An Identity Management and Authentication Scheme Based on Redactable Blockchain for Mobile Networks

Jie Xu, Kaiping Xue, *Senior Member, IEEE*, Hangyu Tian, *Student Member, IEEE*, Jianan Hong, David S. L. Wei, *Senior Member, IEEE*, and Peilin Hong

***Abstract*—More and more users are eager to obtain more comprehensive network services without revealing their private information. Traditionally, in order to access a network, a user is authorized with an identity and corresponding keys, which are generated and managed by the network operator. All users' personally identifying information are centralized stored by the network operator. However, this approach makes users lose the control of their personally identifying information. Users are concerned about who can access these sensitive data and whether they have been compromised. In this paper, we propose a blockchain-based identity management and authentication scheme for mobile networks, where users' identifying information are controlled by the users themselves. Our scheme let users generate their self-sovereign identities (SSIs) and corresponding public keys and private keys. The private key used to authenticate the user's identifying information is only known to the user. We use blockchain to record SSIs and public keys of legitimate user, and adopt chameleon hash to delete illegal users' information on the blockchain, while keeping the block head unchanged. Furthermore, other service providers can obtain the user's SSI and public key and authenticate users by querying the blockchain. Experimental results confirm that our scheme can greatly reduce the revocation overhead and communication overhead.**

***Index Terms*—Identity management, mutual authentication, chameleon hash, redactable blockchain.**

I. Introduction

With advances in wireless communication technology, users are eager to obtain more efficient and comprehensive mobile network services. In order to prevent users from illegally accessing or attacking the network, identity management and access authentication play a critical role in mobile network security. In general, users can only use their credentials, such as identity cards, to register with their local network operator to obtain their unique International Mobile Subscriber Identification Number (IMSI) and the corresponding symmetric keys for accessing network [1].

However, there are many shortcomings in existing service-centric wireless mobile network architecture. First of all, mutual authentication between users and operators is based on symmetric encryption system, which means that users can only be authenticated by their local network operators through symmetric key. In order to authenticate users, the local network operator has to store all the authenticated users' symmetric keys in its Authentication Center (AuC) [2], which is a database in its core network. Each time a user accesses the mobile network, the user's authentication request needs to be sent to the core network. Centralized authentication leads to a single point bottleneck problems. The security of the operator's AuC is critical, and once it is compromised, it may affect millions of users of the network. Secondly, in the service-centric wireless mobile network architecture, if users want to access other network services, they must rely on local operators to provide their own information to other service providers. Thus, users have no control over how much their information is exposed. Users' identity information is very sensitive and needs to be strictly controlled [3]. Operator's coarse-grained exposure of the user's information for commercial benefit or other reasons may cause serious damage of privacy violation to the user. Even worse, users cannot effortlessly change their network operators without giving up their original digital identities. In other words, once a user changes his or her network provider, he or she must give up the personal information in other applications associated with the original identity. As a result, users have less control over their identity, the relationship between operators and users is difficult to change, and market competition decreases.

Self-Sovereign Identity (SSI) can be seen as an identity management model where each identity is fully owned, controlled and managed by the entity to which the identity belongs [4]. Unlike traditional identity management schemes, such as isolated identity model or federated identity model, each user can not only control the secure storage and access of personally identifying information, but also add or delete his/her personally identifying information. Therefore, this kind

Manuscript received December 15, 2019; revised February 21, 2020; accepted March 27, 2020. Date of publication April 8, 2020; date of current version June 18, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61972371 and in part by the Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant 2016394. The review of this article was coordinated by Prof. L. Zhu. *(Corresponding author: Kaiping Xue.)*

Jie Xu, Kaiping Xue, Hangyu Tian, and Peilin Hong are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China (e-mail: xj47@mail.ustc.edu.cn; kpxue@ustc.edu.cn; thy2014@mail.ustc.edu.cn; plhong@ustc.edu.cn).

Jianan Hong is with the Huawei Shanghai Research Institute, Shanghai 201206, China (e-mail: hongjianan@huawei.com).

David S. L. Wei is with the Department of Computer and Information Science, Fordham University, Bronx, NY 10458 USA (e-mail: wei@cis.fordham.edu).

Digital Object Identifier 10.1109/TVT.2020.2986041

of identity management infrastructure does not need to trust any central authority and can exist in a decentralized environment that does not belong to or be controlled by a centralized authority.

Meanwhile, blockchain [5] can be treated as a public, digitized, and distributed ledger built on peer-to-peer network, which has been introduced and applied many network scenarios [6]–[9]. In blockchain system, data generated by participating entities are published as a transaction, and transactions are packaged into a block. Blocks are added to the blockchain by miners in chronological order. It is worth noting that miners who add data are independent individuals, and there are no centralized third-party authorities in the blockchain. All participating entities store the blockchain and periodically update the blockchain. It is easy to share information among multiple participating entities. Blockchain helps with implementing a system with no need of a trusted party such as a central Certification Authority (CA). Therefore, blockchain is an ideal mechanism to serve as a decentralized public publishing and query platform for users' self-sovereign identities management.

Motivated by the above observations, we introduce a blockchain-based self-sovereign identity management and authentication scheme to enable users to manage personally identifying information. In our user-centric identity management, users generate their SSI and obtain personally identifying information associated with the SSI from distributed trusted entities. Network operators authenticate users through SSIs and their personally identifying information (recorded as $claim$). The SSIs and corresponding public keys of legitimate users are added to blockchain by network operators for others to query. To defend illegal use of network services by users, user dynamic revocation is an essential part of user identity management schemes. However, a trivial solution by which network operators maintain and update the revocation list requires heavy storage overhead. In order to reduce storage overhead, network operators utilize chameleon hash [10] to remove illegal users' information on blockchain instead of maintaining revocation lists to implement dynamic user revocation.

The main contributions of this paper can be summarized as follows:

1) We propose a self-sovereign identity management scheme for wireless mobile networks. By introducing self-sovereign identity, even if a user changes his/her operator, he/she can still retain control over his/her identity and personally identifying information.
2) We design a lightweight authentication protocol based on blockchain between users and network operators or service providers. Network operator or service provider can authenticate user and negotiate session key by querying blockchain. There is no need for service provider to maintain a database to save users' keys; and the user can authenticate with service provider without creating a redundant identity.
3) We further provide an efficient and fine-grained dynamic user revocation method by utilizing the chameleon hash. The network operator can revoke users at any time by modifying users' register information on blockchain with no need of maintaining the revocation list, thereby reducing the authentication delay, and storage overhead.

The rest of this paper is organized as follows. We review related work in Section II, and introduce the system model, threat model and design goals in Section III. Then, we describe the detailed construction of our proposed scheme in Section V. Security analysis and performance evaluation are presented in Section VI and Section VII, respectively. Finally, we conclude this paper in Section VIII.

II. Related Work

In this section, we discuss the related work in terms of identity management and traditional authentication for wireless mobile networks.

A. *Identity Management Scheme*

Centralized infrastructures are the most common identity management [11]–[13], where users' personally identifying information are controlled by an organization rather than the user himself/herself. It means that the organization may disclose the information of the user at any time. Once an organization‘s database is compromised, the adversary can enter into each user's account. In addition, the centralized system is a closed system, making it difficult to transfer users' personally identifying information from one application to another [14]. In order to solve the problem of centralization, some studies have proposed federated identity management [15], [16]. Users are allowed to log into multiple services within the federated domain using one identity. Although federated identity management achieve identity portability to some extent, the identity of users is still controlled and managed by federated service providers. Meanwhile, there have been quite a few proposed schemes contributing to meet the user privacy protection requirements [17]–[19]. They focus on user oriented paradigm, called user-centric identity management, enabling users to selectively authorize personal data under various conditions and reveal the credentials presented in response to authentication requests. In the literature [17], Singh *et al.* proposed a user-centric model, which is recorded as a privacy-aware Personal Data Storage (PDS). By active learning, PDS can automatically make privacy-aware decisions on third-party access requests based on user preferences.

However, the user-centric model does not enable users to fully control and manage their personally identifying information, while self-sovereign identity management achieves it using blockchain technology. Alboaie *et al.* [20] introduced a private data system, in which all private data are encrypted and stored in nodes throughout the Internet. The user has full control over the decryption key and can share and revoke access to private data at any time. After that, Othman *et al.* [21] combined a decentralized SSI ecosystem with Biometric Open Protocol Standard (BOPS). In [21], any entity can access the user's personally identifying information only after successful biometric authentication. In 2019, Ferdous *et al.* [22] aimed at giving a comprehensive and rigorous concept of self-sovereign identity. They also demonstrated several envisaged self-sovereign identity management processes. However, it is just a rough idea and there is no detailed design. In addition, there are also several projects that combine distributed ledger technology (DLT) and SSI, such as uPort [23], Sovrin [24] and ShoCard [25].

Although [20]–[25] can solve the problem of users' losing the control of their identifying information [11]–[19], these schemes are unable to let the network operator revoke users whenever the network operator controls users access to the network. In this paper, we introduce a self-sovereign identity management scheme for mobile networks. It allows the network operator to revoke users, while the user's identity is still valid in other service application even if the user is revoked by the network operator.

B. Authentication Schemes for Wireless Mobile Networks

Due to the open access characteristics of wireless mobile networks, it is important to perform network access control to thwart malicious attacks. Although public key infrastructure (PKI) has been widely used to implement mutual authentication between users and servers, it is still a big challenge for PKI to establish a trust authority to manage all certificates. Recent research [26], [27] indicates that Identity-Based Cryptography (IBC) authentication system can eliminate heavy certificate management burden based on PKI. While IBC-based works show great advantages for application scenarios involving devices with restricted resources, these work assume that a trusted centralized Private Key Generator (PKG) exists. This means that all participants are required to fully trust the centralized PKG to generate private keys, which may lead to single-point bottleneck. In [28] and [29], authors propose to authenticate users by using group signatures, but these schemes may not be suitable for some IoT devices with limited computing resource because of the high computational overhead of group signatures.

Nowadays, in order to provide users with a wider range of network services, many studies have done a great deal of work on roaming authentication, such as [29]–[32]. In these schemes, a user only have one identity and its corresponding key generated by the local domain server for network access. When a user roams to a foreign network, the user sends access network requests to the foreign operator, and the foreign operator forwards the user‘s authentication request to user's local domain server for authentication. The local domain server authenticates the user and returns the authentication result to the foreign domain server [29], [30]. However, since authentication of the user requires the participation of the user's local server, authentication delay caused by communication between the foreign server and the local server is inevitable. Thus, some research [31], [32] are committed to two-party roaming protocols, so that user authentication does not require local network operators to be online in real time. Unfortunately, in these schemes, the foreign domain server cannot discover the users revoked by the local domain in time, resulting in illegal access.

In addition, it is worth noting that all of the above authentication schemes must involve the central server. This means that once the central server crashes, the entire system breaks down. Therefore, in this paper, we propose a distributed authentication scheme, which can authenticate users at the edge of the network, thus eliminating single point bottleneck and enhancing robustness.

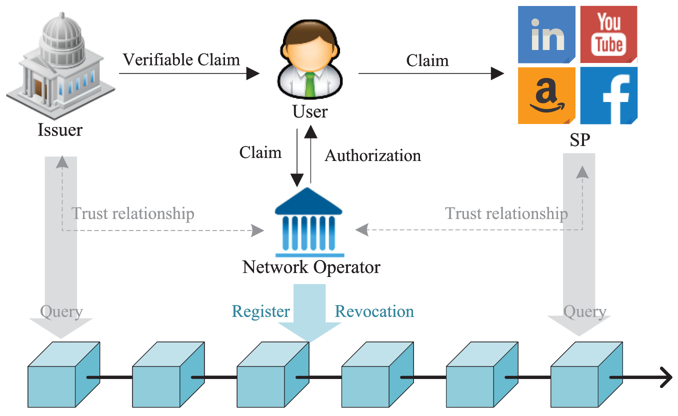


Fig. 1. System model.

III. System Model, Threat Model, and Design Goals

In this section, we introduce the system model, threat model, and design goals of our proposed scheme.

A. System Model

As depicted in Fig. 1, there are mainly five components, which are described in detail as follows:

- *User*: The user is the owner of an SSI. A user generates SSI and its corresponding public key, uniquely and completely control and manage SSI. A user can have multiple SSIs at the same time, and there is no connection between SSIs. The user stores claims related to his/her SSI locally, and presents them to the service provider as needed.
- *Issuer*: The issuers are distributed trusted entities such as schools, banks, companies, etc. They issue verifiable claims for certain attributes of the SSI owner. These verifiable claims include the issuer's signature for verification by others.
- *Network operator*: It verifies users' verifiable claims and determines whether users can access the network. All participating network operators form a consortium to jointly maintain the blockchain. Network operators can also revoke users by modifying transactions on blockchain. There is an existing trust relationship between the verifier and the issuer.
- *Service provider (SP)*: It obtains the user's SSI and corresponding public key by querying the blockchain. Users can prove their own SSIs to SPs by signing with their private keys.
- *Blockchain*: It is a consortium blockchain maintained by network operators for publishing users' SSIs and public keys. Any entity can read the information on the blockchain. The transactions in a block constitute a merkle hash tree, where the first layer of hash is a chameleon hash. Network operators can modify transactions while keeping block headers unchanged.

B. Threat Model

We assume that there may be passive attack and active attack in wireless mobile networks. Active adversaries may involve modifying messages or disrupting communications between user and network operator or service provider. Typical active attacks include replay attacks, injection attacks, DoS attacks, and DDoS attacks. An active adversary may be a user, performing various attacks in order to illegally occupy network resources. Unlike active attacks, passive adversaries do not actively interrupt user communications. The passive adversaries may get user's private information by eavesdropping on user messages or by analyzing user traffic.

In addition, we assume that there are $3f + 1$ network operators in the consortium to maintain the blockchain, and there are no more than f malicious nodes in the consortium.

C. Design Goals

We aim to achieve a secure and efficient identity management and authentication in wireless mobile networks. Specifically, our design goals include the following:

- *Identity Security:* For confidentiality, personally identifying information interactions must be transmitted over a secure channel. Only users themselves have the private key corresponding to their own SSI. No adversary can impersonate a user to apply for claims or present claims. Identity security also includes the availability of identity. Even if a user's SSI is revoked by a network operator, the user still controls his or her SSI and related claims, which means that the user can still use his/her SSI in other applications.
- *Mutual Authentication:* The authentication scheme includes mutual authentication between users and network operators, and also between users and SPs. Network operators and SPs only provide services to users who have successfully authenticated. Meanwhile, users only trust the services of network operators and SPs, which are successfully authenticated.
- *Session Key Security:* After successful authentication, a network operator and a user negotiate a session key. The session keys satisfy forward security and backward security, that is, even if the current session key is compromised, the previous or subsequent session keys remain secret.
- *Portability:* An SSI can be used in multiple platforms at the same time. Moreover, users can start or quit using SSI on the platforms at any time. Even if an SSI of the user is revoked in one platform, it does not affect the SSI used in other platforms.
- *Scalability:* There is no single point bottleneck problem. The system can support large-scale user identity management and handle a large number of identity authentication requests at the same time.

IV. Preliminaries

In this section, we illustrate background knowledge used in this paper, including definition of discrete logarithm and its security assumptions, chameleon hash algorithm, and description of verifiable claim.

A. Discrete Logarithm

Definition 1 (Discrete Logarithm Problem (DLP)): Let G be a cyclic multiplicative group generated by g with the prime order q. For any Probabilistic Polynomial Time (PPT) adversary $\mathcal{A}$, the probability of finding $x \in Z_q^*$ to satisfy the equation $y = g^x$ is negligible, when giving the parameters g and y.

B. Chameleon Hash

Chameleon hash was first proposed by Krawczyk and Rabin in 2000 [10]. It is an important function of chameleon signature, which was originally designed to not allow the recipient to disclose the contents of the signed information to anyone without the consent of the signer. Chameleon hash can be viewed as a cryptographic hash function that contains a trapdoor. If there is no trapdoor, it is collision-resistant, but is collision tractable with the knowledge of the trapdoor information. Later, Ateniese *et al.* [33] proposed an approach that uses chameleon hash to replace the hash function in block head to make blockchain redactable. After that, Derler *et al.* [34] further uses policy-based chameleon hash to implement transaction-level rewriting, thereby supporting the fine-grained and controllable modification.

Next, we give the definition of chameleon hash, and then show the detailed chameleon hash generation process in Algorithm 1, which is according to the work done by Ateniese *et al.* [33].

Definition 2 (Chameleon-Hash): The chameleon hash CH consists of four algorithms $(Gen, Hash, Ver, Col)$. The algorithm details are as follows:

- $(hk, tk) \leftarrow \mathbf{CH}.\mathbf{Gen}(1^\kappa)$: The key generation algorithm takes the security parameter $\kappa \in \mathbb{N}$ as input, and outputs a public hash key hk and a secret trapdoor key tk.
- $(h) \leftarrow \mathbf{CH}.\mathbf{Hash}(hk, m; r, s)$: The chameleon hash algorithm takes the hash key hk, a message m, and (r, s), where r and s denote the random number used to generate the hash value as input, and outputs a chameleon hash h.
- $d \leftarrow \mathbf{CH}.\mathbf{Ver}(hk, m, (h, r, s))$: The verification algorithm takes the hash key hk, a message m, and (h, r, s) as input, and returns $d = 1$ if and only if *CH.Hash*$(hk, m; r, s) = h$. Otherwise, d equals 0.
- $(r,' s') \leftarrow \mathbf{CH}.\mathbf{Col}(tk, h, m')$: The collision finding algorithm takes the trapdoor key tk, a chameleon hash h, and a new message m' as input, and returns a new pair of $(r,' s')$ such that h=*CH.Hash*$(hk, m'; r,' s')$. If (tk, h, m') is invalid, then the algorithm returns $NULL$.

C. Claim

Definition 3 (Claim): Claims are assertion that describe various attributes related to an entity.

Definition 4 (Verifiable Claim): Verifiable Claims are claims made by a third party about another entity. Verifiable claims can be verified by the public key associated with the issuer's SSI, and cannot be denied. An entity can have multiple verifiable claims at the same time, and stored them locally.

Algorithm 1: Chameleon Hash.

```
    Input: Message $m$, public hash key $y$, and secret
        trapdoor key $x$;
    Output: Chameleon hash $h$;
1:  Select prime $p$, $q$ where $p = 2q + 1$;
2:  Select prime $g$, which is a generator for the subgroup
    of quadratic residues $\mathbb{QR}_p$ of $\mathbb{Z}_p^*$;
3:  Select random value $r$ and $s$, where $r, s \in \mathbb{Z}_q$;
4:  Compute chameleon Hash value
    $h = r - (y^{H(m\|r)} \cdot g^s \mod p) \mod q$, where
    $H : \{0,1\}^* \to \mathbb{Z}_q$ is a standard-collision resistant
    hash function;
5:  return $(h, r, s)$;
```

Algorithm 2: Operator Initialization.

```
    Input: Security parameter $\kappa$;
    Output: Secret trapdoor key $x$ and public hash key $y$;
1:  Select prime $p$, $q$, where $p = 2q + 1$;
2:  Select $g$, which is a generator for the subgroup of
    quadratic residues $\mathbb{QR}_p$ of $\mathbb{Z}_p^*$;
3:  Select a random value $x \in [1, q - 1]$ as the secret
    trapdoor key;
4:  Set the public hash key $y = g^x$;
5:  return $(x, y)$;
```

V. Our Proposed Scheme

In this section, we first give the overview of our proposed self-sovereign identity management and authentication scheme. In the following, we provide a detailed description of our scheme, which mainly consists of five phases: Initialization Phase, Consensus Phase, User Authentication Phase, Dynamic Revocation, and Payment Phases.

A. Overview

In self-sovereign identity management, users generate their self-sovereign identities ID and the corresponding public and private keys (pk, sk). Before a user requests service from a network operator, the user should sends (ID, pk) to the issuer trusted by the network operator for verifiable claims. It is noteworthy that a user can generate several different (ID, pk, sk) to be authenticated by different issuers to obtain multiple verifiable claims. In order to access the network, the user provides the verifiable claim to the network operator for authorization. The network operator packages authorized users' SSI and public keys into a transaction. The transaction is eventually added to the consortium blockchain maintained by network operators. After that, not only the network operator but also any third party, such as a service provider, can use the user's public key obtained by querying the blockchain to authenticate the user. After authentication, the network operator and the user negotiate for session keys by sharing secret parameters to ensure the privacy of subsequent session information, and so does between the service provider and the user.

If a user is detected illegally accessing network, the network operator who allows the user to access network can revoke the user. More precisely, the network operator first generates a new transaction that does not contain the revoked user‘s registration information to replace the original transaction that contained the revoked user's registration information. Then, the network operator generates a chameleon hash of the new transaction, broadcasting the new transaction and the new chameleon hash parameters to other network operators in the consortium. After successfully verifying the chameleon hash parameters, other network operators replace the original stored transaction with the new one. Therefore, users on blockchain are now legitimate users, there is no need to maintain a revocation list.

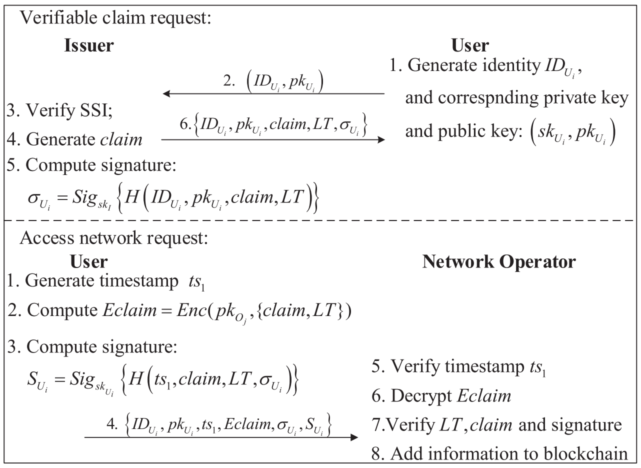

Verifiable claim request:
Issuer
User
1. Generate identity ID_{U_i}, and correspnding private key and public key: (sk_{U_i}, pk_{U_i})
2. (ID_{U_i}, pk_{U_i})
3. Verify SSI;
4. Generate *claim*
5. Compute signature:
$\sigma_{U_i} = Sig_{sk_I}\{H(ID_{U_i}, pk_{U_i}, claim, LT)\}$
6. $\{ID_{U_i}, pk_{U_i}, claim, LT, \sigma_{U_i}\}$
Access network request:
User
Network Operator
1. Generate timestamp ts_1
2. Compute $Eclaim = Enc(pk_{O_j}, \{claim, LT\})$
3. Compute signature:
$S_{U_i} = Sig_{sk_{U_i}}\{H(ts_1, claim, LT, \sigma_{U_i})\}$
4. $\{ID_{U_i}, pk_{U_i}, ts_1, Eclaim, \sigma_{U_i}, S_{U_i}\}$
5. Verify timestamp ts_1
6. Decrypt *Eclaim*
7.Verify LT, *claim* and signature
8. Add information to blockchain

Fig. 2. User initialization.

B. Details of Our Proposed Scheme

Here we give a detailed description of our proposed system, which can be divided into five phases: 1) Initialization phase, 2) Consensus phase, 3) User authentication phase, 4) Dynamic user revocation phase, and 5) Payment phase.

1) Initialization Phase: The initialization phase can be divided into two parts. One part is the network operator initialization, which generates the network operator's secret trapdoor key and public hash key through Algorithm 2.

The other part is user initialization part, which is illustrated in Fig. 2. The detailed steps are described as follows.

1) User U_i generates his/her identity ID_{U_i} and the corresponding public key pk_{U_i} and private key sk_{U_i} by using a public key algorithm such as RSA. Then, the user sends (ID_{U_i}, pk_{U_i}) to the issuer through a secure channel.
2) Upon receiving the user's message, the issuer I firstly verifies (ID_{U_i}, pk_{U_i}). If ID_{U_i} has already been registered or it is invalid, the issuer rejects the request. Otherwise, the issuer generates a verifiable claim $claim$ and its signature $\sigma_{U_i} = Sig_{sk_I}\{H(ID_{U_i}, pk_{U_i}, claim, LT)\}$ for U_i, where LT is the lifetime for the verifiable claim. Then, I sends $\{ID_{U_i}, pk_{U_i}, claim, LT, \sigma_{U_i}\}$ to U_i through a secure channel.

Algorithm 3: Transactions Generation.

```
Input: All the authorized users‘ identities, public
    keys, operator's public hash key $y$;
Output: Transaction $tx_{O_j}$;
1: foreach authorized user $ID_{U_i}$
2:     Set $\lambda_{ij} = \{ID_{U_i}^{O_j}, pk_{U_i}\}$;
3: end
4: Set $w_j = \{\lambda_{1j}, \ldots, \lambda_{ij}\}$;
5: Set message $m = \{ID_{O_j}, w_j, Sig_{sk_j}\{H(w_j)\}\}$;
6: Select random value $r$ and $s$, where $r, s \in \mathbb{Z}_q$;
7: Compute chameleon hash value $h = r - (y^{H(m||r)} \cdot g^s \mod p) \mod q$, where $H : \{0,1\}^* \to \mathbb{Z}_q$ is a
   standard-collision resistant hash function;
8: Set $tx_{O_j} = \{m, h\}$;
9: return $t_{O_j}$;
```

3) Upon receiving σ_{U_i} from the issuer, the user can send network access requests to network operator O_j. Firstly, the user generates a timestamp ts_1, and then computes $Eclaim = Enc(pk_{O_j}, \{claim, LT\})$ and signature $S_{U_i} = Sig_{sk_{U_i}}\{H(ts_1, claim, LT, \sigma_{U_i})\}$. Finally, the user sends $\{ID_{U_i}, pk_{U_i}, ts_1, Eclaim, \sigma_{U_i}, S_{U_i}\}$ to O_j.

4) Upon receiving the message from U_i, O_j verifies whether the timestamp ts_1 is within the allowed range compared to current time. If not, O_j rejects the request; otherwise, O_j continues to check whether the lifetime LT is within the allowed time. If not, O_j stops the session. Otherwise, O_j decrypts $Eclaim$ to get the $claim$, and verifies the signature S_{U_i}. If the signature is valid, O_j authorizes U_i to access the network.

Similarly, a user can generate several SSIs and corresponding public and private keys to obtain verifiable claims of different issuers and store them locally. Besides, a user can use one SSI to register with multiple network operators for authorization at the same time. Thus, the user can hand over to different networks according to external network environment without changing his/her identity.

2) Consensus Phase: After a network operator authorizes a user, the network operator adds the user's registration information to blockchain, which is a consortium blockchain maintained by network operators. To improve efficiency, each network operator periodically packs the information of its authorized users in a time slot into a transaction. Algorithm 3 illustrates the detail of the process of transactions generation. After the transaction is generated, the network operator broadcasts $\{t_{O_j}\}$ to other network operators in the consortium.

Blockchain is a distributed system that does not depend on a central authority. Therefore, an algorithm is needed to enable network operators in the consortium to reach a consensus. In our proposed scheme, the consensus of consortium blockchain is Practical Byzantine Fault Tolerance (PBFT) [35]. We assume that there are a total of $3f+1$ network operators in the consortium. There is only one leader in each time period, and the leader is rotated by network operators.

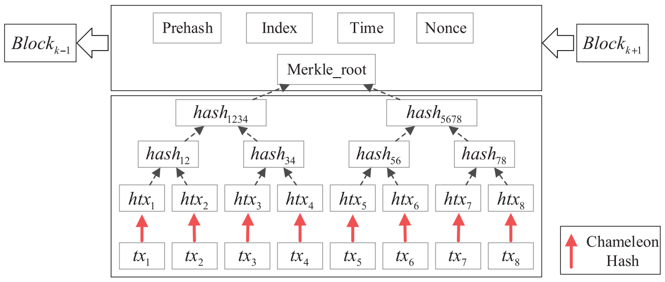


Fig. 3. Structure of block with chameleon hash.

Every once in a while, network operators verify the validity of the received transaction with the corresponding (r, s). Then, each network operator multicasts the verification results of the transactions. After the pre-prepare, prepare, commit, and reply phase of the PBFT consensus, legal transactions are added to the consortium blockchain maintained by all network operators. It is important to highlight that the hash of transaction is chameleon hash. As it can be seen from Fig. 3, the first level of merkle tree is the chameleon hash value h, which are contained in transactions.

3) User Access Phase: After a user's SSI and public key are added to blockchain, the user can authenticate with a network operator or a service provider to negotiate a session key to encrypt communication message. The detailed process for the user U_i to access the network operator O_j's network is as follows, and the process of the user U_i requiring the service from SP is similar.

1) User U_i first generates random value r_1, timestamp ts_2, and computes signature $\alpha_{U_i} = Sig_{U_i}\{H(r_1, ts_2)\}$. Then, U_i sends $\{ID_{U_i}, r_1, ts_2, \alpha_{U_i}\}$ to network operator O_j.

2) Upon receiving the message from U_i, O_j first verifies the timestamp. If ts_2 is not within the allowed range compared to current time, O_j rejects the access request; otherwise, O_j searches for pk_{U_i} on blockchain with ID_{U_i}. If there is no pk_{U_i}, O_j rejects user's access request. Otherwise, O_j verifies the signature α_{U_i} with pk_{U_i}. If α_{U_i} is invalid, O_j stops the session; otherwise, O_j randomly generates the $seed$ of session key for U_i, and timestamp ts_3. Then, O_j computes the signature $\beta_{O_j} = Sig_{O_j}\{H(ts_3, Sig_{O_j}(r_1), Enc(pk_{U_i}, seed))\}$ and the session key $SK_{ij} = H(ID_{U_i}||ID_{O_j}||ts_2||ts_3||seed)$. Finally, O_j sends $\{ts_3, Sig_{O_j}(r_1), Enc(pk_{U_i}, seed), \beta_{O_j}\}$ to U_i.

3) Upon receiving the message from O_j, U_i checks if ts_3 is within the allowed range compared to the current time. If so, U_i queries the network operator's public key on blockchain, and then verifies the received signature β_{O_j}. If β_{O_j} is valid, U_i decrypts the $Enc(pk_{U_i}, seed)$ to get the $seed$, and computes the session key $SK_{ij} = H(ID_{U_i}||ID_{O_j}||ts_2||ts_3||seed)$; otherwise, U_i restarts the access request.

It should be pointed out that if any other service provider or foreign network operator trusts the local network operator O_j to verify the user's claim, the service provider or foreign network operator can trust the user on the blockchain added by O_j and not verify the user's claim again. Therefore, they can search for

Algorithm 4: Chameleon Collision.

```
Input: Updated message $m'$, chameleon hash $h$,
    operator's secret trapdoor key $x$;
Output: updated $(r,' s')$;
1: Select a random value $k \in [1, q-1]$;
2: Compute updated $r' = h + (g^k \mod p) \mod q$;
3: Compute updated $s' = k - H(m'||r') \cdot x \mod q$;
4: return $(r,' s')$;
```

pk_{U_i} from the blockchain to authenticate user U_i, and support the user accessing service. As a result, the user can have access to multiple services with one identity ID_{U_i} and the corresponding private key.

4) Dynamic Revocation: Considering that some users may illegally access network, and network operators need to revoke these illegal users and prevent them from accessing network. In general, each network operator needs to maintain and periodically update a revocation list. However, it might bring heavy storage overhead and communication overhead. In our scheme, we utilize chameleon hash to delete illegal users' registration information stored on blockchain instead of revocation list. If there is no public key of a user on the blockchain, the user can be considered as an illegal network user. In order to delete the registration information on the blockchain, the network operator who authorizes users to access the network, generates a new transaction that does not contain illegal user's registration information in place of the transaction containing the legitimate user's registration information on the blockchain. After the new transaction is verified by other network operators in network operator consortium, it replaces the original transaction. The specific transaction replacement steps are as follows:

1) If O_j needs to revoke user U_i, whose registration information is contained in tx_{O_j}, O_j first generates a new message $m' = \{ID_{O_j}, \lambda_{1j}, \ldots, \lambda_{i-1j}, Sig_{sk_j}\{H(w'_j)\}\}$, where $\lambda_{kj} = \{ID_{U_k}, pk_{U_k}\}$, $w'_j = \{ID_{O_j}, \lambda_{1j}, \ldots, \lambda_{i-1j}\}$. m' is the same as m except that m' does not contain the registration information of U_i.
2) O_j generates chameleon hash collision with m' and its private trapdoor key. Algorithm 4 illustrates the process of the chameleon hash collision.
3) After executing Algorithm 4, O_j can obtain r' and s'. Then, O_j generates a new transaction $tx'_{O_j} = \{m,' h\}$. O_j broadcasts $\{tx_{O_j},' r,' s'\}$ to other operators.
4) After other network operators received the new transaction, they first verify whether h is equal to $r' - (y^{H(m'||r')} \cdot g^{s'} \mod p) \mod q$. If so, other network operators store the new transactions tx'_{O_j} and delete the original transaction tx_{O_j}.

5) Payment Phase: Cryptocurrency can provide users and network operators with more convenient and secure transaction methods of trading. However, the processing speed of cryptocurrency transactions is limited and transaction fees is overwhelming even for small value transactions. Instead, micropayment channel [36] is the most promising solution to significantly reduce transaction time and save transaction costs. As shown in

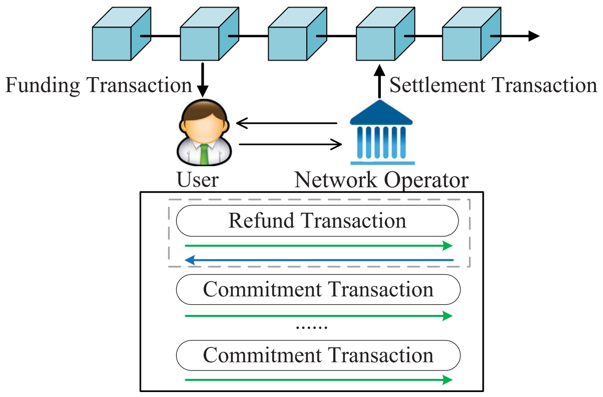


Fig. 4. The structure of Userchain.

Fig. 4, we propose to utilize micropayment channel to conduct secure payments offchain. The detail of micropayment channel can be found in [36]. The micropayment channel scheme mainly consists of three steps: establish channel, update channel, and close channel.

1) User U_i first creates a funding transaction on bitcoin blockchain. U_i pays an amount of bitcoins b_0 to a P2SH output in funding transaction, whose $2-of-2$ multi-signature redeem script requires signatures from both U_i and O_j. The user sends the transaction to O_j, and O_j holds the bitcoin hostage. User U_i creates a refund transaction, whose input is the input of the funding transaction, and the output is the address of the user. User U_i sends the refund transaction to SP for signature. If O_j checks that the refund transaction is valid, then O_j signs the refund transaction and sends it to U_i. Finally, U_i broadcasts funding transaction and refund transaction. The refund transaction is invalid until the time-lock in the transaction is unlocked. Thus, the micropayment channel is established.
2) If a user needs to pay b bitcoins to O_j every minute as service fee, then the user generates a commitment transaction every minute. The input of the commitment transaction generated by the user for the first time is the input of the funding transaction, that is b_0, and the output is b for O_j and $b_0 - b$ for U_i. U_i signs the first commitment transaction and sends it to O_j. After verifying the commitment transaction, O_j continues to provide service. The input of the commitment transaction generated by the user for the second time is b_0, and the output is $2 \cdot b$ for O_j and $b_0 - 2 \cdot b$ for the user. The user signs the second commitment transaction and sends it to O_j. After verifying the commitment transaction, O_j continues to provide service. So on and so forth, the input of the commitment transaction generated by the user for the kth time is b_0, and the output is $k \cdot b$ for O_j and $b_0 - k \cdot b$ for the user.
3) After k' times service, if $b_0 - (k'+1) \cdot b < 0$, O_j could close the channel. To close the channel, O_j broadcasts the most updated commitment transaction, which is as the settlement transaction. After the miners add the settlement transaction to the blockchain, O_j and the user can get the

output of the settlement transaction. In this way, O_j can get $k' \cdot b$ bitcoins, and user can get back $b_0 - k' \cdot b$ bitcoins.

It is worth noting that the settlement transaction needs to be broadcast before the time lock takes effect. Otherwise, the micropayment channel might close and all the bitcoins may return to the user. As described above, only the funding transaction and settlement transaction are on-chain, and all commitment transactions are off-chain. Therefore, micropayment channel could improve the scalability and reduce the transaction fees while ensuring the fairness.

VI. Security Analysis

In this section, we analyze the security of our proposed scheme according to the design goals described in III-C.

A. Identity Security

It is worth mentioning that in our proposed scheme, user's identity U_i, public key pk_{U_i}, and private key sk_{U_i} are generated by the user him/herself. The issuer authenticates the user and issues $claim$ and $\sigma_{U_i} = Sig_{sk_I}\{H(ID_{U_i}, pk_{U_i}, claim, LT)\}$, which contains (ID_{U_i}, pk_{U_i}) that cannot be tampered with by any adversary. Network operator authorizes the user and adds the user's (ID_{U_i}, pk_{U_i}) to blockchain, so it cannot be tampered with by any adversary. Thus, each legal identity of the user has exactly one pair of authenticated (pk_{U_i}, sk_{U_i}). For any probabilistic polynomial-time (PPT) adversary, he/she cannot derive the private key according to the corresponding public key or encrypted messages. Moreover, the user only uses the private key to decrypt or sign during all authentication processes, and does not reveal the private key, so the user's private key is only controlled by the user. Consequently, our proposed scheme offers identity security.

B. Mutual Authentication

In initialization phase, U_i obtains σ_{U_i}, which contains ID_{U_i} and pk_{U_i} signed by the issuer. The network operator authenticates the user by verifying signature σ_{U_i} and S_{U_i}. Since the user‘s private key is only controlled by the user and we assume that the issuer's private key is secure. Thus, no adversary without sk_I and sk_{U_i} can forge a feasible σ_{U_i} and S_{U_i} to pass validation by the network operator.

In user access phase, the network operator authenticates the user by verifying the signature α_{U_i} with the pk_{U_i}, where pk_{U_i} is the result of querying the blockchain with ID_{U_i}. Since we have proved that sk_{U_i} is only controlled by the user, without sk_{U_i}, no adversary can forge a feasible $\alpha_{U_i} = Sig_{U_i}\{H(r_1, ts_2)\}$ to pass validation by the network operator. The user authenticates the network operator through the challenge-response method. As the network operator's sk_{O_j} is secure, no adversary without sk_{O_j} can forge the network operator's signature on r_1. Therefore, our scheme achieve mutual authentication.

C. Session Key Security

In each session between the user and the network operator, operator O_j randomly generates $seed$ for the session with session key $SK_{ij} = H(ID_{U_i}||ID_{O_j}||ts_2||ts_3||seed)$. Due to the one-way nature of the hash, no adversary can get the $seed$ directly. Adversaries can only get the $seed$ that is encrypted by the user's public key. Thus, no adversary without sk_{U_i} can decrypt $Enc(pk_{U_i}, seed)$ to get $seed$. For this reason, no adversary can directly intercept and decrypt the message to obtain $seed$. Furthermore, in each session, the $seed$ is randomly generated, and thus has no correlation with the seed of the previous session and nor with the seed of the subsequent session. This means that all session keys are independent. Even if an adversary obtains one of the session keys, he/she cannot decrypt the messages of other sessions between the user and the network operator. Overall, our scheme could provide session key security, including the forward security and backward security.

D. Portability

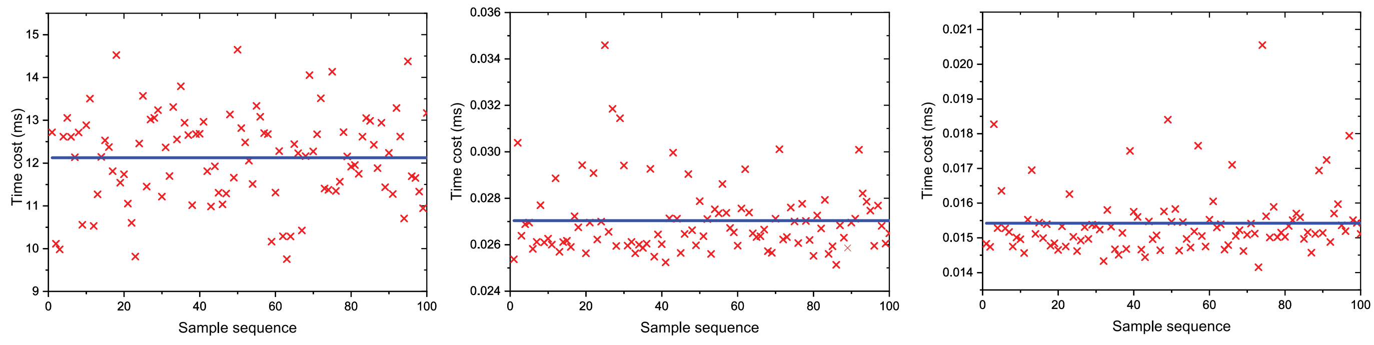
In our scheme, the user generates his/her own identity and private key, and sends it to the issuer for verification to obtain a verifiable claim. The verifiable claim corresponding to the identity is securely stored by the user him/herself, and only the user can prove the control of the verifiable claim with his/her private key. Since user's identity and verifiable claim have nothing to do with network operators and service providers, the user can always manage and use his/her own identity and corresponding verifiable claims to other service providers to obtain services. Therefore, the user's identity is independent of each other in different services. An identity can be used not only in multiple services at the same time, but also in other services after being revoked by one service.

E. Scalability

There is no central authority in the proposed scheme, and users obtain personally identifying information from distributed trusted entities. Thus, there is no single point bottleneck problem, and the system can support large-scale user identity management. In addition, mutual authentication is based on the public key found on the blockchain, so the edge nodes of the network can authenticate users without the involvement of the central server. This approach improves the efficiency of authenticating users and supports handling large-scale authentication requests.

F. Security Comparison

We compare our scheme with four other representative authentication schemes for wireless mobile networks in terms of mutual authentication, privacy preservation, session key security, dynamic revocation, portability, scalability, and centralized trusted authority. As shown in Table I, we conclude that only our scheme does not need a centralized trusted authority, and still achieves portability and scalability.



(a) Chameleon hash keys Generation (b) Chameleon hash (c) Collision Generation

Fig. 5. Time cost for Chameleon Hash Algorithm. (a) Chameleon hash keys Generation. (b) Chameleon hash. (c) Collision generation.

TABLE I
SECURITY COMPARISON WITH RELATED WORK

Schemes	[27]	[28]	[30]	[31]	Our Scheme
Mutual Authentication	YES	YES	YES	YES	YES
Privacy Preservation	YES	YES	YES	YES	YES
Session Key Security	YES	YES	YES	YES	YES
Dynamic revocation	NO	YES	NO	YES	YES
Portability	NO	NO	NO	NO	YES
Scalability	NO	NO	NO	NO	YES
Centralized Trusted Authority	YES	YES	YES	YES	NO

VII. PERFORMANCE EVALUATION

In this section, we conduct experiments to evaluate the effectiveness and feasibility of our scheme. Firstly, we test the performance of chameleon hash. Then, we compare our scheme with the scheme using revocation list in terms of revocation overhead.

A. *Chameleon Hash Algorithm Implementation*

In order to show the performance of chameleon hash more clearly, we tested the computational overhead of various functions of chameleon hash. Specifically, it includes chameleon hash key generation, chameleon hash calculation, and chameleon hash collision generation. Considering that the computational overhead of chameleon hash key generation is equivalent to the computational overhead of the network operator initialization, the methods for generating chameleon hash keys, calculating chameleon hash, and calculating chameleon hash conflict are shown in Algorithm 2, Algorithm 1, and Algorithm 4 respectively.

The experiment environment is a standard 64-bit Linux Mint release 17, Linux version 3.13.0-24-generic with Intel (R) Core (TM) i7-4790, 3.60 GHz processor. All our evaluations were performed by programs in C. Furthermore, the algorithms related to discrete logarithms and big number calculation are realized with OpenSSL 1.0.1f.

In order to ensure the reliability of the experiment, we define the running time of an algorithm as the average time of executing the algorithm 1000 times. Then, we conducted 100 experiments.

TABLE II
KEY PARAMETERS IN CRL

Parameters	ID_U	ID_O	ts	$hash$	$Signature$
Length (Bytes)	32	32	4	32	128

Therefore, we obtained 100 experimental data, and each experimental data represents the average computation overhead of the algorithm running 1000 times. All experimental data obtained are shown in Fig. 5. Fig. 5(a) represents the overhead of the network operator for chameleon hash key generation in initialization phase, and Fig. 5(b) represents the overhead of calculating the chameleon hash when the network operator generates a transaction. Fig. 5(c) indicates the overhead of the network operator for a chameleon hash collision generation to replace a specific transaction. Finally, we have that the average time for the network operator to generate the chameleon hash keys is 12.126 ms, the average time for calculating the chameleon hash is 0.027 ms, and the average time for generating the chameleon hash collision is 0.015 ms.

B. *Comparison of Revocation Overhead*

The most common means of user revocation is Certificate Revocation Lists (CRLs). The main contents of the CRLs include a list of triples (certificate identity, revocation time, revocation reason), and signatures of these triples signed by CAs [37]. Network operators not only need to store the certificates of the currently valid users, but also need to store the information about all users who have been revoked in the past. Users need to periodically download and cache CRLs to use them when authenticating other users. However, in our scheme, network operators do not need to maintain CRLs, and users do not need to download the CRLs. Table II illuminates the key parameters in CRLs and certificates, which are designed according to bitcoin [5]. In addition, according to the design of the original Bitcoin block [5], we assume that the size of the block is 1 MB, and each transaction contains 1000 users' information.

The comparison of network operator's storage overhead of traditional CRL-based scheme and our proposed scheme are shown in Fig. 6. In both schemes, the storage cost of the network

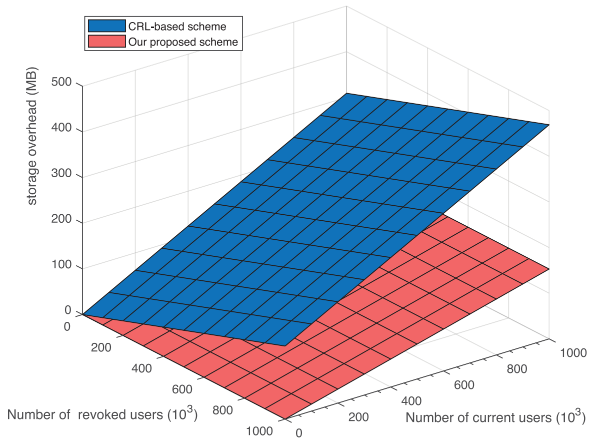


Fig. 6. Storage overhead.

operator increases with the increase of the number of existing users. However, when the number of revoked users increases, the CRL-based scheme storage overhead increases and our scheme remains unchanged. In general, the storage overhead of the CRL-based scheme is much higher than the storage overhead of our scheme.

VIII. Conclusion

In this paper, we proposed a self-sovereign identity management and authentication scheme for wireless mobile networks, which is based on redactable blockchain. In our user-centric identity management, users' identifying information is completely controlled and managed by the owner, thereby reducing the leakage of privacy. We adopted blockchain to record SSI and public key to achieve user authentication and session key negotiation without redundant registration. Furthermore, network operators can dynamically revoke users access to the network, and maintain a redactable blockchain instead of using revocation lists. Performance evaluation showed that our scheme not only reduces the storage overhead of network operators, but also reduces the network access delay for users.

References

[1] M. Khan and V. Niemi, "Privacy enhanced fast mutual authentication in 5G network using identity based encryption," *J. ICT Standardization*, vol. 5, no. 1, pp. 69–90, 2017.

[2] Y. Huang, C. Shen, and S. W. Shieh, "S-AKA: A provable and secure authentication key agreement protocol for UMTS networks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 9, pp. 4509–4519, Nov. 2011.

[3] Y. Zhang, R. Deng, E. Bertino, and D. Zheng, "Robust and universal seamless handover authentication in 5G hetnets," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2019.2927664.

[4] K. C. Toth and A.-P. Alan, "Self-sovereign digital identity: A paradigm shift for identity," *IEEE Secur. Privacy*, vol. 17, no. 3, pp. 17–27, May–Jun. 2019.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf, Accessed on: Apr. 3, 2020.

[6] K. Gai, Y. Wu, L. Zhu, Z. Zhang, and M. Qiu, "Differential privacy-based blockchain for industrial internet-of-things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4156–4165, Jun. 2020.

[7] J. Xu *et al.*, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8770–8781, Oct. 2019.

[8] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7992–8004, Oct. 2019.

[9] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Distributed resource allocation in blockchain-based video streaming systems with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 695–708, Jan. 2019.

[10] H. Krawczyk and T. Rabin, "Chameleon signatures," in *Proc. 7th Netw. Distrib. Syst. Secur. Symp.*, 2000, pp. 143–154.

[11] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 3, pp. 608–619, May-Jun. 2020.

[12] Z. Liu, Z. Liu, Z. Lin, and X. Lin, "MARP: A distributed MAC layer attack resistant pseudonym scheme for VANET," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2018.2838136.

[13] K. Xue *et al.*, "A secure, efficient, and accountable edge-based access control framework for information centric networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1220–1233, Jun. 2019.

[14] R. Dhamija and L. Dusseault, "The seven flaws of identity management: Usability and security challenges," *IEEE Secur. Privacy*, vol. 6, no. 2, pp. 24–29, Mar.-Apr. 2008.

[15] J. Carretero, G. I.-Moreno, M. V.-Cabezas, and J. G.-Blas, "Federated identity architecture of the European eID system," *IEEE Access*, vol. 6, pp. 75 302–75 326, 2018.

[16] U. Premarathne, I. Khalil, Z. Tari, and A. Zomaya, "Cloud-based utility service framework fortrust negotiations using federated identity management," *IEEE Trans. Cloud Comput.*, vol. 5, no. 2, pp. 290–302, Apr.-Jun. 2017.

[17] B. C. Singh, B. Carminati, and E. Ferrari, "Privacy-aware personal data storage (P-PDS): Learning how to protect user privacy from external applications," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2019.2903802.

[18] H. Gunasinghe and E. Bertino, "PrivBioMTAuth: Privacy preserving biometrics-based and user centric protocol for user authentication from mobile phones," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 4, pp. 1042–1057, Apr. 2018.

[19] P. Bramhall, M. Hansen, K. Rannenberg, and T. Roessler, "User-centric identity management: New trends in standardization and regulation," *IEEE Secur. Privacy*, vol. 5, no. 4, pp. 84–87, Jul.-Aug. 2007.

[20] S. Alboaie and D. Cosovan, "Private data system enabling self-sovereign storage managed by executable choreographies," in *Proc. IFIP Int. Conf. Distrib. Appl. Interoperable Syst.*, 2017, pp. 83–98.

[21] A. Othman and J. Callahan, "The Horcrux Protocol: A method for decentralized biometric-based self-sovereign identity," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2018, pp. 1–7.

[22] M. S. Ferdous, F. Chowdhury, and M. Alassafi, "In search of self-sovereign identity leveraging blockchain technology," *IEEE Access*, vol. 7, pp. 103 059–103 079, 2019.

[23] C. Lundkvist, R. Heck, J. Torstensson, Z. Mitton, and M. Sena, "Uport: A platform for self-sovereign identity," 2016. [Online]. Available: https://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf, Accessed on: Apr. 3, 2020.

[24] "Sovrin," [Online]. Available: https://sovrin.org, Accessed on: Apr. 3, 2020.

[25] "ShoCard," [Online]. Available: https://shocard.com, Accessed on: Apr. 3, 2020.

[26] N. Lo and J. Tsai, "An efficient conditional privacy-preserving authentication scheme for vehicular sensor networks without pairings," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1319–1328, May 2016.

[27] D. He, S. Zeadally, B. Xu, and X. Huang, "An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 12, pp. 2681–2691, Dec. 2015.

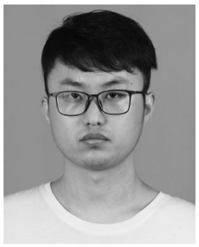
[28] D. He, S. Chan, and M. Guizani, "An accountable, privacy-preserving, and efficient authentication framework for wireless access networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 3, pp. 1605–1614, Mar. 2016.

[29] D. He, J. Bu, S. Chan, C. Chen, and M. Yin, "Privacy-preserving universal authentication protocol for wireless communications," *IEEE Trans. Wireless Commun.*, vol. 10, no. 2, pp. 431–436, Feb. 2011.

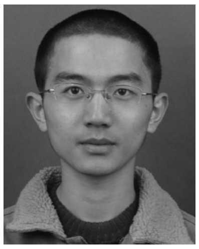
[30] C. Chang and H. Tsai, "An anonymous and self-verified mobile authentication with authenticated key agreement for large-scale wireless networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 11, pp. 3346–3353, Nov. 2010.
[31] H. J. Jo, J. H. Paik, and D. H. Lee, "Efficient privacy-preserving authentication in wireless mobile networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 7, pp. 1469–1481, Jul. 2014.
[32] Q. Yang, K. Xue, J. Xu, J. Wang, F. Li, and N. Yu, "AnFRA: Anonymous and fast roaming authentication for space information network," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 2, pp. 486–497, Feb. 2019.
[33] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable blockchain–or–rewriting history in bitcoin and friends," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, 2017, pp. 111–126.
[34] D. Derler, K. Samelin, D. Slamanig, and C. Striecks, "Fine-grained and controlled rewriting in blockchains: Chameleon-hashing gone attribute-based," in *Proc. 26th Netw. Distrib. Syst. Secur. Symp.*, 2019, doi: 10.14722/ndss.2019.23066.
[35] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. 3rd Symp. Operating Syst. Design Implementation*, 1999, pp. 173–186.
[36] J. Poon and T. Dryja, "The Bitcoin lightning Network: Scalable off-chain instant payments," 2016. [Online]. Available: https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-n etwork-paper.pdf, Accessed on: Apr. 3, 2020.
[37] L. Zhang *et al.*, "Analysis of SSL certificate reissues and revocations in the wake of Heartbleed," in *Proc. Conf. Internet Meas. Conf.*, 2014, pp. 489–502.

Jie Xu received the B.S. degree from the University of Science and Technology of China (USTC) in 2017. She is currently a Graduate Student in communication and information System with the Department of Electronic Engineering and Information Science (EEIS), USTC. Her research interests include network security and cryptography.

Kaiping Xue (Senior Member, IEEE) received the B.S. degree from the University of Science and Technology of China (USTC), in 2003 and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2007. From May 2012 to May 2013, he was a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, University of Florida. Currently, he is an Associate Professor in the School of Cyber security and the Department of EEIS, USTC. His research interests include next-generation Internet, distributed networks and network security. He serves on the editorial Board of several journals, including the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (TWC) and the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT (TNSM). He is a Fellow of the IET.



Hangyu Tian received the B.S. degree from the University of Science and Technology of China (USTC) in July, 2018. He is currently a Graduate Student in communication and information system with the Department of Electronic Engineering and Information Science (EEIS), USTC. His research interests include network security and cryptography.



Jianan Hong received the B.S. degree from the University of Science and Technology of China (USTC), in 2012 and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2018. Currently, he is a Research Engineer in Huawei Shanghai Research Institute, Shanghai. His research interests include secure cloud computing and mobile network security.

David S.L. Wei (Senior Member, IEEE) received the Ph.D. degree in computer and information science from the University of Pennsylvania, in 1991. From May 1993 to August 1997 he was at the Faculty of computer science and engineering with the University of Aizu, Japan (as an Associate Professor and then a Professor). He has authored and coauthored more than 100 technical papers in various archival journals and conference proceedings. He is currently a Professor with Computer and Information Science Department at Fordham University. His research interests include cloud computing, big data, IoT, and cognitive radio networks. He was a Guest Editor or a Lead Guest Editor for several Special Issues in the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON CLOUD COMPUTING AND IEEE TRANSACTIONS ON BIG DATA. He also served as an Associate Editor of IEEE TRANSACTIONS ON CLOUD COMPUTING, 2014–2018, and an Associate Editor of *Journal of Circuits, Systems, and Computers*, 2013–2018.

Peilin Hong received the B.S. and M.S. degrees from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1983 and 1986. Currently, she is a Professor and Advisor for Ph.D. candidates with the Department of EEIS, USTC. Her research interests include next-generation Internet, policy control, IP QoS, and information security. She has published 2 books and over 150 academic papers in several journals and conference proceedings.