33] improves efficiency and reduces the communication overhead comparing with the previous solution. Ying and Nayak [4] imported 5G network to the multiserver system and gave a lightweight protocol using self-certified public key cryptography (SCPKC), the performance analysis shows Ying's protocol can reduce the communication and computational overhead.

From 2020, Wu et al. [3] proposed an enhanced protocol for the 5G multiserver network using only hash functions, and the result of formal security analysis shows [3] can resist various attacks such as privileged insider attacks. Wang and Zhu [5] pointed out that the scheme in [4] is vulnerable to multiple attacks such as identity guessing and password impersonation attacks. Reference [5] also proposed an improved protocol based on SCPKC. For the healthcare system, Limbasiya et al. [34] proposed an AKA scheme. Roy and Bhattacharya [35] designed a lightweight MAKA protocol based on the group key. Shamshad et al. [36] proposed an identity-based authentication protocol for TMIS based on physically unclonable function (PUF).

Likewise, there are also some works on the authentication schemes for the cloud-fog hybrid architecture.

In fog-cloud based vehicular ad hoc networks scenes, Yang et al. [37] proposed a certificateless aggregate

signcryption scheme (CASS) that realized the privacy protection of sender. Based on the proposed CASS, Yang et al. designed a new privacy-preserving aggregation authentication scheme (PPAAS). This scheme effectively reduces the bandwidth overhead by allowing the vehicle to preload the public keys of all RSUs on the driving route, and the RSUs aggregate the received ciphertext. However, it incurs a lot of computational cost due to the use of the bilinear map, which is not suitable for vehicles with limited computing resources. In TMIS scenes, Liu et al. [15] constructed a novel privacy-preserving mutual authentication (NPMA) scheme based on certificateless cryptography and shared secrets. This scheme can provide double anonyms for patients and edge-severs to protect the privacy during the authentication and trace malicious patients or edge-severs.

In fog-cloud based Internet of Things (IoT) scenes, Ali and Sridevi [38] proposed an attributes-based authentication mechanism for IoT devices. This scheme assumes that the fog broker is a high-performance central fog node responsible for the key and credential generation (KCG). Also, the IoT device is defined as an object with seven attributes. The KCG can use unique identifiers and features such as user names and passwords to generate the device's credentials. However, there is no attribute update process in the scheme in [38]. Additionally, the secret is shared by the KGC to the IoT device and the specific fog server using the public key, while it is guaranteed that only the particular fog server can authenticate this device. However, it is not resistant to insider attacks. Loffi et al. [39] designed a mutual authentication protocol with a tolerance time rate (ttr) to defend man-in-the-middle attacks and imitation attacks in the IoT. In this scheme, both the two parties maintain a total time counter (ctt) and use the sum of the processing time (tp^A), network delay (tr^B), and ttr to infer whether the currently received datagram has been attacked, that is, if $ctt \leq tp^A + tr^B + trr$ holds, it means that the message is less likely to be tampered or intercepted. Otherwise, the receiver believes that the message may be tampered with or intercepted. However, there are the following problems in the scheme in [39]. On the one hand, for different devices, the server needs to save different ttr, which is lack of dynamic update process of ttr. On the other hand, the process of getting ttr is not friendly to energy-constrained IoT devices.

Based on the analysis in the abovementioned work, there are few existing authentication models and security solutions for electronic bill service. Therefore, it is key point to provide secure and robust authentication for an electronic bill architecture.

3. System Model and Goal

3.1. System Model. As shown in Figure 1, the electronic bill security authentication and authority control system mainly includes the following four parts: users, electronic bill service device Device, electronic bill authentication server Service, and electronic bill business server Server. In our scheme, we suppose the electronic bill authentication server and electronic bill business server are located in the cloud, the electronic bill service device is situated in the fog, and user can access electronic bill service through the 5G network.

(i) Electronic Bill Authentication Server: It is a cloud server and is mainly responsible for authenticating the user. If the authentication is successful, it issues an authentication token to the user. If the authentication fails, the corresponding prompt information are directly returned to the user. The electronic bill authentication service, as the original signer, can issue his signature right to the electronic bill service device. The electronic bill authentication service also has the function of a key distribution center.

(ii) Electronic Bill Business Server: It is a cloud server and is a provider of electronic bill services. After completing user authentication, the user can send a query with a TOKEN to the electronic bill business server. The electronic bill business server can verify the TOKEN. It is the verifier and can effectively distinguish between the original and proxy signatures.

(iii) Electronic Bill Service Device: It is a fog server and is mainly responsible for authenticating the users who bind with it. It is a subsystem of the electronic bill authentication service and can legally generate the proxy signature of the electronic bill authentication service.

(iv) User: The user is the issuer and receiver who use the electronic bill service. Before using the electronic bill service, the user must download an electronic bill client. Each legal user has a unique legal user ID and a matching public and private key pair.

Before users try to access the Server, they must sign in and get the TOKEN from the Device bound to them as shown in Figure 2. Subsequently, the user sends the access request with the TOKEN to Server, specifically including the following two steps. (1) The user needs to choose a Device from the Service and then assist the Device to complete the service registration and device binding phase. (2) When the user accesses the *Server*, he needs to provide the Device with multifactor information required for authentication: biometrics, a time-based one-time password (TOTP) code, password, and authentication information combined with the user's private key. The Device calculates and compares the authentication information provided by the user with the corresponding authority value, then signs the TOKEN by proxy signature and issues the TOKEN to the user. The user initiates a data access request with a TOKEN. The Server verifies the token in the data access request and completes the user's request.

3.2. Protocol Architecture. The proposed scheme includes 11 polynomial time algorithms shown as follows.

(1) System Initialization Phase: By specifying two prime numbers, p and q, Service generates its ID_M, a key pair (x_M, y_M), where x_M is the private key of

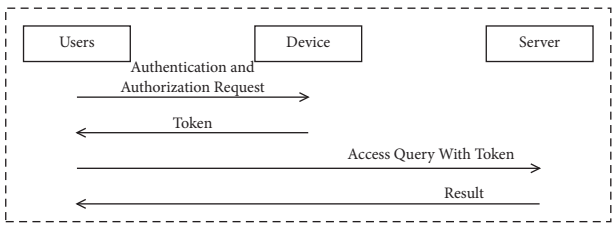

Figure 2: Authentication and authentication access process.

the Service, and y_M is the public key. The Service also generates required general parameters, which all parties use to the agreement, and ID_M and y_M are included in these public parameters.

(2) User Registration: The user A registers with the service M through a secure channel.

(3) Service Registration and Binding Phase: The user first chooses a device, like the "nearest" fog server as the target Device$D$. The user provides authentication information to the Device$D$ and completes the service registration and the binding of the user and the device D. If this stage is successful, the user can initiate an authentication request to the Device$D$ instead of requesting Service.

(4) Proxy Signature Key Agreement Phase: The Device$D$ applies to the *Service* for proxy signature authority. The Service responds to the Device$D$ in a polynomial time and signs its proxy signature warrant. Finally, the Device$D$ generates a legal secret key pair (x_p, y_p) where x_p is the D's proxy signature private key and y_p is the proxy signature public key.

(5) User Authentication and Proxy Signature Generation Phase: After the Device$D$ completes the previous two stages, it can obtain the authentication vector of the user bound to it and the proxy signature authority. In this stage, Device$D$ can use the above information to complete the user's authentication and issue the TOKEN.

(6) Proxy Signature Verification and Authentication Phase: After the user is authenticated, he or she can access the Service. The Service verifies the TOKEN carried in the user request and determines user permissions. If all the abovementioned criteria pass, the user can access the Service. Otherwise, the Service rejects the user's request.

(7) Proxy Signature Batch Verification Phase: The Server often handles massive user requests. In the 5G cloud-fog hybrid environment, the Server can perform batch verification to deal with massive user requests received at the same time. If the batch verification fails, the Server can use the binary search to find the wrong signature.

(8) Binding Release Phase: The user can unbind with the Device$D$ bound to it for personal reasons. In the binding release phase, the user only needs to interact with the Device$D$ and does not interact with the Service, reducing the communication overhead.

(9) Proxy Signer Privacy Revocation Phase: When a Server finds that the proxy signer h_D is abnormal when verifying the TOKEN, it can obtain the real ID of the proxy signer h_D from the Service$M$.

(10) Proxy Signature Authority Revocation Phase: Under normal circumstances, after the time limit described in the m_w of the Device$D$ expires, the proxy signature authority of Device$D$ is automatically revoked. For Device$D$ that abuses its power, the Service$M$ maintains a proxy signature revocation list. The proxy signature revocation list holds part of the device information, the timestamp T when the device was added to the list, and the urgency level G of the revocation event. For different urgency levels G, we give different handling methods. At the same time, to reduce system overhead, the data put into the proxy signature revocation list is automatically removed by the system when the corresponding proxy warrant expires, thereby avoiding unlimited expansion of the list.

(11) User Multifactor Update Phase: In our scheme, the user can update the authentication factor. The user needs to input the original and updated authentication factors simultaneously to update the user authentication information on the Device$D$ and the Service$M$.

In addition, what needs to be explained is that the user assists the Device to complete the service registration and binding phase. The Service safely delivers the user's multifactor authentication information to the Device and generates a service authentication code. The Device automatically completes the agreement of the proxy signature right and the proxy signature key of the Service.

The user needs to perform authentication and authorization before attempting to access other services of the electronic bill service. After the user initiates an authentication request to the Device, the Device completes user authentication according to the multifactor authentication information input by the user. It then issues a TOKEN with a proxy signature. After adding the TOKEN to the data request, the user can access other services of the electronic bill service. After the Server receives the TOKEN, it can perform signature verification or batch verification. Suppose the Server discovers that a proxy signer has committed multiple violations. The Server can ask the Service to revoke the privacy of the proxy signer. If the circumstances are severe, the Service can revoke the proxy signature authority of the corresponding proxy signer in advance. When the Device is no longer needed, the user can unbind with the Device. The user can update the multiple factors required for authentication.

The timestamps involved in this scheme are all UNIX timestamps. The protocol continues if the received

timestamp is less than 5s earlier than the current timestamp. Otherwise, the protocol terminates.

3.3. Secure Goal. A secure protocol must satisfy the following goals.

(i) Mutual Authentication: Participants of the authentication protocol can authenticate other participants to ensure that the protocol participants are honest and credible.

(ii) Data Confidentiality and Integrity: Unauthorized entities cannot crack the message, and once the message is tampered, deleted, or replaced, it can be noticed by the recipient.

(iii) Resist Several Types of Attacks: The protocol must be resistant to some attacks such as reply attacks, mobile device loss attack, offline password guessing attack, privileged insider attack, and so on.

Furthermore, a proxy-signed scheme must meet the following security goals defined in [40–43].

(i) Proxy Privacy: No one can judge who signs the proxy signature from only the proxy signature.

(ii) Strong Unforgeability: A legal proxy signature can only be generated by a legal proxy signer.

(iii) Proxy signer's deviation: The proxy signer can only create a proxy signature, not an original signature [44].

(iv) Verifiability: A valid proxy signature can pass the verification equation.

(v) Strong identifiability: Proxy signatures and general signatures can be distinguished easily.

(vi) Strong Recognizability: Any verifier can distinguish who signs the proxy signature by a legal proxy signer.

(vii) Strong Non-repudiation: Once a legal proxy signature is generated, the proxy signer denies that its signature is prohibited.

(viii) Anti-abuse: Proxy signers can only exercise their powers within the scope specified by the proxy signature warrants and cannot use them beyond their authority.

3.4. Preliminary. Discrete Logarithm Problem. Denote its group operation by multiplication for any finite cyclic group G, and its generator is g. Let a be an element of G, and it is not easy to solve an x that satisfies $x = \log_g a$.

(1) The Computational Diffie–Hellman Problem (CDH): Given the finite cyclic group G, the generator g of G, it is challenging to find $g^{uv} \in G$ when $g^u \in G$ and $g^v \in G$ are known.

(2) The Decisional Diffie–Hellman Problem (DDH): Given the finite cyclic group G, the generator g of G, it is not easy to judge whether $g^w = g^{uv}$ holds when $g^u \in G$ and $g^v \in G$ are known.

Fuzzy extractors have the ability of feature extraction and secret recovery of biological features with specific probability distribution characteristics, such as fingerprints. As defined in [45], a quintuple of $(\mathcal{M}, m, \ell, t, \varepsilon)$ can represent a fuzzy extractor where $\mathcal{M}$ represents a set. Fuzzy extractors include a pair of procedures (generate (GEN) and reproduce (REP)).

(1) GEN: $\mathcal{M} \xrightarrow{R} \{0,1\}^\ell \times \{0,1\}^*$,.

$$\mathrm{GEN}(b) = (B, C). \tag{1}$$

For any low-entropy string b, the process GEN output is a pair of strings (B, C). B is called a characteristic string, and C is an auxiliary string.

(2) REP: $\mathcal{M} \times \{0,1\}^* \xrightarrow{D} \{0,1\}^\ell$

$$\mathrm{REP}(b', C) = B \text{if} \mathrm{dis}(b, b') \le t. \tag{2}$$

For each $b, b' \in \mathcal{M}$, if (B, C) are generated by $\mathrm{GEN}(b)$ and $\mathrm{dis}(b, b') \le t$, the fuzzy extractor can recover the characteristic string B through the process $\mathrm{REP}(b', C)$. The function $\mathrm{dis}(b, b')$ is similar to the function that calculates the Hamming distance, and its output represents a measure of the difference between two variables.

4. Proposed Scheme

In this section, we introduce the proposed scheme in detail.

4.1. System Initialization Phase. Given large prime number p, choose a prime number q which satisfies $q|p-1$. Choose a generator g with order q on Z_p^*. The *Service M* selects a key pair (x_M, y_M), a random number γ, where x_M is M's private key, y_M is M's public key satisfying $y_M = g^{x_M} mod p$. γ is a secret value used for M to calculate the authentication vector of the member. M publishes $\{ID_M, p, q, g, y_M\}$ and saves γ securely.

For each device D in this system, the *Service M* distributes ID_D and R_D, and device D will use the public parameters to generate a key pair (x_D, y_D). D publishes ID_D, y_D and saves R_D securely.

For each user A, the *Service M* distributes ID_A, a security device with a public and private key pair (x_A, y_A) and a shared secret R_A. A publishes ID_A, y_A and saves R_A securely. The notations used in our proposed scheme are shown in Table 1.

4.2. User Registration. The user A needs to apply his key device to the system first, which should be performed offline, so we do not discuss it here. User A requires to provide identity ID_A and authentication factors such as password pw to send to the service during the signup process. Once the service M receives user information, it checks whether the identity ID_A exists. If it exists, the registrant needs to be warned. Otherwise, the service M will complete the user registration process after verifying the received message. The specific steps are shown in Figure 3.

Table 1: Notations.

Notation	Description
p, q	Large prime and $q \mid p-1$
g	Generator of order q on Z_p^*
m_w	Proxy signature warrant
σ	Digital signature
(x_M, y_M)	Entity M's public and private key pair and satisfy $y_M = g^{x_M} mod p$
$h()$	Hash function
γ	A secret value used for M to calculate the authentication vector of the member
ID_M	Entity M's ID
BIO	User's biometric information
R_D	A secret key shared between M and D
k	A random number, $k \in_R Z_p^*$
T	A timestamp
AV_x^y	An authentication vector sent by entity X to entity Y
pw	User's password
opt	TOTP code, opt = $TOTP(R_X, T)$
AM_M^A	Message authentication code between entity M and A
S_M^D	Secret shared between entities M and D
GEN(BIO)	A process on the fuzzy extractor that can generate characteristic string B and auxiliary string C
REP(BIO′, C)	A process on the fuzzy extractor that can restore the characteristic string B
TOTP(R, T)	Return a time-based one-time password
(ENC/DES)	Symmetric encryption and decryption function

4.3. Service Registration and Binding Phase. At the current phase, the user A needs to complete the binding with the device. The device D requires to assist the user A in completing this process to obtain key information (R_A, CM_A, SCR_M) for user authentication. The user A first needs to provide his identity ID_A and authentication factors such as password and send the calculated value to the device. The device D then needs to use its own private key x_D and shared secret R_D with the service to safely assist the user in this process. The specific steps are shown in Algorithm 1. Method freshRandomNumber() returns a random number and Method checkTimestamps (T) judges whether the input timestamp T is within the validity period, if so, it returns true; otherwise, it returns false.

4.4. Proxy Signature Key Agreement Phase. After the device D completes the binding with the user A, it needs to apply for the proxy signature authority of the service M, so as to have the ability to generate a legal TOKEN for the user instead of the service M. During this process, the device D will generate a proxy signature warrant m_w, and the service M will check whether m_w is compliant. If the m_w is compliant, the service M will compute a signature σ for it. Finally, the proxy signing key pair for device D is (x_p, y_p). The detailed steps of this stage are shown in Algorithm 2.

4.5. User Authentication and Proxy Signature Generation Phase. To complete user authentication, the user needs to enter ID_A and a series of authentication factors such as password pw, private key x_A. The user equipment calculates the above factors and delivers it to the device D, and the device D uses its stored authentication information (R_A, CM_A, SCR_M) to authenticate the user by $g^{CM_A+OPT} = SCR_M \cdot r_c^{HM} \cdot y_A^{h(TOTP(R_A,T_a))} \bmod p$. If the authentication is successful, a TOKEN signed by the proxy signature private key x_p will be issued, and the device D and the user A will successfully negotiate a session key K_S. Otherwise, the device D will prompt the user authentication failure. The detailed steps are shown in Algorithm 3.

4.6. Proxy Signature Verification and Authentication Phase. Once the user A receives the token, the user sends a query to server S with a TOKEN. The server S should use the following steps to perform the authentication and token verification on user's requests.

(1) Check the validity of m_w: Server S checks the validity of m_w by the following steps.
   (a) Check the validity of m_w by $g^{\sigma} = y_M \cdot r_g^{h(m_w \| r_g \| h_D \| T_e)} \bmod p$
   (b) If passed, server S checks the validity of the r_g by the following query.
      (i) Is the r_g in the revocation list?
      (ii) If the r_g is in the revocation list, is the T_g earlier than the ΔT?
      (iii) If the r_g is in the revocation list, is the urgency of revocation "Serious"?

   The valid r_g must meet the condition "do not exist in the revocation list *or* (the T_g is earlier than ΔT *and* the urgency of revocation is not "Serious")".

(2) Check the validity of the σ_m: Server S checks the validity of σ_m by the following steps.
   (a) Combine ID_A, $h(AAL_A)$, T_a, r_a, r_u, and T_g into msg.
   (b) Verify the σ_m by $g^{\sigma_m} = y_p \cdot r_u^{h(msg)}$.

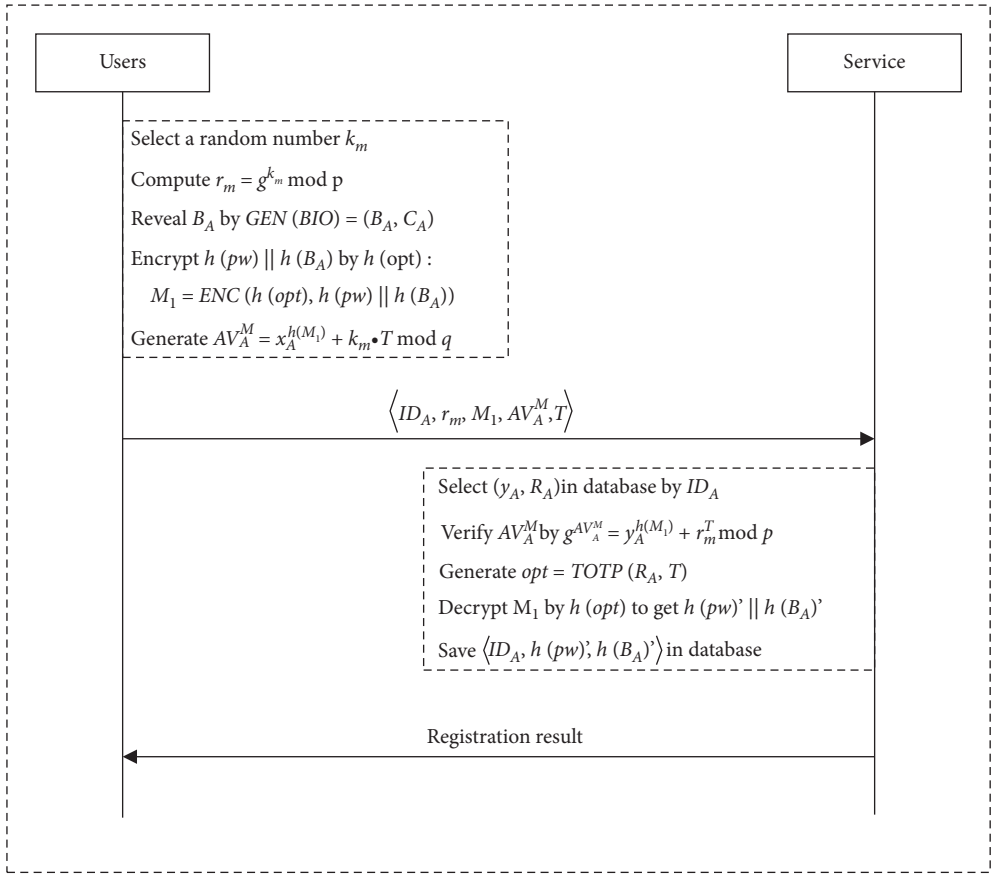

Figure 3: User registration phase.

Step 1: User $A \longrightarrow$ Device D
Input: User A enters ID_A, pw, BIO, opt
(1) $k_l \longleftarrow$ freshRandomNumber()
(2) $r_l \longleftarrow g^{k_l} \bmod p$
(3) $(B_A, C_A) \longleftarrow$ GEN(BIO)
(4) HPW $\longleftarrow h(h(pw) \| h(B_A))$
(5) $AV_A^M \longleftarrow x_A \cdot HPW + k_l \cdot h(T_l \| h(\text{opt})) \bmod q$
Output: Send $\langle ID_A, r_l, T_l, AV_A^M \rangle$ to device D
Step 2: Device $D \longrightarrow$ Service M
Input: Device D's private key x_D and message from user A
(1) $k_a \longleftarrow$ freshRandomNumber
(2) $r_a \longleftarrow g^{k_a} \bmod p$
(3) $AV_D^M \longleftarrow x_D \cdot h(AV_A^M) + k_a \cdot T_a \bmod q$
Output: Send $\langle ID_D, ID_A, AV_D^M, AV_A^M, r_l, r_a, T_l, T_a \rangle$ to service M
Step 3: Service $M \longrightarrow$ Device D
Input: Service M's private key x_M, shared informations R_A, R_D, user information $h(pw)$ and the message from device D
(1) if checkTime stamps (T_l) and checkTimestamps (T_a) then
(2) HPW$' \longleftarrow h(h(pw)' \| h(B_A)')$
(3) HOPT$' \longleftarrow h(\text{TOT}P(R_A, T_l))$
(4) if $g^{AV_A^M} == y_A^{HPW'} \cdot r_l^{h(T_l \| \text{HOPT}')} \bmod p \,\&\&\, g^{AV_D^M} == y_D^{h(AV_A^M)'} \cdot r_a^{T_a} \bmod p$ then
(5) $k_c \longleftarrow$ freshRandomNumber
(6) $k_d \longleftarrow$ freshRandomNumber
(7) $k_e \longleftarrow$ freshRandomNumber
(8) $M_A \longleftarrow h(ID_A \| ID_D \| ID_M \| \sigma)$
(9) $\text{CR}_A \longleftarrow \text{ENC}(R_D, R_A)$
(10) $\text{CHM}_A \longleftarrow \text{ENC}(\text{HPW}', M_A)$

Algorithm 1: Continued.

(11) $r_c \longleftarrow g^{k_c} mod p$
(12) $r_e \longleftarrow g^{k_e} mod p$
(13) $\mathrm{CM}_A \longleftarrow x_M \cdot R_D + k_c \cdot h(\mathrm{HPW}' \parallel M_A) \mathrm{mod} q$
(14) for Server S in System do
(15) $\mathrm{AM}_X^A \longleftarrow x_M \cdot h(\mathrm{ID}_A \| R_X \| h(\mathrm{ID}_X)) \mathrm{mod} q$
(16) $\mathrm{AL}_A \longleftarrow (h(\mathrm{ID}_M), \mathrm{AM}_M^A), (h(\mathrm{ID}_D), \mathrm{AM}_D^A)$
(17) end for
(18) $h_D \longleftarrow x_M \cdot h(\mathrm{ID}_D \parallel \mathrm{ID}_A) \cdot k_d \mathrm{mod} q$
(19) $t_a \longleftarrow (\mathrm{AL}_A, \mathrm{ID}_A \text{ and } \mathrm{CHM}_A)$
(20) $\mathrm{AV}_M^A \longleftarrow x_M \cdot h(t_a) + k_e \cdot T_b \mathrm{mod} q$
(21) $t_b \longleftarrow (\mathrm{AV}_M^A, \mathrm{CM}_A, \mathrm{CR}_A, r_c, h_D)$
(22) $\mathrm{AV}_M^D \longleftarrow x_M \cdot h(t_b) + k_e \cdot T_b \mathrm{mod} q$
(23) Save $\langle \mathrm{ID}_D, \mathrm{ID}_A, h_D, k_d, \mathrm{AL}_A \rangle$
Output: Send $\langle T_b, r_c, r_e, \mathrm{CHM}_A, \mathrm{CM}_A, \mathrm{AL}_A, \mathrm{CR}_A \mathrm{AV}_M^D, \mathrm{AV}_M^A, h_D \rangle$, to device D
(24) end if
(25) end if
Output: Send an error message to device D
Step 4: Device $D \longrightarrow$ User A
Input: Device D's private key x_D, shared information R_D and message from service M
(1) if this message is not an error message, then
(2) if checkTimestamps (T_b) then
(3) if $g^{\mathrm{AV}_M^D} == y_M^{h(\mathrm{AV}_M^A \| \mathrm{CM}_A \| \mathrm{CR}_A \| r_c \| h_D)} \cdot r_e^{T_b} \mathrm{mod} p$ &&$g^{\mathrm{AV}_M^A} == y_M^{h(\mathrm{AL}_A \| \mathrm{ID}_A \| \mathrm{CHM}_A)} \cdot r_e^{T_b} \mathrm{mod} p$ then
(4) Save $(CM_A, h_D, r_c, ID_A, CR_A, SCR_M = y_M^{R_D}, DM \longleftarrow y_A \cdot y_M^{ID_A \| R_D \| h(ID_D)})$
Output: Send $\langle AL_A, CHM_A, r_e, T_b, AV_M^A \rangle$ to user A
(5) end if
(6) end if
(7) end if
Output: Send an error message to user A
Step 5: User A
Input: User A's private key x_A, shared information R_A and message from Device D
(1) if this message is not an error message, then
(2) if $g^{AV_M^A} == y_M^{h(AL_A \| ID_A \| CHMA_A)} \cdot r_e^{T_b} \mathrm{mod} p$ then
(3) Keep $(\mathrm{CHM}_A, \mathrm{AL}_A, C_A)$ securely
Output: Success
(4) end if
(5) end if
Output: Failure

Algorithm 1: Service registration and binding phase.

Step 1: Device $D \longrightarrow$ Service M
Input: Device D's private key x_D and shared information R_D
(1) $k_f \longleftarrow$ freshRandomNumber()
(2) $k_h \longleftarrow$ freshRandomNumber()
(3) $r_f \longleftarrow g^{k_f} \mathrm{mod} p$
(4) $r_h \longleftarrow g^{k_h} \mathrm{mod} p$
(5) $m_w \longleftarrow (\mathrm{ID}_M, h_D, \Delta T, y_N)$
(6) $\mathrm{AV}_D^M \longleftarrow x_D \cdot h(m_w \parallel R_D) + k_f \cdot T_d \mathrm{mod} q$
Output: Send $\langle T_d, r_f, m_w, \mathrm{ID}_D, h_D, \mathrm{AV}_D^M \rangle$ to service M
Step 2: Service $M \longrightarrow$ Device D
Input: Service M's private key x_M, shared information R_D and the message from device D
(1) $(h_D, R_D) \leftarrow$ Select form its database by ID_D
(2) if $g^{\mathrm{AV}_D^M} == y_D^{h(m_w \| R_D)} \cdot r_f^{T_d} \mathrm{mod} p$ then
(3) $k_g \longleftarrow$ freshRandomNumber()
(4) $r_g \longleftarrow g^{k_g} \mathrm{mod} p.$
(5) $\sigma \longleftarrow x_M + k_g \cdot h(m_w \| r_g \| h_D \parallel T_e) \mathrm{mod} q$
(6) $\mathrm{AV}_M^D \longleftarrow x_M \cdot h(R_D \parallel \sigma) + k_g \cdot h(T_e \parallel r_g)$
Output: Send $\langle T_e, r_g, \sigma, \mathrm{AV}_M^D \rangle$ to device D

Algorithm 2: Continued.

(7) end if
Output: Send an error message to device D
Step 3: Device D
Input: Device D's private key x_D, shared information R_D and the message from service M
(1) if this message is not an error message, then
(2) if $g^{\mathrm{AV}_M^D} == y_M^{h(R_D\|\sigma)} \cdot r_g^{h(T_e\|r_g)} \bmod p \&\& g^{\sigma} == y_M \cdot r_g^{h(m_w\|r_g\|h_D\|T_g)} \bmod p$ then
(3) $x_p \longleftarrow x_D \cdot (y_D + 1) + \sigma + k_h \cdot r_h \bmod p$
(4) $y_p \longleftarrow g^{x_p} mod p = y_D^{y_D} \cdot g^{\sigma} \cdot y_N mod p$
Output: Success
(5) end if
(6) end if
Output: Failure

Algorithm 2: Proxy signature key agreement phase.

Step 1: User $A \longrightarrow$ Device D
Input: User A's $A(\mathrm{ID}_A, \mathrm{AL}_A, pw, \mathrm{BIO})$
(1) $k_a \longleftarrow \mathrm{freshRandomNumber}()$
(2) $r_a \longleftarrow g^{k_a} \bmod p$
(3) $\mathrm{OPT} \longleftarrow x_A \cdot h(\mathrm{opt}) \bmod q$
(4) $\mathrm{AAM}_X^A \longleftarrow x_A + k_a \cdot T_a + \mathrm{AM}_X^A \bmod q$
(5) $\mathrm{AAL}_A \longleftarrow (h(\mathrm{ID}_X), \mathrm{AAM}_X^A)$
(6) $(B_A, C_A) \longleftarrow \mathrm{GEN}(\mathrm{BIO})$
(7) $M_A \longleftarrow \mathrm{DES}(\mathrm{HPW}, \mathrm{CHM}_A)$
(8) $\mathrm{HM} \longleftarrow h(\mathrm{HPW} \parallel M_A)$
Output: Send $\langle \mathrm{ID}_A, \mathrm{HM}, \mathrm{OPT}, \mathrm{AAL}_A, r_a, T_a \rangle$. To device D
Step 2: Device $D \longrightarrow$ User A
Input: authentication information $(R_A, \mathrm{CM}_A, \mathrm{SCR}_M)$ and the message from user$A$
(1) if $g^{\mathrm{AAM}_D^A} == r_a^{T_a} \cdot \mathrm{DM} \bmod p$ then
(2) $R_A \longleftarrow \mathrm{DES}(R_D, \mathrm{CR}_A)$
(3) if $g^{\mathrm{CM}_A + \mathrm{OPT}} == \mathrm{SCR}_M \cdot r_c^{\mathrm{HM}} \cdot y_A^{h(\mathrm{TOTP}(R_A, T_a))} \bmod p$ then
(4) $k_u \longleftarrow \mathrm{freshRandomNumber}()$
(5) $r_u \longleftarrow g^{k_u} \bmod p$
(6) $K_S \longleftarrow r_a^{k_u}$
(7) $\mathrm{msg} \longleftarrow (ID_A, h(\mathrm{AAL}_A), T_a, r_a, r_u, T_g)$
(8) $\sigma_m \longleftarrow x_p + k_u \cdot h(\mathrm{msg}) \bmod p$
(9) $\mathrm{TOKEN} \longleftarrow (m_w, \sigma, r_g, T_e, r_u, T_g, y_p, ID_A, \mathrm{AAL}_A, T_a, r_a, \sigma_m)$
Output: Send $\langle \mathrm{TOKEN} \rangle$ to The user A
(10) end if
(11) end if
Output: Send an error message to user A
Step 3: User $A \longrightarrow$ Device D
Input: The message from device D
(1) if this message is not an error message, then
(2) $k_c \longleftarrow \mathrm{freshRandomNumber}()$
(3) $K_S \longleftarrow r_u = g^{k_a \cdot k_u}$
(4) $\mathrm{msg} \longleftarrow h(K_S \| T_c \| \mathrm{TOTP}(R_A, T_c))$
Output: Send $\langle \mathrm{msg}, \mathrm{ID}_A, T_c \rangle$ to device D
(5) end if
Output: Failure
Step 4: Device D
Input: The message from user$A$
(1) if $\mathrm{msg}' == h(K_S \| T_c \| \mathrm{TOTP}(R_A, T_c))$ then Output: Success
(2) end if
Output: Failure

Algorithm 3: User authentication and proxy signature generation phase.

If passed, server S checks user A's access right.

(3) Check user A's access right: The server S performs the following steps to check user A's access right.

(a) Check the existence of its own AAM_X^A in AAL_A.
(b) If AAM_X^A exists, verify the AAM_X^A by $g^{AAM_X^A} = y_A \cdot r_a^{T_a} \cdot y_M^{h(ID_A \| R_X \| h(ID_X))} \bmod p$.

If all of the above criteria passed, server S responds to the user A. Otherwise, server S terminates this session.

4.7. Proxy Signature Batch Verification Phase. If server S receives a large number of tokens generated by the same device D simultaneously, server S can use the following steps to perform batch verification.

(1) Server S uses the Step 1 and Step 3 defined in the Proxy Signature Verification and Authentication Phase to check the validity of m_w and user A's access right.

(2) If Step 1 and Step 3 pass, Server S performs the following steps.

(a) Combine ID_A, $h(AAL_A)_i$, T_{ai}, r_{ai}, r_{ui}, and T_{gi} into msg_i.
(b) Verify σ_{mi} by $\sum_{i=1}^{n} g^{\sigma_{mi}} = y_p \cdot \sum_{i=1}^{n} r_{ui}^{h(msg_i)} \bmod p$.

4.8. Binding Release Phase. User A can release their binding device D in this phase as shown in Algorithm 4. The authentication information, including (pw, BIO, opt), are needed to ensure that only the user can release their own devices.

4.9. Proxy Signer Privacy Revocation Phase. The server S can send the alias h_D to the service M to obtain the real identity. The service M can use the public key y_s of the server S and the shared information R_S to ensure that only the server S can obtain the real identity, and no attacker can get it from the intercepted message. A detailed description of the steps can be found in Algorithm 5.

4.10. Proxy Signature Authority Revocation Phase. Under normal circumstances, after the time limit described in device D's proxy ticket expires, device D's proxy authority will be automatically revoked. However, there will still be a problem that device D abuses its proxy signature authority. Service M should establish a public signature revocation list to solve the abovementioned issues. The general signature revocation list R_L records the r_g of the revoked device D, the timestamp T when the device was added to the list, and the degree of urgency G of the revocation.

Any verifier should check whether the r_g is on the public signature revocation list when performing proxy signature verification. If r_g is on the public revocation list, it means that service M has discovered the proxy signer for abuse of rights. At this time, it is necessary to consider the operations that need to be performed in conjunction with the timestamp ΔT and the urgency G. Table 2 shows the cancellation urgency and corresponding disposal methods.

At the same time, in order to reduce system overhead, when the ΔT of the revoked m_w expires, the r_g of the m_w will be automatically deleted from the list by the system to prevent unlimited expansion.

4.11. User Multifactor Update Phase. For the key pair (x_A, y_A), user A should apply to the service M for an update in a secure channel. Then, the service M generates a new key pair (x_A', y_A') and reissues it to user A in a secure way, then the service M publishes the new (ID_A, y_A'). User A also will update the other factors during these phases. This phase is detailed in Algorithm 6.

5. Security Analysis

The security goals provided by our scheme are shown as follows.

5.1. Informal Verification. Mutual Authentication: Our protocol can ensure the mutual authentication between the user A and the device D. At the same time, it can provide the mutual authentication between the service M and the device D.

The user's authentication vector CM_A, the one-time password secret R_A, and the service access and authentication code list AL_A are securely shared by the service M to device D during the service registration and binding process. The device D can use equation $g^{AV_M^D} = y_M^{h(AV_M^A \| CM_A \| CR_A \| r_c \| h_D)} \cdot r_e^{T_b} \bmod p$ and $g^{AV_M^A} = y_M^{h(AL_A \| ID_A \| CHM_A)} \cdot r_e^{T_b} \bmod p$ to verify the received shared secret information to ensure the data integrity of the shared secret.

In the subsequent interaction process, the service M can authenticate user A according to the authentication information passed by user A, or authenticate device D according to the authentication vector given by device D. User A and the device D can authenticate the service M according to the authentication vector returned by the service M, thereby the process can realize the mutual authentication among device D, user A, and service M.

When user A sends the authentication request to device D, device D authenticates the user by the equation $g^{CM_A + OPT} = SCR_M \cdot r_c^{HM} \cdot y_A^{h(TOTP(R_A, T_a))} \bmod p$ with the locally stored information (R_A and CM_A). After user A receives the TOKEN issued by device D, user A can authenticate device D according to the signature value. In this way, user A and device D complete the mutual authentication.

Data Confidentiality and Integrity: In this scheme, symmetric encryption is used to ensure data confidentiality and a signature is used to ensure data integrity. The private data such as $\langle pw, BIO, B, C, opt, x_A, M_A, R_A \rangle$ are all securely shared in the abovementioned form in our protocol. It is not transmitted or transferred in plain text to the user or device. The attacker cannot obtain the private data by

Step 1: User $A \longrightarrow$ Device D
Input: User A inputs pw, BIO, opt
(1) $k_a \longleftarrow$ freshRandomNumber()
(2) $\mathrm{OPT} \longleftarrow x_A \cdot h(\mathrm{opt}) \bmod q$
(3) $\mathrm{AAM}_X^A \longleftarrow x_A + k_a \cdot T_a + \mathrm{AM}_X^A \bmod q$
(4) $\mathrm{AAL}_A \longleftarrow (h(\mathrm{ID}_X), \mathrm{AAM}_X^A)$
(5) $(B_A, C_A) \longleftarrow \mathrm{GEN}(\mathrm{BIO})$
(6) $M_A \longleftarrow \mathrm{DES}(\mathrm{HPW}, \mathrm{CHM}_A)$
(7) $\mathrm{HM} \longleftarrow h(\mathrm{HPW} \parallel M_A)$
Output: Send $\langle \mathrm{ID} \rangle \mathrm{ID}_A, \mathrm{HM}, \mathrm{OPT}, \mathrm{AAL}_A, r_a, T_a \rangle$ to device D
Step 2: Device D
Input: The message from user A
(1) $R_A \longleftarrow \mathrm{DES}(R_D, \mathrm{CR}_A)$
(2) if $g^{\mathrm{CM}_A + \mathrm{OPT}} == \mathrm{SCR}_M \cdot r_c^{\mathrm{HM}} \cdot y_A^{h(\mathrm{TOTP}(R_A, T_a))} \bmod p$ then
(3) delete all information about ID_A
Output: Success
(4) end if
Output: Failure

Algorithm 4: Binding release phase.

Step 1: Server $S \longrightarrow$ Service M
Input: Server S's private key x_s
(1) $k_a \longleftarrow$ freshRandomNumber()
(2) $r_a \longleftarrow g^{k_a} \bmod p$
(3) $\mathrm{AV}_S^M \longleftarrow x_s \cdot h(\mathrm{ID}_S \parallel h_D) + k_a \cdot h(T_a \parallel r_a) \bmod q$
Output: Send $\langle \mathrm{ID} \rangle \mathrm{ID}_S, h_D, r_a, T_a, \mathrm{AV}_S^M \rangle$ to service M
Step 2: Service $M \longrightarrow$ Server S
Input: The message from server S and the shared information R_S
(1) if $g^{\mathrm{AV}_S^M} == y_s^{h(\mathrm{ID}_S \| h_D)} \cdot r_a^{h(T_a \| r_a)} \bmod p$ then
(2) select (ID_D, k_d) from its database by h_D
(3) $k_c \longleftarrow$ freshRandomNumber()
(4) $r_c = g^{k_c} \bmod p$
(5) $\mathrm{CM}_S = \mathrm{ENC}(R_S, \mathrm{ID}_D \| \mathrm{ID}_A \| y_M^{k_d})$
(6) $\mathrm{AV}_M^S = x_M \cdot h(T_b \parallel \mathrm{CM}_S) + k_c \cdot T_b \bmod q$
Output: Send$< \langle T_b, r_c, \mathrm{AV}_M^S, \mathrm{CM}_S \rangle$ to server S
(7) end if
Output: Send an error message to server S
Step 3: Server S
Input: The message from service M and the shared information R_S
(1) if this message is not an error message, then
(2) if $g^{\mathrm{AV}_M^S} == y_M^{h(T_b \| \mathrm{CM}_S)} \cdot r_c^{T_b} \bmod p$ then
(3) $(\mathrm{ID}_D \| \mathrm{ID}_A \| y_M^{k_d}) \longleftarrow \mathrm{DES}(R_S, \mathrm{CM}_S)$
(4) if $g^{h_D} == y_M^{k_d \cdot h(\mathrm{ID}_D \| \mathrm{ID}_A)} \bmod p$ then
Output: ID_D
(5) end if
(6) end if
(7) end if
Output: Failure

Algorithm 5: Proxy signer privacy revocation phase.

Table 2: Cancellation urgency and corresponding disposal method table.

Level	Approach
Attention	The service M will retain all proxy signatures generated by r_g.
Warning	After r_g is added to the list, all proxy signatures generated by r_g are invalid.
Serious	All proxy signatures generated by r_g are invalid.

Step 1: User $A \longrightarrow$ Device D
Input: User A inputs $(pw, \mathrm{BIO}, pw_n, \mathrm{BIO}_n)$
(1) $k_a \longleftarrow \mathrm{freshRandomNumber}()$
(2) $k_l \longleftarrow \mathrm{freshRandomNumber}()$
(3) $r_a \longleftarrow g^{k_a} \bmod p$
(4) $\mathrm{OPT} \longleftarrow x_A \cdot h(\mathrm{TOTP}(R_A, T_a)) \bmod p$
(5) $r_l \longleftarrow g^{k_l} \bmod p$
(6) $\mathrm{AAM}_X^A \longleftarrow x_A + k_a \cdot T_a + \mathrm{AM}_X^A \bmod q$
(7) $\mathrm{AAL}_A \longleftarrow (h(\mathrm{ID}_X), \mathrm{AAM}_X^A)$
(8) $(B_A, C_A) \longleftarrow \mathrm{GEN}(\mathrm{BIO})$
(9) $M_A \longleftarrow \mathrm{DES}(\mathrm{HPW}, \mathrm{CHM}_A)$
(10) $\mathrm{HM} \longleftarrow h(\mathrm{HPW} \parallel M_A)$
(11) $(B_{An}, C_{An}) \longleftarrow \mathrm{GEN}(\mathrm{BIO}_n)$
(12) $M_1 \longleftarrow \mathrm{ENC}(R_A, h(pw_n) \parallel h(B_{An}))$
(13) $(B_{An}, C_{An}) \longleftarrow \mathrm{GEN}(\mathrm{BIO}_n)$
(14) $\mathrm{HPW}_n \longleftarrow h(h(pw_n) \parallel h(B_{An}))$
(15) $AV_A^M \longleftarrow x_{An} \cdot \mathrm{HPW}_n + k_l \cdot h(T_a \| h(\mathrm{TOTP}(R_A, T_a)) \| M_1) \bmod q$
Output: Send $\langle \mathrm{ID}_A, \mathrm{HM}, \mathrm{OPT}, \mathrm{AAL}_A, r_a, T_a r_l, \mathrm{AV}_A^M, M_1 \rangle$, to device D
Step 2: Device $D \longrightarrow$ Service M
Input: The message from user A
(1) $k_d \longleftarrow \mathrm{freshRandomNumber}()$
(2) $r_d \longleftarrow g^{k_d} \bmod p$
(3) $\mathrm{AV}_D^M \longleftarrow x_D \cdot h(\mathrm{AV}_A^M) + k_d \cdot T_b \bmod p$
Output: Send $\langle \mathrm{ID}_D, \mathrm{ID}_A, \mathrm{AV}_D^M, \mathrm{AV}_A^M, r_l, r_d T_b, T_a, M_1 \rangle$, $T_b, T_a, M_1 >$ to server M
Step 3: Service $M \longrightarrow$ Device D
Input: The message from device D
(1) selects $(h(pw), h(B_A), R_A)$ from its database by ID_A
(2) $\mathrm{HOPT} \longleftarrow h(\mathrm{TOTP}(R_A, T_a))$
(3) $(h(pw_n) \parallel h(B_{An})) \longleftarrow \mathrm{DES}(R_A, M_1)$
(4) $\mathrm{HPW}_n' \longleftarrow h(h(pw_n) \parallel h(B_{An}))$
(5) if $g^{\mathrm{AV}_A^M} == y_A^{\mathrm{HPW}_n'} \cdot r_l^{h(T_a \| h(\mathrm{TOTP}(R_A, T_a)) \| M_1))} \bmod p$ && $g^{\mathrm{AV}_A^M} == y_A^{\mathrm{HPW}_n'} \cdot r_l^{h(T_a \| h(\mathrm{TOTP}(R_A, T_a)) \| M_1))} \bmod p$ then
(6) $k_{\longleftarrow} \longleftarrow \mathrm{freshRandomNumber}()$
(7) $k_e \longleftarrow \mathrm{freshRandomNumber}()$
(8) $r_{\longleftarrow} \longleftarrow g^{k_{\longleftarrow}} \bmod p$
(9) $r_e \longleftarrow g^{k_e} \bmod p$
(10) $\mathrm{CR}_{An}^D \longleftarrow \mathrm{ENC}(R_D, R_{An})$
(11) $\mathrm{CR}_{An}^A \longleftarrow \mathrm{ENC}(R_A, R_{An})$
(12) $\mathrm{CM}_{An} \longleftarrow x_M + k_{\longleftarrow} \cdot h(\mathrm{HPW}_n' \parallel M_A) \cdot R_D \bmod q$
(13) $\mathrm{AV}_M^A \longleftarrow x_M \cdot h(\mathrm{CR}_{An}^A) + k_e \cdot T_c \bmod q.$
(14) $\mathrm{AV}_M^D \longleftarrow k_e \cdot T_c + x_M \cdot h(\mathrm{CM}_{An} \| \mathrm{CR}_{An}^A \| \mathrm{CR}_{An}^D \parallel r_{\longleftarrow}) \bmod q.$
Output: Send $\langle T_c, r_{\longleftarrow}, r_e, \mathrm{CM}_A, \mathrm{CR}_{An}^D, \mathrm{CR}_{An}^A, \mathrm{AV}_M^D \mathrm{AV}_M^A \rangle$, to device D.
(15) end if
Output: Send an error message to device D
Step 4: Device $D \longrightarrow$ User A
Input: The message from service M
(1) if this message is not an error message, then
(2) if $g^{\mathrm{AV}_M^D} == y_M^{h(\mathrm{CM}_{An} \| \mathrm{CR}_{An}^A \| \mathrm{CR}_{An}^D \| r_{\longleftarrow})} \cdot r_e^{T_c} \bmod p$ then
(3) $R_{An} \longleftarrow \mathrm{DES}(R_D, \mathrm{CR}_{An}^D)$
(4) Update its database $(\mathrm{CM}_{An}, r_{\longleftarrow}, R_{An})$ by ID_A
Output: Send $\langle T_c, r_e, \mathrm{CR}_{An}^A, \mathrm{AV}_M^A \rangle$ to user A
(5) end if
(6) end if
Output: Send an error message to user A
Step 5: User A
Input: The message from device D
(1) if this message is not an error message, then
(2) if $g^{\mathrm{AV}_M^A} == y_M^{h(\mathrm{CR}_{An}^A)} \cdot r_e^{T_c} \bmod p$ then
(3) $R_{An} \longleftarrow \mathrm{DES}(R_A, \mathrm{CR}_{An}^A)$
Output: Success
(4) end if
(5) end if
Output: Failure

Algorithm 6: User multi-factor update phase.

calculating the discrete logarithm. In addition, its private key of each entity is used to sign each interactive data to protect the integrity. Once the received data are modified, the verification equation does not hold. Therefore, the proposed scheme can guarantee data integrity.

Resistance to Several Types of Attacks:

(i) Replay Attack: Our scheme uses the timestamp and the nonce to resist replay attacks. When the authentication vector is generated, they are encrypted with the private key of the user or device. Based on the abovementioned proof, the attacker cannot obtain the user or device's private key in polynomial time. Therefore, the attacker cannot modify the replay message's timestamp or random number to implement the replay attack successfully.

(ii) Forward and Backward Secrecy: The session key $K_S = r_a^{k_u}$ is only affected with the random numbers k_a and k_u. Even if the Eve obtains all long-term secrets of A and D and can intercept all messages, the previous and future session keys are still secure because of the CDH and DDH problems.

(iii) Mobile Device Loss Attacks: Once Eve steals a mobile device of A, Eve can extract all stored values $\langle C_A, CHM_A, ALA\rangle$ by side-channel attacks. Eve can also get ID_A by analyzing the messages. Eve will try to guess A's pw and B_A if the above process happens. It is intractable to guess the two values in the meantime because of the collision-resistant property of the hash function.

(iv) Offline Password Guessing: Only if Eve can obtain the pw, and B_A, he or she can successfully impersonate user A. However, Eve cannot reveal the B_A using only C_A without knowing the BIO. Eve also needs pw to compute HPW to decrypt CHM_A, However, HPW is not stored in any device. There is no way but to guess for Eve to obtain the pw in offline password guessing attack.

(v) Privileged Insider Attacks: In our scheme, the user A's pw is not transmitted in any phases. We use B_A and random number r_c to mask the pw. Obviously, only A owns the pw, so our scheme is resistant to privileged insider attacks.

Table 3 epitomizes the comparison of the security properties among ours and the schemes in [12–15].

Proxy Privacy: We use the alias h_D to protect the device's privacy in our scheme. The alias h_D not only hides the device's true ID but shows the relationship of ID_A and ID_D as well. Therefore, for each user bonded with the device ID_D, the device will own a unique alias h_D to present the relationship. The alias h_D is calculated by the private key of service M, ID_A, ID_D, and k_d through the hash function. No one can directly obtain the true ID through the alias h_D or infer from the other alias the association implied by this alias. Therefore, this scheme can realize the anonymity of proxy signatures.

Strong Unforgeability: The proxy signature private key x_P is comprised of the proxy private keys of the proxy signer and the original signer and the secret value of the proxy signer. y_P, the public key of the proxy signer, includes the proxy signer's customized and secret data and the original signer's data. When the proxy signature public key y_P, the signature σ of the proxy signature warrant, and the device public keys y_D and y_N are known, the original signer or attacker cannot obtain the proxy signatures private key x_P through the free combination or calculation of the above information. Therefore, this scheme satisfies the strong unforgeability of proxy signatures.

Proxy signer's deviation: In our scheme, when the proxy signer tries to generate a valid m_w, the difficulty lies in finding the original signer's x_M and $r^{-1} mod p$ in polynomial time when only one alias h_D and a proper original signature σ are known. However, the attacker cannot solve the discrete logarithm in polynomial time, so this scheme satisfies the nondivergence of proxy signatures.

Verifiability: In our scheme, the proxy signatures can be verified by $g^{\sigma_m} = y_p \cdot r_u^{h(msg)} \bmod p$.

Strong identifiability: The format is different between the raw signatures (m, σ, r_g, T_e) and proxy signatures $(m_w, h_D, \sigma, r_g, r_u, T_e, T_g, y_p, \sigma_m, m)$ in our scheme. Anyone can distinguish the raw signatures and proxy signatures in polynomial time.

Strong Recognizability: When verifying the signature, anyone can get $\langle h_D, m_w, \sigma\rangle$ in the proxy signature. Only the original signer M can generate the alias h_D and the correct signature of proxy signature warrant in this scheme. Therefore, any verifier can believe that the h_D is who produces the proxy signature.

Strong Nonrepudiation: When the original signer M tries to attack a proxy signature private key x_P, the original signer M needs to know x_D, x_M, and k_h. However, in this scheme, $x_N = x_D + k_h \cdot r_h mod p$ is something that the original signer M and any attacker cannot calculate in polynomial time.

Anti-abuse: In this scheme, a signature revocation list is added to prevent the proxy signer from abusing the proxy signature authority. Once the proxy signer is found to be abusive, the original signer can add the corresponding information to the signature revocation list and correct the proxy signer's abuse further.

5.2. Scyther Simulation. In this section, we conduct a Scyther simulation [46] to prove the security of our scheme. The Scyther tool can output the protocol role execution trajectory. In addition, the several security models are included in Scyther simulation tool, such as the standard Dolev–Yao model and other nine adversary models, such as CK model and eCK model etc. The Scyther tool uses the Security Protocol Description Language (SPDL) to describe and analyze security protocols and can detect protocol vulnerabilities such as replay, man-in-the-middle, and reflection

Table 3: Security properties comparison.

Security properties	[12]	[13]	[14]	[15]	Our scheme
Mutual authentication	Y	Y	Y	Y	Y
Multifactor authentication	3 factors	3 factors	3 factors	1 factor	4 factors
Authorization	N	Y	N	N	Y
Privileged insider attacks	Y	Y	Y	Y	Y
Mobile device loss attacks	Y	Y	Y	N	Y
Forward and backward secrecy	Y	Y	Y	Y	Y

Protocol description Settings

```
1 usertype text;
2 hashfunction H, gexp, exp, REP, TOTP;
3 protocol mpaa(A, P){
4   role A{
5     fresh IDD, IDA, pw, BIO, opt, Ta, ka, CA, xa, MA, AMDA: Nonce;
6
7     // authentication phase begin
8     var HM, OPT, AALA : text;
9     match(HM, H(H(pw), H(REP(BIO,CA)), MA));
10    match(OPT, exp(xa, H(opt)));
11    match(AALA, {xa, exp(ka, Ta), AMDA}sk(A));
12    send_1(A, P, IDA, HM, OPT, AALA, gexp(ka), Ta);
13    fresh xp, ku, Tg , hD:Nonce;
14    var signM : text;
15    var KS:text;
16
17    recv_2(P, A, gexp(xp), signM, IDA, hD, gexp(ku), Tg, Ta, gexp(ka), AALA, Ta);
18    match(KS, exp(gexp(ku), ka));
19    var msg:text;
20    fresh TC:Nonce;
21    match(msg, H(KS,TC,TOTP(TC)));
22    send_3(A, P, IDA, TC, msg);
23
24        claim(A, Secret, pw);
25        claim(A, Secret, BIO);
26        claim(A, Secret, opt);
27        claim(A, Secret, xa);
28        claim(A, Secret, CA);
29        claim(A, Secret, MA);
30        claim(A, Secret, AMDA);
31        claim(A, Secret, ka);
32        claim(A, Secret, KS);
33        claim(A, Alive);
34        claim(A, Weakagree);
35        claim(A, Niagree);
36        claim(A, Nisynch);
37    // authentication phase end
38  }
39  role P{
40    fresh IDD, IDA, ka, Ta : Nonce;
41    var HM, OPT, AALA : text;
42    // authentication phase begin
43    recv_1(A, P, IDA, HM, OPT, AALA, gexp(ka), Ta);
44    fresh Tg, ku, xp, hD: Nonce;
45    var msg, signM : text;
46    var KS:text;
47    match(KS, exp(gexp(ka),ku));
48    match(msg, (IDA, H(AALA), Ta, gexp(ka), gexp(ku), Tg));
49    match(signM, {xp, exp(ku, H(msg))}sk(P));
50    send_2(P, A, gexp(xp), signM, IDA, hD, gexp(ku), Tg, Ta, gexp(ka), AALA, Ta);
51    var msg:text;
52    fresh TC:Nonce;
53    recv_3(A, P, IDA, TC, msg);
54
55        claim(P, Secret, xp);
56        claim(P, Secret, ku);
57        claim(P, Secret, KS);
58        claim(P, Alive);
59        claim(P, Weakagree);
60        claim(P, Niagree);
61        claim(P, Nisynch);
62
63    // authentication phase end
64  }
65 }
```

Figure 4: Simulation code during registration phase.

Protocol description Settings

```
1 usertype text;
2 hashfunction H, gexp, exp, REP, TOTP;
3 protocol mpaa(A, P){
4   role A{
5     fresh IDD, IDA, pw, BIO, opt, Ta, ka, CA, xa, MA, AMDA: Nonce;
6
7     // authentication phase begin
8     var HM, OPT, AALA : text;
9     match(HM, H(H(pw), H(REP(BIO,CA)), MA));
10    match(OPT, exp(xa, H(opt)));
11    match(AALA, {xa, exp(ka, Ta), AMDA}sk(A));
12    send_1(A, P, IDA, HM, OPT, AALA, gexp(ka), Ta);
13    fresh xp, ku, Tg , hD:Nonce;
14    var signM : text;
15    var KS:text;
16
17    recv_2(P, A, gexp(xp), signM, IDA, hD, gexp(ku), Tg, Ta, gexp(ka), AALA, Ta);
18    match(KS, exp(gexp(ku), ka));
19    var msg:text;
20    fresh TC:Nonce;
21    match(msg, H(KS,TC,TOTP(TC)));
22    send_3(A, P, IDA, TC, msg);
23
24        claim(A, Secret, pw);
25        claim(A, Secret, BIO);
26        claim(A, Secret, opt);
27        claim(A, Secret, xa);
28        claim(A, Secret, CA);
29        claim(A, Secret, MA);
30        claim(A, Secret, AMDA);
31        claim(A, Secret, ka);
32        claim(A, Secret, KS);
33        claim(A, Alive);
34        claim(A, Weakagree);
35        claim(A, Niagree);
36        claim(A, Nisynch);
37    // authentication phase end
38  }
39  role P{
40    fresh IDD, IDA, ka, Ta : Nonce;
41    var HM, OPT, AALA : text;
42    // authentication phase begin
43    recv_1(A, P, IDA, HM, OPT, AALA, gexp(ka), Ta);
44    fresh Tg, ku, xp, hD: Nonce;
45    var msg, signM : text;
46    var KS:text;
47    match(KS, exp(gexp(ka),ku));
48    match(msg, (IDA, H(AALA), Ta, gexp(ka), gexp(ku), Tg));
49    match(signM, {xp, exp(ku, H(msg))}sk(P));
50    send_2(P, A, gexp(xp), signM, IDA, hD, gexp(ku), Tg, Ta, gexp(ka), AALA, Ta);
51    var msg:text;
52    fresh TC:Nonce;
53    recv_3(A, P, IDA, TC, msg);
54
55        claim(P, Secret, xp);
56        claim(P, Secret, ku);
57        claim(P, Secret, KS);
58        claim(P, Alive);
59        claim(P, Weakagree);
60        claim(P, Niagree);
61        claim(P, Nisynch);
62
63    // authentication phase end
64  }
65 }
```

Figure 5: Simulation code during authentication phase.

attacks based on several different security objectives. The Scyther tool will automatically output the verification results: if the assumption is valid, then output ok and verified. If the assumption is not valid, then output is false and give the existing attack.

In the proposed model described by SPDL, to simplify the analysis, this article defines three roles, A, P, and M, which represent user A, device D, and service M,

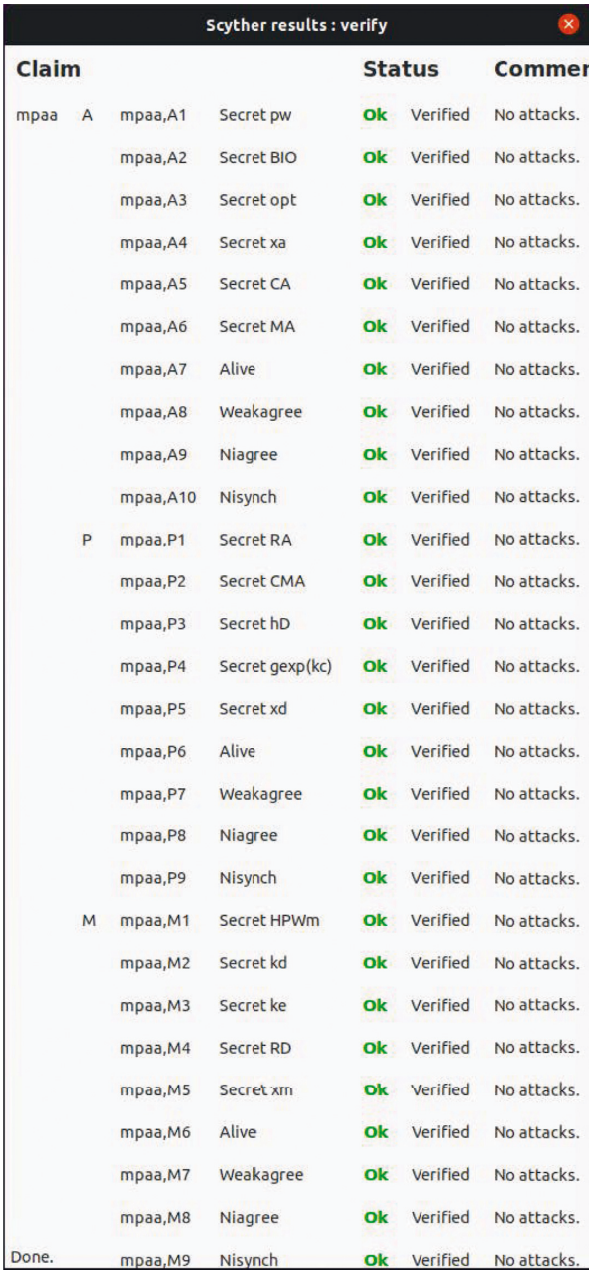

Figure 6: Simulation results during the registration phase.

respectively. We verify the proposed scheme under the Dolev–Yao attack model.

As shown in Figures 4 and 5, this article uses the SPDL language to model the protocol. The modeling of this scheme is divided into two steps, registration and certification. The security of the proposed scheme can be ensured through a series of Scyther's requirements. As shown in Figures 6 and 7, the designed scheme meets all the security objectives in Scyther, and no attack is found under its simulation.

5.3. Authentication Proof Using BAN Logic. We use BAN logic [47] to prove the security of the proposed scheme. Since the registration phase runs in a secure channel, only the security of the User Authentication and Proxy Signature

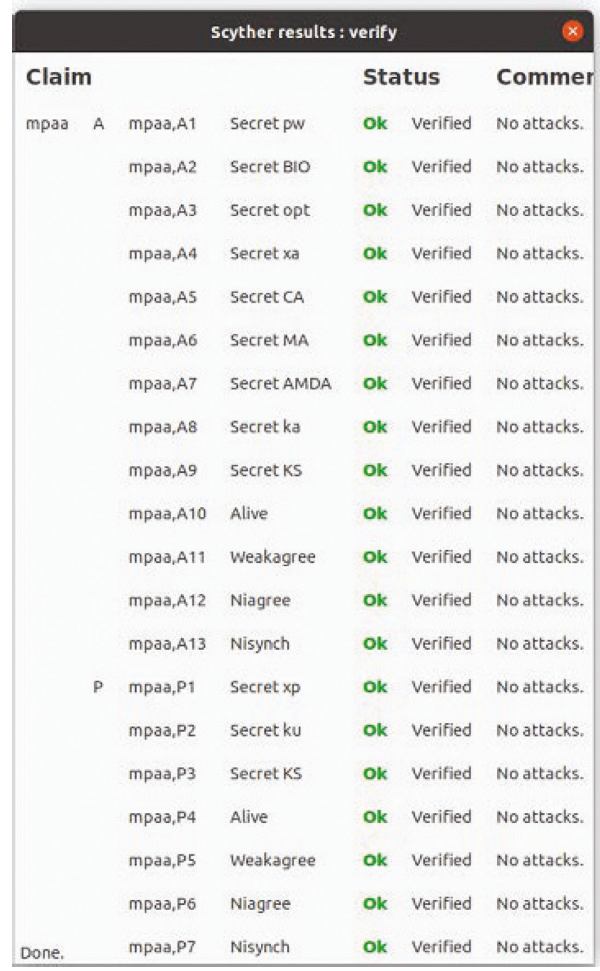

Figure 7: Simulation results during the authentication phase.

Generation Phase needs to be considered. The User Authentication and Proxy Signature Generation Phase must meet the following goals:

Goal 1. User $|\equiv (Device \overset{KS}{\leftrightarrow} User)$

Goal 2. Device $|\equiv (Device \overset{KS}{\leftrightarrow} User)$

Goal 3. User $|\equiv$ Device $|\equiv (Device \overset{KS}{\leftrightarrow} User)$

Goal 4. Device $|\equiv$ User $|\equiv (Device \overset{KS}{\leftrightarrow} User)$

First, we convert the protocol as follows:

Msg 1. User $\longrightarrow$ Device: $\left\{\{N_A\}_{K_A^{-1}}, N_A\right\}$

Msg 2. Device $\longrightarrow$ User: $\left\{N_B, \{N_B\}_{K_P^{-1}}\right\}$

Msg 3. User $\longrightarrow$ Device: $\{N_C\}_{K_S}$

We then define the hypotheses and initial status as follows:

$A1$. User $|\equiv \sharp N_A$

$A2$. Device $|\equiv \sharp N_B$

$A3$. User $|\equiv$ Device $|\Rightarrow N_B$

$A4$. Device $|\equiv$ User $|\overset{k_P}{\Rightarrow} N_A$

$A5$. User $|\equiv| \overset{k_A}{\longrightarrow}$ Device

$A6$. Device $|\equiv| \longrightarrow$ User

Table 4: Performance analysis symbol definition.

Operation	Symbol
Exponential operation of discrete logarithms	T_{dlexp}
Modulo operation	T_{mod}
Multiplication operation of discrete logarithms	T_{dlmul}
Hash operation	T_h
AES operation	T_{aes}
Point double operation of discrete logarithms	T_{eccmul}

Table 5: Time requirements for various operations (unit: microseconds μs).

Symbol	Client	Server
T_{dlexp}	5.927	4.829
T_{mod}	2.477	2.201
T_{dlmul}	0.008	0.005
T_h	0.050	0.033
T_{aes}	0.670	0.392
T_{eccmul}	15.100	12.980

Table 6: Computation costs of related schemes (unit: microseconds μs).

Computation costs	Client	Server	Total
[12]	$9T_h + T_{\text{aes}} + 2T_{\text{eccmul}} = 30.942$	$4T_h + 3T_{eccmul} = 38.96$	69.283
[13]	$15T_h + 3T_{\text{eccmul}} = 45.42$	$17T_h + 3T_{\text{eccmul}} = 39.025$	84.445
[14]	$T_{\text{mod}} + 7T_h + 5T_{\text{eccmul}} = 78.033$	$4T_h + 4T_{\text{eccmul}} = 51.94$	129.973
[15]	$5T_h + 2T_{\text{eccmul}} = 30.24$	$15T_h + 6T_{\text{eccmul}} = 77.955$	108.195
Ours	$2T_{\text{DLP}} + 3T_{\text{mod}} + 9T_h + 2T_{\text{dlmul}} + T_{\text{aes}} = 20.127$	$7T_{\text{DLP}} + 4T_{\text{mod}} + 5T_h + 4T_{\text{dlmul}} + T_{\text{aes}} = 43.156$	63.283

Table 7: Storage cost (bytes).

Scheme	Client	Server
Ours	112	180
[12]	80	16
[14]	49	16
[13]	224	144
[15]	64	112

Table 8: Communication overhead (bytes).

Scheme	Rounds	Overhead
Ours	4	132 + 124 + 52 = 308
[12]	2	200
[14]	4	224
[13]	4	528
[15]	4	516

Based on the above informations, the proofs are presented as follows:

From the Msg 1, we have the following:

$$\text{Device} \triangleleft \{N_A\}_{K_A^{-1}}. \tag{3}$$

From $A6$, with the successful confirmation and the message-meaning rule, we can get as follows:

$$\text{Device} \mid\equiv \text{User} \mid \sim \left\{\{N_A\}_{K_A^{-1}}\right\}. \tag{4}$$

According to the N_A, freshness rule, $A1$ and $A4$, we can get the following:

$$\text{Device} \mid \equiv \sharp\{N_A\}. \tag{5}$$

According to the nonce verification rule and N_A, we can derive the following:

$$\text{Device} \mid \equiv \text{User} \mid \equiv \{N_A\}. \tag{6}$$

According to $A4$, N_A and the jurisdiction rule, we can get the following:

$$\text{Device} \mid \equiv \{N_A\}. \tag{7}$$

Device can compute the $K_S = N_A^{N_B}$, according to the belief rule, we can derive the Goal 2:

$$\text{Device} \mid \equiv \left(\text{User} \overset{K_S}{\leftrightarrow} \text{Device}\right). \tag{8}$$

Equally, we can derive Goal 1 according to Msg 2 and the same deductions.

From Msg 3, we get the following:

$$\text{Device} \triangleleft \{N_C\}_{K_S}. \tag{9}$$

According to Goal 2 we have proved, and message-meaning rule, we can derive the following:

$$\text{Device} \mid \equiv \text{User} \mid \sim \{N_C\}. \tag{10}$$

Based on the $A2$, the random number and freshness rule and the nonce verification rule, we can derive the following:

$$\text{Device} \mid \equiv \text{User} \mid \equiv \{N_C\}. \tag{11}$$

Since N_C is composed of $TOTP$ codes, legal $TOTP$ codes can only be generated by legal users, and according to Device $\mid \equiv$ User $\mid \equiv \{N_A\}$ and the belief rule, we can derive the Goal 4:

$$\text{Device} \mid \equiv \text{User} \mid \equiv \left(User \overset{K_S}{\leftrightarrow} Device\right). \tag{12}$$

Equally, we can derive Goal 3 according to Msg 2 and the same deductions.

6. Performance Analysis

In this section, we analyze the performance of the solution from three aspects: (1) communication overhead; (2) storage overhead; and (3) calculation overhead and compare with [12–15]. Table 4 shows the symbols used in this section.

In terms of communication overhead, this article mainly considers the communication overhead in user authentication and proxy signature generation phases. Without loss of generality, the security level defined in this article is equivalent to Advanced Encryption Standard (AES) 128 bits [48, 49], and the AES algorithm is used as the symmetric encryption algorithm. Specifically, the length of ID and PW is set as 128 bits, the key length based on the discrete logarithm algorithm is 256 bits. The key length of the ECC-based cryptographic algorithm is 256 bits, the hash function outputs 256-bits-length result, the random number is 128 bits, the identification length is 128 bits, the timestamp is 32 bits, and the auxiliary string C is 128 bits. 7 hash functions are defined in [15], and we consider that the output length and computational overhead of the above hash functions are consistent with those used in this paper.

On the computational cost, we only consider the algorithm shown in Table 4, ignoring other less time-consuming operations, and this article assumes that the GEN, REP, and $TOTP$ functions all have the time for hashing complexity. This article uses the JAVA BigInteger library and JPBC library on device 1 (Intel(R) Core(TM) i7-6700K CPU @ 4.00 GHz) as a client and device 2 (Intel(R) Core(TM) i5-10600 KF CPU @ 4.10 GHz) as a server to measure the time of related operations, as shown in Table 5. Refering to [13], we consider GW_i as device 2. For convenience, we divide the overhead on GW_i as part of the AAS overhead.

Table 6 lists the computational cost of the schemes and related schemes in [12–15] in the process of User Authentication and Proxy Signature Generation phase. The running time of the proposed scheme is 63.283 (20.127 for client, 43.156 for server). The running time of [12–15] is 69.902 (30.942 for client, 38.96 for server), 84.445 (45.52 for client, 39.025 for server), 129.973 (78.033 for client, 51.94 for server), and 108.195 (30.24 for client, 77.955 for server), respectively. The results show that the computational overhead of our scheme, especially the client's, is smaller than [12–15]. The server is always considered a rich-resource entity. Therefore, it is acceptable that the computational overhead of the server in our scheme is a little larger than that in [12].

On the storage cost, the client has to store (CHM_A, AL_A, C_A, x_A) and the server needs to store $(CM_A, h_D, r_c, r_g, k_h, ID_A, CR_A, m_w, \sigma, T_e, x_D, x_p)$. Table 7 shows that the total storage cost of our scheme is lower than that of the scheme in [13], which is little larger than that of the schemes in [12, 14, 15].

On the communication overhead, the proposed scheme completing authentication protocol needs 4 rounds. Table 8 shows the communication overhead of [12–14] and ours. From Table 8, it can be inferred that the communication overhead of our scheme is smaller than that of the scheme in [13], which is little larger than the schemes in [12, 14].

Based on the above results of comparison and analysis, our scheme outperforms the schemes in [12–15] in the term of computation cost, and better than the scheme in [14] in the term of storage cost and communication cost. Our scheme needs more storage than the scheme in [12] due to the more fine-grained access control of users.

7. Conclusions

We propose a secure multifactor authentication and access control mechanism base on proxy signature for cloud-fog hybrid electronic bill service in the 5G network. The results of BAN logic and Scyther simulation show that our scheme is security, and performance analysis shows that our scheme has a lower computational overhead compared with other schemes. In addition, the proposed scheme can provide more security features without sacrificing performance.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Key R&D Program of China (No. 2018YFB0803900), the National Natural Science Foundation of China (Nos. 62102298, 62172317, and U1836203), the Key R&D Program of Shaanxi (No. 2020ZDLGY08-08), and the Fundamental Research Funds for the Central Universities and the Innovation Fund of Xidian University (No. YJS2114).

References

[1] L. L. Guo, S. P. Wang, S. S. Feng, W. Lu, and F. Z. Luo, "Exploration and innovative application of electronic invoices based on "Internet + power marketing," *POWER DSM*, vol. 18, no. S1, pp. 61–63, 2018.

[2] F. Wang, *Study on Synergy Effect between E-Invoice System and enterprise's Accounting system*, Capital University of Economics and Business, China, 2017.

[3] 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.

[4] 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.

[5] 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 ID 102660, 2020.

[6] Y. C. Yu, *Research on Identity Authentication Protocol in Multi-Server Network environment*, Jilin University, China, 2020.

[7] Y. Yao, *Research on Key Technologies of Internet Cross-Domain Authentication[D]*, Northeastern University, Boston, MA, USA, 2012.

[8] S. C. Lu, *Research on the Multi-Identity Management Mechanism of Electronic Bills Service system*, Xidian University, China, 2019.

[9] E. Ahvar, S. Ahvar, S. M. Raza, J. Manuel Sanchez Vilchez, and G. M. Lee, "Next generation of SDN in cloud-fog for 5G and beyond-enabled applications: opportunities and challenges," *Network*, vol. 1, no. 1, pp. 28–49, 2021.

[10] R. Deng, R. Lu, C. Lai, and T. H. Luan, "Towards Power Consumption-Delay Tradeoff by Workload Allocation in Cloud-Fog computing," in *Proceedings of the 2015 IEEE International Conference on Communications (ICC)*, pp. 3909–3914, IEEE, London UK, 2015.

[11] 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.

[12] H. Amintoosi, M. Nikooghadam, S. Kumari, S. Kumar, and C.-M. Chen, *TAMA: Three-Factor Authentication for Multi-Server Architecture*, p. 11 Human-centric Computing and Information Sciences, Seoul, Republic of Kora, 2021.

[13] S. Shin and T. Kwon, "A privacy-preserving authentication, authorization, and key agreement scheme for wireless sensor networks in 5G-integrated Internet of Things," *IEEE access*, vol. 8, pp. 67555–67571, 2020.

[14] H. Luo, F. Wang, and G. Xu, "Provably secure ECC-based three-factor Authentication scheme for mobile cloud computing with offline registration centre," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8848032, 12 pages, 2021.

[15] X. Liu, W. Ma, and H. Cao, "NPMA: a novel privacy-preserving mutual authentication in TMIS for mobile edge-cloud architecture," *Journal of Medical Systems*, vol. 43, no. 10, pp. 1–16, 2019.

[16] D. He and D. Wang, "Robust biometrics-based authentication scheme for multiserver environment," *IEEE Systems Journal*, vol. 9, no. 3, pp. 816–823, 2014.

[17] 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.

[18] 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.

[19] 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.

[20] S. Barman, H. P. H. Shum, S. Chattopadhyay, and D. Samanta, "A secure authentication protocol for multi-server-based E-healthcare using a fuzzy commitment scheme," *IEEE Access*, vol. 7, pp. 12557–12574, 2019.

[21] H. Lin, F. Wen, and C. Du, "An improved anonymous multi-server authenticated key agreement scheme using smart cards and biometrics," *Wireless Personal Communications*, vol. 84, no. 4, pp. 2351–2362, 2015.

[22] J. Moon, J. Yu, H. Yang, and D. Won, "Improvement of biometrics and smart cards-based authentication scheme for multi-server environments," in *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, pp. 1–8, Danang Vietnam, January 2016.

[23] 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, pp. 3219–3228, 2015.

[24] A. Irshad, S. A. Chaudhry, S. Kumari, M. Usman, K. Mahmood, and M. S. Faisal, "An improved lightweight multiserver authentication scheme," *International Journal of Communication Systems*, vol. 30, no. 17, p. e3351, 2017.

[25] C.-T. Chen and C.-C. Lee, "A two-factor authentication scheme with anonymity for multi-server environments," *Security and Communication Networks*, vol. 8, no. 8, pp. 1608–1625, 2015.

[26] J. Moon, Y. Choi, J. Jung, and D. Won, "An improvement of robust biometrics-based authentication and key agreement scheme for multi-server environments using smart cards," *PLoS One*, vol. 10, no. 12, Article ID e0145263, 2015.

[27] C. Wang, X. Zhang, and Z. Zheng, "Cryptanalysis and improvement of a biometric-based multi-server authentication and key agreement scheme," *PLoS One*, vol. 11, no. 2, Article ID e0149173, 2016.

[28] A. G. Reddy, E.-J. Yoon, A. K. Das, V. Odelu, and K.-Y. Yoo, "Design of mutually authenticated key agreement protocol resistant to impersonation attacks for multi-server environment," *IEEE access*, vol. 5, pp. 3622–3639, 2017.

[29] S. Chatterjee, S. Roy, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment," *IEEE Computer Architecture Letters*, vol. 15, no. 05, pp. 824–839, 2018.

[30] H. Qiao, X. Dong, and Y. Shen, "Authenticated key agreement scheme with strong anonymity for Multi-Server environment in TMIS," *Journal of Medical Systems*, vol. 43, no. 11, pp. 1–13, 2019.

[31] C.-C. Lee, D.-C. Lou, C.-T. Li, and C.-W. Hsu, "An extended chaotic-maps-based protocol with key agreement for multi-server environments," *Nonlinear Dynamics*, vol. 76, no. 1, pp. 853–866, 2014.

[32] S. Jian, "An authenticated key agreement protocol based on extended chaotic maps," *Acta Physica Sinica*, vol. 63, no. 5, 2014.

[33] T. Limbasiya and S. K. Sahay, "Secure and Energy-Efficient Key-Agreement Protocol for Multi-Server architecture," in *Proceedings of the International Conference on Secure Knowledge Management in Artificial Intelligence Era*, pp. 82–97, Springer, Singapore, 2019.

[34] T. Limbasiya, S. K. Sahay, and B. Sridharan, "Privacy-preserving mutual authentication and key agreement scheme for multi-server healthcare system," *Information Systems Frontiers*, vol. 23, pp. 1–14, 2021.

[35] P. K. Roy and A. Bhattacharya, "A group key-based lightweight Mutual Authentication and Key Agreement (MAKA) protocol for multi-server environment," *The Journal of Supercomputing*, vol. 78, pp. 1–28, 2021.

[36] S. Shamshad, M. F. Ayub, and K. Mahmood, "An identity-based authentication protocol for the telecare medical information system (TMIS) using a physically unclonable function," *IEEE Systems Journal*, 2021.

[37] Y. Yang, L. Zhang, Y. Zhao, K.-K. R. Choo, and Y. Zhang, *Privacy-Preserving Aggregation-Authentication Scheme for Safety Warning System in Fog-Cloud Based VANET*, IEEE Transactions on Information Forensics and Security, USA, 2022.

[38] H. S. Ali and R. Sridevi, *Credential-Based Authentication Mechanism for IoT Devices in Fog-Cloud Computing*, Springer, Singapore, pp. 307–318, 2022.

[39] L. Loffi, C. M. Westphall, L. D. Grüdtner, and C. B. Westphall, "Mutual authentication with multi-factor in IoT-Fog-Cloud environment," *Journal of Network and Computer Applications*, vol. 176, Article ID 102932, 2021.

[40] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 79, no. 9, pp. 1338–1354, 1996.

[41] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures for delegating signing operation," in *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pp. 48–57, New Delhi India, March 1996.

[42] B. Lee, H. Kim, and K. Kim, "Strong proxy signature and its applications," in *Proceedings of the 2001 symposium on cryptography and information security (SCIS'01)*, vol. 2, p. 2, Oiso, Japan, January 2001.

[43] B. Lee, H. Kim, and K. Kim, "Secure mobile Agent Using strong Non-designated Proxy signature," in *Proceedings of the Australasian Conference on Information Security and Privacy*, Springer, Berlin, Heidelberg, pp. 474–486, 2001.

[44] N. R. Sunitha and B. B. Amberker, "Forward-Secure Proxy Signature Scheme for Multiple Proxy Signers Using Bellare-Miner Scheme with Proxy Revocation," in *Proceedings of the 2008 the Fourth International Conference on Information Assurance and Security*, pp. 73–78, IEEE, 2008.

[45] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: how to generate strong keys from biometrics and other noisy data," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 523–540, 2004.

[46] C. J. F Cremers, *Scyther: Semantics and Verification of Security protocols*, Eindhoven university of Technology, Eindhoven, Netherlands, 2006.

[47] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," in *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 426, no. 1871, pp. 233–271, London, UK, 1989.

[48] E. Barker, W. Barker, W. Burr et al., *SP 800-57. Recommendation for Key Management, Part 1: General (Revised 4),* National Institute of Standards &Technology, Washington, DC, USA, 2016.

[49] E. Barker, L. Chen, S. Keller, A. Roginsky, A. Vassilev, and R. Davis, *Sp 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revision 2),* National Institute of Standards &Technology, Washington, DC, USA, 2013.