An Efficient Traceable and Anonymous Authentication Scheme for Permissioned Blockchain

Qianqian Su[1,2], Rui Zhang[1,2(✉)], Rui Xue[1,2], and You Sun[1,2]

[1] State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences,
Beijing 100093, China
{suqianqian,zhangrui,xuerui,sunyou}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences,
Beijing 100049, China

Abstract. Blockchain has become a hot topic in recent years. Many applications apply permissioned blockchain to achieve secure data sharing across organizations such as healthcare blockchain. In the permissioned blockchain, on the one hand, the blockchain system is required to support efficient and dynamic authentication for adding and deleting users in a distributed environment. On the other hand, in some particular applications such as healthcare domain, users prefer to keep anonymity in the process of authentication. Although many solutions for anonymous authentication have been proposed, they often require the participation of a central trusted party in the process of authentication and are not efficient enough. In this paper, we focus on designing an efficient traceable and anonymous authentication scheme, which supports efficient authentication while without revealing user's identity information and does not requires the participation of a central trusted party. While, in case of dispute, the identity of users can be revealed. Moreover, the proposed scheme is able to support dynamic adding and deleting users. Finally, we analyze the security and privacy properties of the proposed scheme and evaluate its performance in terms of computational cost. The experimental results show that the proposed scheme is more efficient than exist schemes and can be easily deployed in the permissioned blockchain.

Keywords: Authentication · Anonymous · Traceable · Permissioned blockchain

1 Introduction

Recently, blockchain [13] as a breaking new technology attracts more and more attention to academia and industry due to its decentralization and tamper-proof characteristics. Permissioned blockchain is one type of blockchain. Unlike the

J. Miller et al. (Eds.): ICWS 2019, LNCS 11512, pp. 110–125, 2019.
https://doi.org/10.1007/978-3-030-23499-7_8

permissionless blockchain [11], permissioned blockchain such as R3 [7], Hyper-Ledger fabric [2], Ripple [3], FISCO [1] requires trusted nodes to authorize permissions to users to read information stored on the blockchain. One important application of permissioned blockchain is that it can achieve data sharing among authorized users.

Consider the following scenario: users (nodes) from different organizations, such as healthcare centers, pharmacies, insurance companies, researchers, and patients, maintain a permissioned blockchain together to share electric healthcare records (EHRs). One key challenge in this scenario is efficiently authenticating users from different organizations in a distributed network. Apparently, the traditional authentication schemes, which requires a central trusted party to authenticate users, cannot be directly applied in this scenario. Moreover, users usually prefer to keep their identity anonymity in the process of authentication. For example, a patient is willing to share his EHRs with other healthcare providers and researchers, but he does not want to blockchain provider and other users to know his real identity. While, in the event of a medical incident or dispute, users' identity can be revealed.

Our goal is to design an efficient traceable and anonymous authentication scheme that can be applied to the permissioned blockchain. To achieve this goal, the following problems should be taken into consideration. (1) The authentication scheme used in permissioned blockchain should be implemented in a distributed manner. Unlike the centralized manner in which a central authentication entity is needed in the authentication phase, the identity of users should be verified without the help of the central authority. (2) In order to protect the privacy of users, the anonymous property should be supported in the authentication scheme. (3) The system should support dynamic adding and revocation of users after system has been initialized.

The main contributions of this paper can be summarized as follows. We present a novel authentication scheme for permissioned blockchain which can efficiently achieve anonymous authentication without the participant of a central trusted party. The real identity of users can be revealed when disputed occurs. The proposed scheme also can support dynamic joining and revocation of users after the system has been initialized. The experimental results show that the proposed scheme is more efficient than exist schemes and can be easily deployed in the permissioned blockchain.

2 Related Work

Numerous anonymous authentication protocols have been proposed. In general, researches on anonymous authentication schemes are classifies into two types: three-party (password, smart card and biometric) schemes [5,6,16,18] and two-party (password and smart card) schemes [8,10,14,15,17].

Zhu et al. [18] proposed a three-party authentication scheme but the proposed scheme failed to provide user anonymity and backward secrecy, and it is susceptible to forgery attack. Gope et al. [5] proposed an efficient scheme which

fulfilled strong user anonymity but it requires large storage to store pseudo-identities. Moreover, it is not secure against desynchronization attack and DoS attack because users require to update their pseudo-identities. Wu et al. [16] proposed a scheme which fulfilled strong user anonymity and strong key establishment property but it is not secure against desynchronization attack due to the pseudo-identity of users need to update. Meanwhile, some two-party schemes were also proposed, but their computational costs were very high. Ni et al. proposed an anonymous mutual authentication protocol by utilizing the BBS+ signature in [10], but their scheme cannot protect the privacy of user. Yang et al. introduced the concept of two-party anonymous authentication scheme, and proposed a pseudo-identity-based authentication scheme in [17]. Unfortunately, the proposed scheme is vulnerable to private key reveal attack and has heavy computation overhead at users side. Jo et al. proposed an efficient pseudo-identity two-party authentication scheme in [15]. Their scheme employs signcryption to minimize the number of pseudo-identities stored on users side.

These existing anonymous authentication schemes are difficult to be applied to the distributed permissioned blockchains scenario for the following reasons. First of all, in the three-party authentication schemes, the third trusted parity participates in the authentication process. If this connection is interfered and broken by an adversary, corresponding users cannot get successful access of their services. Secondly, users requires a huge storage capacity to store all corresponding public keys of group managers in advance, since those schemes employ cryptography methods to help users authenticate the group manager. Without knowing these public keys of group managers, the user cannot verify the signatures generated by a group manager during authentication. In addition, group managers require multiple interactions with each other, since they need to download and update its whole revocation list or other parameters from the other group managers periodically.

3 Preliminaries

In this section, we review some preliminary cryptography knowledge, including bilinear map, Discrete Logarithm (DL) problem, q-SDH problem, Bonen-Boyens SDH equivalence, Forking Lamma and anonymous authentication.

3.1 Bilinear Pairings

Let G_1, G_2 and G_T be three multiplicative cyclic groups of the order q, where q is a large prime number. Let g_1, g_2 be the generator of G_1 and G_2, respectively. We say a map $e : G_1 \times G_2 \rightarrow G_T$ is a bilinear pairing if it satisfies:

(1) Bilinearity: For any $P \in G_1, Q \in G_2$ and $a, b \in Z_q^*$, $e(P_1^a, Q_1^b) = e(P_1, Q_1)^{ab}$.
(2) Non-degeneracy: Whenever g_1 is a generator of G_1 and g_2 is a generator of G_2, $e(g_1, g_2) \neq 1$.
(3) Computability: There is an efficient algorithm to compute $e(P, Q)$ for any $P \in G_1, Q \in G_2$.

A bilinear parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter k as input and outputs $(q, G_1, G_2, G_T, e, g_1, g_2)$ as bilinear parameters.

3.2 Discrete Logarithm (DL) Problem

For $x \in Z_q^*$, given $g, g^x \in G_1$ as input, output x. The DL assumption in G_1 holds if it is computationally infeasible to solve the DL problem in G_1.

3.3 *q*-SDH Problem

Takes $(q+2)$ tuple $(g_1, g_2, g_2^{\gamma}, ..., g_2^{\gamma^q})$ as the input, output a SDH pair and that equals $(g_1^{1/(x+\gamma)}, x)$ where $x \in Z_p^*$. If $|Pr[A(g_1, g_2, g_2^{\gamma}, ..., g_2^{\gamma^q}) = (g_1^{1/\gamma+x}, x)] \geq \epsilon$, the algorithm A has an advantage ε in solving $q-SDH$ in (G_1, G_2). This problem is considered hard to solve in polynomial time and ϵ should be negligible.

3.4 Bonen-Boyens SDH Equivalence

Given a $q-SDH$ instance $(g_1, g_2, g_2^{\gamma}, ..., g_2^{\gamma^q})$, and then applying the Boneh and Boyen's Lemma found in [4] we could obtain $g_1 \in G_1$, $g_2 \in G_2$, $w = g_2^{\gamma}$ and $(q-1)$ SDH pairs (A_i, x_i) (such that $e(A_i, wg_2^{x_i}) = e(g_1, g_2)$) for each i. Any SDH pair besides these $(q-1)$ ones can be transformed into a solution to the original q-SDH instance.

3.5 The Forking Lamma

Given only public data as input, if an adversary $\mathcal{A}$ with polynomial computation ability can find a valid signature $(M, \delta_0, c, \delta_1)$ with non-negligible probability, then there exists a replay with a different oracle, which can output new valid signatures $(M, \delta_0, c', \delta_1')$ with non-negligible probability where $c \neq c'$.

3.6 Anonymous Authentication

The anonymous authentication scheme [9] consists of the following algorithms:

- ***Setup*** (1^k)**.** Takes security parameter k as input, outputs the system parameters $params$, the master private key $msk = \{u, v\}$, the master public key $mpk = \{U_1, U_2, V_1\}$, and a hash function H.
- ***KeyGen*** $(params, msk, i)$**.** Takes parameters $params$, master private key msk and the identity of user i as inputs, outputs an anonymous key $AK_i = (s_i, S_i)$ for user i.
- ***PseudoGen*** $(params, AK)$. Takes parameters $params$ and anonymous key AK as inputs, outputs temporary private keys $(x_1, x_2, ..., x_l)$ and corresponding public keys $(Y_1, Y_2, ..., Y_l)$.

- ***Sign*** $(params, S, x, Y, m)$**.** Takes parameters $params$, an anonymous key AK, a temporary private key pairs (x, Y) and message m as inputs, generates an anonymous certification $Cert$ and a signature σ as outputs.
- ***Verify*** $(params, m, \sigma, Cert)$**.** Takes parameters $params$, the message m, certification $Cert$ and signature σ as inputs, output 1 if $Cert$ and σ is valid, or output $\perp$ otherwise.
- ***Open*** $(params, msk, Cert)$**.** Takes parameters $params$, master private key msk and certification $Cert$ as inputs, output the identity of user i.

4 Definitions

4.1 System Model

The model of the proposed scheme involves a trusted third party (TTP) and N organizations. Each organization contains a control center (CCenter), an authentication dealing module (ADM) and M users. For clarity and without loss of generality, we simplify the system model with only one TTP and two organizations, and each organization contains a user, as shown in Fig. 1.

- *Organization (Org)*: Organization is the participant of the system and maintainer of the permissioned blockchain. These organizations can be hospitals, Banks or companies. Every organization contains a control center (CCenter), an authentication dealing module (ADM) and M users.
- *Trusted Third Party (TTP)*: TTP is assumed powered with sufficient resources, and fully trusted by all organizations. The main tasks of TTP are (1) managing the registration of the organization, joining and revoking, (2) generating and distributing private key pairs (sk, pk) for CCenters. It's worth mentioning that TTP will remain offline after initialization until there is an organization joining or revoking.
- *Control Center (CCenter)*: CCenter is the manager of an organization. CCenter is responsible for (1) generating public parameters and master private keys, and (2) managing the registration of users, joining and revoking.
- *Authentication Dealing Module (ADM)*: ADM is responsible for handling the authentication process, including verify the identity and generate the request response.

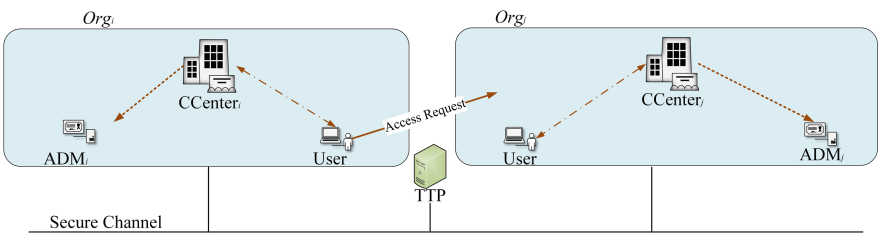

Fig. 1. System model

– *Users*: Users access to organizations to obtain some services or data. In this paper, we consider the scenario where a user requests services or data from other organizations within the system.

4.2 Formal Definition of the Proposed Scheme

In this section, we formally define the algorithms that form the proposed scheme.

Definition 1: A traceable and anonymous authentication scheme consists of the following algorithms:

– **SSetup**(λ, ID_{Org}) is an algorithm run by TTP. Taking a security parameter λ and the identity of the Org as the input, this algorithm outputs private key pairs (pk, sk) for $CCenter$ belongs to ID_{Org}.
– **CSetup**(λ, pk, sk) is an algorithm run by $CCenter$. Taking a security parameter λ and key pairs (sk, pk) generated by TTP as the input, each $CCenter$ outputs the system parameters gpk, the master private key $gmsk$, and the partial key Δ.
– **KeyGen**$(gpk, gmsk, U)$ is an algorithm run by $CCenter$. Taking the system parameters gpk, the master private key $gmsk$ and the users identity U as the input. This algorithm outputs an authorized anonymous key ASK for U.
– **ProofGen**(gpk, M, ASK, Δ) is an algorithm run by the user U. Taking the system parameters gpk, a message M, the authorized anonymous key ASK of the user U and the partial key Δ as the input, this algorithm outputs a certification $Cert$ on the message M.
– **ProofVer**$(gpk, M, Cert, \Delta)$ is an algorithm run by ADM. Taking the system parameters gpk, the certification $Cert$, the signature σ, the partial key Δ, and the message M as the input, this algorithm outputs $true$ for a valid proof $Cert$ and $false$ otherwise.
– **Reveal**$(gpk, gmsk, Cert)$ is an algorithm run by $CCenter$. Taking the system parameters gpk, the master private key $gmsk$ and the certification $Cert$ as the input, this algorithm outputs an identity for a valid certification or $false$ otherwise.
– **KeyUpdate**$(gpk, gmsk)$ is an algorithm run by $CCenter$. Taking the system parameters gpk and the master private key $gmsk$ as the input, this algorithm outputs the partial key Δ.

We say that a traceable and anonymous authentication scheme is correct, meaning that for any message M, any non-revoked user U registered to $CCenter$ belonging to a non-revoked organization ID_{Org}, if $(pk, sk) \leftarrow$ **SSetup**(λ, ID_{Org}), $(gpk, gmsk, \Delta) \leftarrow$ **CSetup**(λ, pk, sk), $ASK \leftarrow$ **KeyGen**$(gpk, gmsk, U)$, $Cert \leftarrow$ **ProofGen**(gpk, M, ASK, Δ), then **ProofVer**$(gpk, M, Cert, \Delta) \rightarrow true$.

4.3 Security Definitions

In this section, we formally define the security prosperities that must be required to the proposed scheme.

Definition 2 (Anonymity): A traceable and anonymous authentication scheme is anonymity if the advantage of the success probability of any polynomial-time adversary $\mathcal{A}$ in the following experiment is negligible.

- **Setup:** The adversary $\mathcal{A}$ chooses an organization labeled as ID_{Org}, and the TTP runs $(pk, sk) \leftarrow \mathbf{SSetup}(\lambda, ID_{Org})$, The $CCenter$ runs $(gpk, gmsk, \Delta) \leftarrow \mathbf{CSetup}(\lambda, pk, sk)$. Give the public parameters $(pk, gpk, ID_{Org}, \Delta)$ to the adversary $\mathcal{A}$.
- **KeyGen Query:**
  (1) The adversary $\mathcal{A}$ chooses a user U_i and issues it to the challenger C.
  (2) The challenger C maintain a query list (initialize as empty) to save the queried user's identity and its ASK. After receiving U_i from the adversary $\mathcal{A}$, if U_i is in the query list, the challenger C obtain its ASK_i. Otherwise, the challenger C runs $ASK_i \leftarrow \mathbf{KeyGen}(gpk, gmsk, U_i)$, and adds (U_i, ASK_i) to the query list.
  (3) After that, the challenger C sends ASK_i to the adversary $\mathcal{A}$.
- **ProofGen Query:**
  (1) The adversary $\mathcal{A}$ chooses a user U_i and a message M, and issues it to the challenger C.
  (2) If U_i is in the query list, the challenger C runs $Cert \leftarrow \mathbf{ProofGen}(gpk, M, ASK_i, \Delta)$, and sends $Cert$ to the adversary $\mathcal{A}$. If U_i is not in the query list, the challenger C runs $ASK_i \leftarrow \mathbf{KeyGen}(gpk, gmsk, U_i)$, and adds (U_i, ASK_i) to the query list. After that, the challenger C runs $Cert \leftarrow \mathbf{ProofGen}(gpk, M, ASK_i, \Delta)$, and sends $Cert$ to the adversary $\mathcal{A}$.
- **Challenge:**
  (1) The adversary $\mathcal{A}$ sends a message m^*, and two identities U_0 and U_1 to the challenger C.
  (2) The challenger C randomly chooses a bit $b \in \{0, 1\}$, runs $ASK_b \leftarrow \mathbf{KeyGen}(gpk, gmsk, U_b)$ and $Cert_b \leftarrow \mathbf{ProofGen}(gpk, m^*, ASK_b, \Delta)$.
  (3) The challenger C sends $Cert_b$ to the adversary $\mathcal{A}$.
- **Guess:** After receiving $Cert_b$, the adversary $\mathcal{A}$ outputs a guess b'.

We say the adversary $\mathcal{A}$ succeeds, if $b' = b$. Otherwise, the adversary $\mathcal{A}$ fails.

Definition 3 (Traceability): We say that a traceable and anonymous authentication scheme is traceable, meaning that for any message M, any non-revoked user U registered to $CCenter$ belonging to a non-revoked organization, if $(pk, sk) \leftarrow \mathbf{SSetup}(\lambda, ID_{Org})$, $(gpk, gmsk, \Delta) \leftarrow \mathbf{CSetup}(\lambda, pk, sk)$, $ASK \leftarrow \mathbf{KeyGen}(gpk, gmsk, U)$, $Cert \leftarrow \mathbf{ProofGen}(gpk, M, ASK, \Delta)$, then $\mathbf{Reveal}(gpk, gmsk, Cert) \rightarrow U$.

Definition 4 (Unlinkability): A traceable and anonymous authentication scheme is unlinkability if the advantage of the success probability of any polynomial-time adversary $\mathcal{A}$ in the following experiment is negligible.

- **Setup:** The setup is the same as in Definition 2.
- **KeyGen Query:** The keygen query is the same as in Definition 2.
- **ProofGen Query:**The proofgen query is the same as in Definition 2.
- **Challenge:**
  (1) The adversary $\mathcal{A}$ sends a message m^*, and two identities U_0 and U_1 to the challenger C.
  (2) The challenger C randomly chooses a bit $b \in \{0,1\}$, runs $ASK_b \leftarrow \mathbf{KeyGen}(gpk, gmsk, U_b)$ and $Cert_b \leftarrow \mathbf{ProofGen}(gpk, m^*, ASK_b, \Delta)$. And the challenger C randomly chooses a bit $b' \in \{0,1\}$, runs $ASK_{b'} \leftarrow \mathbf{KeyGen}(gpk, gmsk, U_{b'})$ and $Cert_{b'} \leftarrow \mathbf{ProofGen}(gpk, m^*, ASK_{b'}, \Delta)$.
  (3) The challenger C sends $Cert_b$ and $Cert_{b'}$ to the adversary $\mathcal{A}$.
- **Guess:** After receiving $Cert_b$ and $Cert_{b'}$, the adversary $\mathcal{A}$ outputs a guess *yes* or *no*.

We say the adversary $\mathcal{A}$ succeeds, if $b' = b$ and the adversary A output *yes*, or $b' \neq b$ and the adversary A outputs *no*.

Definition 5 (Unforgeability): A traceable and anonymous authentication scheme is unforgeability if the success probability of any polynomial-time adversary $\mathcal{A}$ in the following experiment is negligible.

- **Setup:** The setup is the same as in Definition 2.
- **KeyGen Query:** The keygen query is the same as in Definition 2.
- **ProofGen Query:**The proofgen query is the same as in Definition 2.
- **Forge:** The adversary $\mathcal{A}$ generates a forged certification $Cert^*$ based on a never queried m^* and U^*, and sends the forged certification $Cert^*$ to the challenger C.
- **Output:** The challenger C runs the $\mathbf{Reveal}(gpk, gmsk, Cert)$ algorithm. If $\mathbf{Reveal}(gpk, gmsk, Cert)$ returns U_i which has been queried, the challenger C outputs $false$. Otherwise, the challenger C runs the $\mathbf{ProofVer}(gpk, M, Cert^*, \Delta)$ algorithm and outputs the result $true/false$.

We say the adversary $\mathcal{A}$ succeeds, if the challenger outputs $true$ for the forged certification $Cert^*$.

5 The Proposed Scheme

5.1 Overview

In our proposed scheme, TTP generates private key pairs (pk, sk) for CCenters, and CCenter generates its system parameters gpk, master private key $gmsk$ and partial key Δ; meanwhile, it issues an authorized anonymous key ASK_i and a partial key Δ to user U_i. When U_i belonging to Org_i wants to access to Org_j, U_i generates the request message M and computes a certification $Cert$. After that, U_i sends $(M, Cert)$ to $Org_r (r = 1, 2, ..., N, r \neq i)$. On received $(M, Cert)$, the $ADM_r (r = 1, 2, ..., N, r \neq i)$ belonging to Org_r will check the valid of $Cert$. If the verification is passed and $r \neq j$, the ADM_r saves M in its local memory; if the

verification is passed and $r = j$, the ADM_r not only saves M, but also generates response. Otherwise, the ADM_r reject the message. By leveraging anonymous authentication, the identity of U_i can be anonymously verified, and CCenter can use gpk and $gmsk$ to reveal the identity of U_i when a dispute occurs.

When Org_{new} wants to join in the system after system initialization, TTP invokes the **SSetup** algorithm to generate private key pairs (pk, sk) for it. Similarly, when U_{new} wants to join in Org, the $CCenter$ belonging to Org invokes the **KeyGen** algorithm for U_{new}. When Org_{exit} is revoked, TTP adds its label ID_{Org} the revoked list (RL) and publish its key pairs to unrevoked organizations. When U_{exit} is revoked, $CCenter$ invokes the **KeyUpdate** algorithm to obtain a new partial key Δ and publish it to non-revoked users. It should be noted that although TTP is introduced in our system, it cannot interrupt the authentication process for TTP will be offline after the system initialization unless there are organizations want to join in or leave out the system.

5.2 Details of the Proposed Scheme

In this section, we construct our proposed scheme, and show the details of the proposed scheme as follows.

SSetup(λ, ID_{Org})**:** TTP generates and distributes the private key pairs (sk, pk) for $CCenter$ belonging to Org. The private key pairs (sk, pk) is used to exchanges information between $CCenter$s.

CSetup(λ, pk, sk)**:** $CCenter$ generates the bilinear parameters $(q, G_1, G_2, G_T, e, g_1, g_2)$ by running $\mathcal{G}en$, chooses two random numbers $a, b \in Z_q^*$, a public collision-resistant hash function $H : \{0,1\}^* \rightarrow Z_q^*$, a signature scheme Π, a random value t_0, and computes $A_1 = g_1^a$, $A_2 = g_2^a$, $B = g_1^b$, $\Delta = g_1^{1/(H(t_0)+a)}$. After that, $gpk = (q, G_1, G_2, G_T, e, g_1, g_2, A_1, A_2, B, H, \Pi)$, $gmsk = (a, b)$.

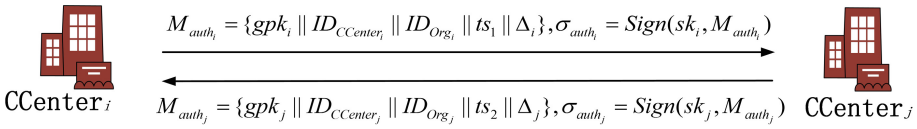

Fig. 2. Information exchange between CCenters

It is noted that $CCenter$ needs to exchange essential informations with each other, as shown in Fig. 2. $CCenter_i$ generates M_{auth_i} and signs it by $Sign(sk_i, M_{auth_i})$, where gpk_i is its system parameters, $ID_{CCenter_i}$ is its identity, ID_{Org_i} is its organization's label, ts is timestamps. Then, $CCenter_i$ sends M_{auth_i} and its signature to $CCenter_j (j \neq i)$. If ts is valid, and the signature is verified by $Verify(sk_i, M_{auth_i})$, M_{auth_i} will be stored in ADM_j by $CCenter_j$.

KeyGen$(gpk, gmsk, U)$**:** $CCenter$ selects a random number $s_i \in Z_q^*$ such that $s_i + a \neq 0 \ mod \ q$, and computes $S_i = g_1^{1/(s_i+a)}$. Let $ASK_i = (s_i, S_i)$,

$CCenter$ stores the tuple (U_i, S_i^a) in the user index table. After that, $CCenter$ sends (ASK_i, Δ_i) to the user U_i.

ProofGen(gpk, M, ASK, Δ)**:** U_i first generates the access request M, where $M = \{ID_{Org_i}||ID_{Org_j}||operation||t_1\}$, ID_{Org_i} is the label of organization that U_i belongs to, ID_{Org_j} is the label of organization that U_i is going to communicate with, and t_1 is a timestamp. Then, U_i selects four random numbers r, r_1, r_2, r_3 $\in Z_q^*$, and computes:

$$T_1 = A_1^r, T_2 = S_i \cdot B^r, \delta = r \cdot s_i \ mod\ q, R_1 = A_1^{r_1}, R_2 = T_1^{r_2}/A_1^{r_3};$$
$$R_3 = e(T_2, g_2^{r_2})/e(B, A_2^{r_1} \cdot g_2^{r_3}), c = H(M, A_1, B, T_1, T_2, R_1, R_2, R_3, \Delta);$$
$$s_1 = (r_1 + c \cdot r) mod\ q, s_2 = (r_2 + c \cdot s_i) mod\ q, s_3 = (r_3 + c \cdot \delta) mod\ q.$$

After that, U_i sets $Cert = \{T_1 \| T_2 \| c \| s_1 \| s_2 \| s_3\}$.

ProofVer$(gpk, M, \Delta, Cert)$**:** ADM_r first checks the valid of t_1. If t_1 is invalid, it rejects the message and outputs $false$. Otherwise, it computes:

$$R_1^{'} = A_1^{s_1}/T_1^c, R_2^{'} = T_1^{s_2}/A_1^{s_3}, R_3^{'} = e(T_2, g_2^{s_2} \cdot A_2^c)/(e(B, A_2^{s_1} \cdot g_2^{s_3}) \cdot e(g_1, g_2^c)).$$

ADM_r checks $c = H(M', A_1, B, T_1', T_2', R_1', R_2', R_3', \Delta^{'})$, where $\Delta^{'}$ is $CCenter_i$ shared to $CCenter_r$ during the $CSetup$ phase. If the above equation holds, it outputs $true$ indicating that the proof is valid. Otherwise, it outputs $false$.

After the verification is passed, ADM_r checks ID_{Org_j}, if $r \neq j$, Org_r is not the organization that U_i wants to communicate to. Then ADM_r stores $M||Cert$; If $r = j$, ADM_r not only stores it, but also dealing with the request.

Reveal$(gpk, gmsk, Cert)$**:** The $CCenter$ computes $S_i^a = T_2^a/T_1^b$, and looks up the user index table corresponding to S_i^a. If (U_i, S_i^a) is in the user index table, it outputs U_i. Otherwise, it outputs $false$.

KeyUpdate$(gpk, gmsk)$**:** The $CCenter$ chooses a random value t_0' to compute a new partial key Δ, and publish it to non-revoked users and other $CCenter$s. And then the non-revoked users and other $CCenter$s update their Δ.

Dynamic Join and Revocation: When an organization wants to join in the system after system initialization, TTP invokes the **SSetup** algorithm to generate private key pairs (pk, sk) for it. Similarly, when a user wants to join in an organization, the $CCenter$ belonging to the organization invokes the **KeyGen** algorithm for the user. And the $CCenter$ sends (ASK, Δ) to the user, where ASK is generated by invoking the **KeyGen** algorithm, and Δ is generated by **CSetup** algorithm during the system initialization.

As shown in Fig. 3, when an organization is revoked, TTP adds its label ID_{Org} to the revoked list (RL) and publish its key pairs to unrevoked organizations. When a user is revoked, the $CCenter$ invokes the **KeyUpdate** algorithm to obtain a new partial key Δ and publish the new Δ to non-revoked users. We illustrate this process in Fig. 4.

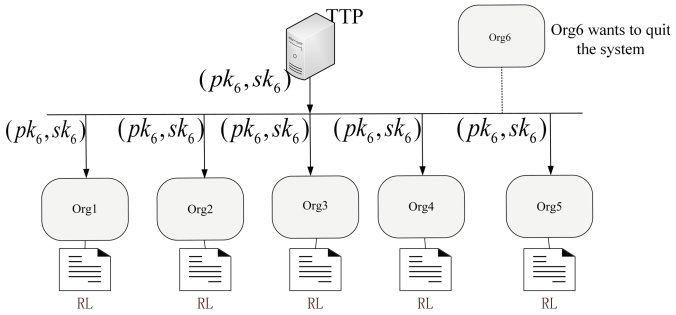

Fig. 3. Organization exit

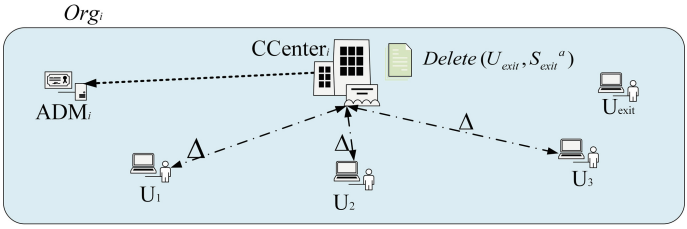

Fig. 4. User exit

6 Security and Performance Analyses

In this section, we first prove that the proposed scheme is correct and achieves the security requirements described in Sect. 4, and then evaluate the performance of the proposed scheme.

6.1 Security Analysis

Theorem 1 (Correctness). *If and only if the certificate Cert are valid, the request message can pass the authentication and be accepted.*

Proof. It is because the following equations hold:

$$R_1^{'} = A_1^{s_1} / T_1^{c} = A_1^{r_1 + c \cdot r} / A_1^{c \cdot r} = A^{r_1} = R_1 \tag{1}$$

$$R_2^{'} = T_1^{s_2} / A_1^{s_3} = T_1^{r_2 + c \cdot s_i} / A_1^{r_3 + c \cdot \delta} = T_1^{r_2} / A_1^{r_3} = R_2 \tag{2}$$

$$\begin{aligned} R_3^{'} &= \frac{e(T_2, g_2^{s_2} \cdot A_2^{c})}{e(B, A_2^{s_1} \cdot g_2^{s_3}) \cdot e(g_1, g_2^{c})} \\ &= \frac{e(T_2, g_2^{r_2})}{e(B, A_2^{r_1} \cdot g_2^{r_3})} \cdot \frac{e(S_i \cdot B^{r}, g_2^{c \cdot s_i} \cdot A_2^{c})}{e(B, A_2^{c \cdot r} \cdot g_2^{c \cdot \delta}) \cdot e(g_1, g_2^{c})} \\ &= \frac{e(T_2, g_2^{r_2})}{e(B, A_2^{r_1} \cdot g_2^{r_3})} = R_3 \end{aligned} \tag{3}$$

Theorem 2 (Traceability). *Each certification Cert generated by a registered user U_i belonging to Org, the identity of U_i can be traced by CCenter belonging to Org.*

Proof. If the *Cert* is generates by a registered user U_i belonging to *Org*, then the *CCenter* belonging to *Org* can get U_i's identity. It is because of $T_2^a/T_1^b = (S_i^a g_1^{r \cdot a \cdot b})/g_1^{r \cdot b \cdot a} = S_i^a$ holds. By looking up the user index table corresponding to S_i^a, *CCenter* can obtain the identity of U_i.

Theorem 3 (Anonymity). *The proposed scheme is anonymity if DL assumption holds in (G_1, G_2).*

Proof. The adversary $\mathcal{A}$ gives $U_{i(0)}, U_{i(1)}$ and the message m to the challenger C, and the challenger C chooses a random toss $b \in \{0, 1\}$, and generates $Cert_b$. Give $Cert_b$ to the adversary $\mathcal{A}$, where $Cert_b = \{T_{1(b)} \| T_{2(b)} \| c_{(b)} \| s_{1(b)} \| s_{2(b)} \| s_{3(b)}\}$, $b \in \{0, 1\}$. If the advantage that the adversary $\mathcal{A}$ with polynomial computation ability to guess the correct b is negligible, the proposed scheme is said to be anonymity.

Suppose that adversary $\mathcal{A}$ can break the anonymity of the proposed scheme, and then it means the adversary $\mathcal{A}$ has a non-negligible advantage to guess the correct b in the above statement. More specifically, given $U_{i(0)}$ and $U_{i(1)}$ and the message m, the adversary $\mathcal{A}$ has a non-negligible advantage to distinguish the tuple $Cert_0$ from $Cert_1$. From the proposed scheme, we have $Cert_0 = \{T_{1(0)} \| T_{2(0)} \| c_{(0)} \| s_{1(0)} \| s_{2(0)} \| s_{3(0)}\}$, $Cert_1 = \{T_{1(1)} \| T_{2(1)} \| c_{(1)} \| s_{1(1)} \| s_{2(1)} \| s_{3(1)}\}$, where $T_{1(i)} = A_1^{r_{(i)}}$, $T_{2(i)} = S_i B^{r_{(i)}}$, $c = H(m, A_1, B, T_{1(i)}, T_{2(i)}, R_{1(i)}, R_{2(i)}, R_{3(i)}, \Delta)$, and $r_{(i)}, s_{1(i)}, s_{2(i)}, s_{3(i)}$ are randomized values for $i = \{0, 1\}$.

If the adversary $\mathcal{A}$ has the ability to distinguish $Cert_0$ from $Cert_1$, it means $\mathcal{A}$ break DL problem, and it contradicts with DL assumption. Thus, it is impossible for $\mathcal{A}$ to distinguish $Cert_0$ from $Cert_1$ with a non-negligible probability, and the proposed scheme is anonymous.

Theorem 4 (Unforgeability). *The proposed scheme is unforgeability if SDH is hard in (G_1, G_2).*

Proof. Given a forged certification $Cert^*$ based on a message m^*, where $Cert^* = \{T_1 \| T_2 \| c_b \| s_1 \| s_2 \| s_3\}$. The probability that **ProofVer**$(gpk, m^*, Cert^*, \Delta)$ outputs *true* is negligible.

Suppose the adversary $\mathcal{A}$ can forge a certification $Cert^* = \{T_1 \| T_2 \| c_b \| s_1 \| s_2 \| s_3\}$ on message m^*. Let $M = m^*$, $\delta_0 = \{T_1, T_2\}$, $\delta_1 = \{s_1, s_2, s_3\}$ and $c = c_{(b)}$. The valid *Cert* on m^* can be viewed as a tuple $<M, \delta_0, c, \delta_1>$. According to Forking Lemma, we can extract a tuple $<\delta_0, c', \delta_1'>$ from $<\delta_0, c, \delta_1>$, where $c' \neq c$. Thus we can create a new SDH tuple denoted as (s_i', S_i') without the knowledge of a.

Using the two tuple $< \delta_0, c', \delta_1' >$ from $< \delta_0, c, \delta_1 >$, we could extract a new SDH tuple, Let $\Delta c = c - c'$, $\Delta s_1 = s_1 - s_1'$, $\Delta s_2 = s_2 - s_2'$ and $\Delta s_3 = s_3 - s_3'$. Divide the two instance of the equations used previously in proving correctness

of the proposed scheme. One instance with c' and the other is with c to get the following:

- Dividing $T_1^c/T_1^{c'} = A_1^{s_1}/A_1^{s_1'}$ we get $A_1^{r'} = T_1$, where $r' = \Delta s_1/\Delta c$.
- Dividing $T_1^{s_2}/T_1^{s_2'} = A_1^{s_3}/A_1^{s_3'}$ we get $\Delta s_3 = r'\Delta s_2$.
- Diving $(e(g_1, g_2)/e(T_2, A_2))^{\Delta c}$ will lead to

$$e(T_2, g_2)^{\Delta s_2} e(B, A_2)^{-r'\Delta c} e(B, g_2)^{-r'\Delta s_2} = (e(g_1, g_2)/e(T_2, A_2))^{\Delta c}$$

Letting $s_i' = \Delta s_2/\Delta c$, we get

$$e(g_1, g_2)/e(T_2, A_2) = e(T_2, g_2)^{s_i'} e(B, g_2)^{-r'} e(B, g_2)^{-r's_i'}$$

This could be rearranged as $e(g_1, g_2) = e(T_2 B^{-r'}, A_2 g_2^{s_i'})$. Let $S_i' = T_2 B^{-r'}$, we get $e(S_i', A_2 g_2^{s_i'}) = e(g_1, g_2)$. Hence, we obtain a new SDH pair (s_i', S_i') breaking Boneh and Boyens theorem. Thus, it is impossible for the adversary $\mathcal{A}$ to generates a forged certification $Cert^*$ and the certification can pass the verification. Thus, the proposed scheme is unforgeability.

Theorem 5 (Unlinkability). *The proposed scheme is unlinkability if DL assumption holds in* (G_1, G_2).

Proof. Unlinkability is covered through full anonymity. If the scheme was linkable, the adversary of the anonymity game can query a proof of U_i and later in the challenge phase include U_i among the users he wants to be challenged upon. The adversary does not to guess the generator of the proof he obtains in the challenge phase, because he can just link it thus breaking full anonymity.

In other words, similarly to the proof in Theorem 3, given U_0 and U_1 and the message m, the challenger response $Cert_b$ and $Cert_{b'}$ to the adversary $\mathcal{A}$, where $b \in \{0, 1\}$ and $b' \in \{0, 1\}$. If the scheme was linkable, the adversary $\mathcal{A}$ has non-negligible advantage to guess whether $b = b'$. It means that the adversary $\mathcal{A}$ has the ability to distinguish $Cert_0$ from $Cert_1$ by solving DL problem, and it contradicts with DL assumption. Thus, it is impossible for $\mathcal{A}$ to guess whether $b = b'$ with a non-negligible probability, and the proposed scheme is unlinkability.

6.2 Performance Analysis

We first analyze the overhead of the proposed scheme for proof generation, proof verifies, identity reveal and dynamic join and revocation. And then, we make the experimental evaluation of the proposed scheme.These experiments are carried out on a Linux machine with an Intel Pentium processor of 2.70 GHz and 8 GB memory.

To generate the proof, the user needs to conduct 1 hash operation, 3 additions, 2 divisions, 6 multiplications, 7 exponentiations in G_1, 3 exponentiations in G_2, 2 bilinear pairings and 1 division in G_T. To verify the proof, ADM needs to conducts 1 hash operation, 2 divisions, 3 multiplications, 5 exponentiations

in G_1, 5 exponentiations in G_2, 4 bilinear pairings and 1 division in G_T. When a new organization wants to join in the system, TTP needs to distribute key pairs for it. When an organization wants to leave out the system, TTP needs to add its key pairs to revoked list (RL) and distribute RL to unrevoked organizations. When a new user wants to join an organization, the *CCenter* needs to conduct 1 exponentiation in G_1. When a user wants to leave out its organization, the *CCenter* needs to conduct 1 eFan2016AFan2016Axponentiation in G_1.

Next, we show the experimental evaluation of the proposed scheme. Firstly, We present the time cost of key generation in Fig. 5. The X-axis presents the number of users. The Y-axis represents the corresponding time overhead. Figure 6 shows that the time cost in proof generation, verifying and identity revealing. The X-axis presents the number of certificate should be generate (verified/revealing). The Y-axis represents the corresponding time overhead. Figure 7 presents the time cost of user join and revocation. The X-axis represents the number of added (revoked) users, and the Y-axis represents the time cost. As described in dynamic join and revocation phase, when a new user joins, there is no impact on the old users for it is only needed that *CCenter* computes ASK and sends (ASK, Δ) to the new user. Therefore, the time cost only happens in *CCenter* side. When the user exit, it is needed that *CCenter* generates a new Δ and sends it to unrevoked users.

In Fig. 8, we compare the time cost of certification generation with the scheme in [12]. It is should note that the scheme in [12] is designed to enable data sharing for the same group in the cloud. Shen's scheme [12] uses group signature to achieve anonymity and traceability. Therefore, we illustrate the efficiency of our scheme by comparing our scheme with [12]. The X-axis presents the number of generated certificate. The Y-axis represents the corresponding time overhead. From Fig. 8, we can see that the time cost of our scheme is more smaller than that of [12]. Figure 9 shows the time cost of verifying in [12] and our proposed scheme. The X-axis presents the number of verified authentication. The Y-axis represents the time cost. From Figs. 8 and 9, it is easily observed that the proposed scheme has advantages in terms of efficiency authentication.

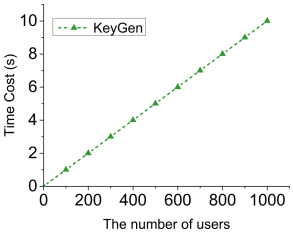

Fig. 5. The time cost of key generation

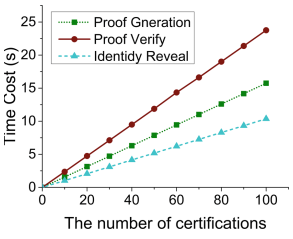

Fig. 6. The time cost of proof generation, verification and identity reveal

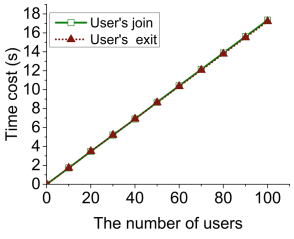

Fig. 7. The time cost of users join and revocation

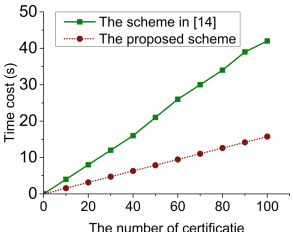

Fig. 8. The comparison of the time cost of generation

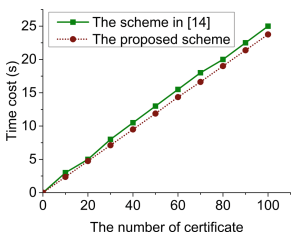

Fig. 9. The comparison of the time cost of verify

7 Conclusion

In this paper, we design a traceable and anonymous authentication scheme with high security and efficiency for permissioned blockchain. In the proposed scheme, an organization can efficiently verify the identity of users in a distributed and anonymous manner. Besides, the proposed scheme can support the traceability of user. In terms of dynamic change in organizations or users, we design an efficient and secure method that supports dynamic joining and revocation for organizations and users. The results of the security and performance analyses demonstrate that the proposed scheme can achieve the required security requirements and achieve efficient authentication.

Acknowledgment. The authors acknowledge the support from National Key R&D Program of China under Grant No.2017YFB1400700 and National Natural Science Foundation of China under Grant No.: 61472414, 61772514, 61602061.

References

1. Fisco. https://en.wikipedia.org/wiki/FISCO
2. Hyperledger. https://www.hyperledger.org
3. Ripple. https://en.wikipedia.org/wiki/Ripple
4. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptol. **21**(2), 149–177 (2008)
5. Gope, P., Hwang, T.: Lightweight and energy-efficient mutual authentication and key agreement scheme with user anonymity for secure communication in global mobility networks. IEEE Syst. J. **10**, 1–10 (2015)
6. He, D., Kumar, N., Chilamkurti, N.: A secure temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. Inf. Sci. **321**, 1–6 (2013)
7. Higgins, S.: Inside R3CEV's plot to bring distributed ledgers to wall street. https://www.coindesk.com/r3cev-distributed-ledger-wall-street
8. Lai, C., Li, H., Liang, X., Lu, R., Zhang, K., Shen, X.: CPAL: A conditional privacy-preserving authentication with access linkability for roaming service. Internet Things J. IEEE **1**, 46–57 (2014)

9. Lu, R., Lin, X., Luan, T.H., Liang, X., Shen, X.: Pseudonym changing at social spots: an effective strategy for location privacy in vanets. Veh. Technol. IEEE Transact. **61**, 86–96 (2012)
10. Ni, J., Zhang, K., Lin, X., Yang, H., Shen, X.: AMA: Anonymous mutual authentication with traceability in carpooling systems. pp. 1–6, May 2016
11. Satoshi, N.: A peer-to-peer electronic cash system (2008)
12. Shen, J., Zhou, T., Chen, X., Li, J., Susilo, W.: Anonymous and traceable group data sharing in cloud computing. IEEE Transact. Inf. Forensics Secur. **13**, 1–1 (2017)
13. Swan, M.: Blockchain: Blueprint for a New Economy. O'Reilly Media, Sebastopol (2015)
14. Tsai, J.L., Lo, N.W., Wu, T.C.: Secure handover authentication protocol based on bilinear pairings. Wirel. Pers. Commun. **73**(3), 1037–1047 (2013)
15. Wang, D., Wang, P.: Two birds with one stone: two-factor authentication with security beyond conventional bound. IEEE Transact. Dependable Secure Comput. **15**, 1–22 (2016)
16. Wu, F., et al.: A novel and provably secure authentication and key agreement scheme with user anonymity for global mobility networks: a novel and provably secure authentication and key agreement scheme with user anonymity for global mobility networks. Sec. Commun. Netw. **9**, 3527–3542 (2016)
17. Yang, G., Huang, Q., Wong, D., Deng, X.: Universal authentication protocols for anonymous wireless communications. Trans. Wireless. Comm. **9**, 168–174 (2010)
18. Zhu, J., Ma, J.: A new authentication scheme with anonymity for wireless environments. IEEE Transact. Consum. Electron. **50**, 231–235 (2004)