Energy 180 (2019) 955–967

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy

Blockchain-based decentralized and secure keyless signature scheme for smart grid

Hongwei Zhang [a, c], Jinsong Wang [a, b, c, *], Yuemin Ding [a]

[a] School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China
[b] Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin, China
[c] National Engineering Laboratory for Computer Virus Prevention and Control Technology, Tianjin, China

ARTICLE INFO

Article history:
Received 10 January 2019
Received in revised form
22 April 2019
Accepted 17 May 2019
Available online 23 May 2019

Keywords:
Blockchain
Key management
Keyless signature
Consensus algorithm
Smart grid
2010 MSC
68-M
11

ABSTRACT

Due to the urgent requirement to achieve secure communication between service providers (SPs) and smart meters (SMs), including reliable mutual authentication and privacy credentials, key management is critical in smart grids. Recently, a number of key management schemes have been proposed. However, schemes based on trusted third parties (TTPs) become insecure if the TTP fails. Furthermore, the SPs in most schemes are centralized to manage their respective SMs, which involve a single point of failure. Furthermore, SPs cannot monitor each other for data traceability or security auditing. To remedy these inadequacies, we propose a decentralized keyless signature scheme based on a consortium blockchain to realize more efficient and secure key management. The SM sends requests and receive responses using a blockchain network for data transmission operations. We designed a decentralized secure consensus mechanism that turns a blockchain into an automated access-control manager that does not require a TTP or trust anchor. The SPs of the proposed scheme can keep each other in check using the blockchain. In our concluding remarks, we describe how the proposed scheme incurs smaller computational time costs and is both cost-effective and scalable.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

The development of network and cryptography technologies has paved the way for improving the security and performance of energy systems. Smart grids, which are considered the next generation of power grid systems, are gaining popularity in the fields of industry, research and academia [1]. Smart grids will make building automated energy delivery networks more secure and more efficient. This new infrastructure will exploit bidirectional transmission flows of electricity and data communication. Based on the system model presented by the National Institute of Standards and Technology (NIST), a smart grid contains three parties: energy generators, service providers (SPs) and end user devices (such as smart meters [SMs]) [2,3]. Generally, the three parties take on different roles [4], as indicated in Fig. 1. Energy generators are responsible for producing electricity using renewable and non-renewable resources and transmitting them to the distribution network. By controlling the electricity flow and transactions in the smart grid, SPs manage the electricity distribution and energy trading system [5]. We call an SP and its attendant SMs a service domain. The primary goal of SMs is to monitor energy consumption in real time and provide power pricing information to consumers [6,7].

SMs are usually deployed in close proximity to users homes and are physically protected. However, it is possible for an attacker to gain access to SMs by physically destroying locks, then using the electricity data for criminal purposes [8]. Furthermore, the communication messages between SPs and SMs can be eavesdropped and interrupted due to insecure areas in the wireless environments in which SMs operate, which could lead to breaches of privacy [3,9]. A didactic example is that an attacker can obtain details of a users lifestyle habits by analyzing messages using special tools, to commit crimes such as burglary. As mentioned previously, SMs are major but vulnerable components of the smart grid. Thus, it is important to design secure communication schemes between SMs and SPs, because these can ensure the privacy of SMs, trusted mutual authentication, and secure communication within non-trusted communication environments [10].

* Corresponding author. School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China.
E-mail address: jswang@tjut.edu.cn (J. Wang).

https://doi.org/10.1016/j.energy.2019.05.127
0360-5442/© 2019 Elsevier Ltd. All rights reserved.

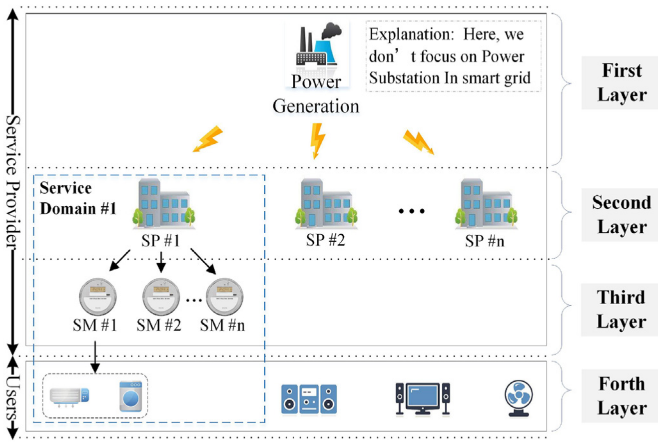

Fig. 1. Traditional network instructure of smart grid.

Cryptography has played a significant role in smart grids since their inception, as it enables optimization of the confidentiality and security of the electricity messages. However, designing an efficient cryptographic model of data transmission in a smart grid is difficult due to the limitations of embedded computing resources in SMs, as described in Refs. [11,12]. Hence, it is essential that we achieve lightweight encryption-oriented communication within smart grids. Over the past decade, many encryption solutions, such as key management schemes, have been presented. However, some of the proposed schemes are based on trusted third parties (TTPs), or trust anchors, some suffer from communication security challenges, some require high storage and computational costs against resource-constrained SMs, and most do not support decentralized data traceability and auditing. In this paper, we propose a new key management scheme for smart grids using the blockchain, which satisfies the security requirements without requiring a TTP or trust anchor. Meanwhile, the new scheme supports authentication record traceability, and the records cannot be tampered with. Furthermore, the nodes (SPs) of the scheme can ensure that the network is secure by mutually supervising the blockchain.

1.1. Our contributions

Although many key management schemes have been proposed recently, most of them are not secure against single points of failure because they use centralized management or weak cryptography. The major contributions of this paper are threefold:

- We analyze the security weaknesses of recently proposed authentication schemes, then propose a new provably secure authenticated keyless scheme for smart grids. SMs and SPs can obtain secure authentication using the Merkle hash tree without any third trusted party. To the best of our knowledge, this is the first time that the technology of blockchain has been used for key management in smart grids. Furthermore, this scheme can improve the reliability of certification and non-repudiation with blockchain technology.
- We propose a consensus algorithm for authentication between SPs. Our analysis shows that the algorithm can ensure the computational efficiency of the system while providing decentralized services.
- We use rigorous formal security analysis to demonstrate that the proposed scheme is secure against common types of attack. Moreover, the performance of the proposed scheme shows that it is suitable for SMs in practical smart grid applications with low computational capabilities.

1.2. Paper roadmap

The remainder of this paper is organized as follows. In Section 2, we discuss some recently proposed approaches to the management of communication keys and message transmission; the blockchain used in this article is also introduced briefly. In Section 3, we describe our system model overview and the process of the proposed scheme, including the secure communication protocol and consensus algorithms based on blockchain for dynamic transaction collection and message authentication. Rigorous formal security analysis is presented in Section 4 to show that the proposed scheme is reliable at preventing attacks. Furthermore, the performance of the proposed scheme is discussed in Section 5. Finally, we conclude the paper in Section 6.

2. Related works

In 2011, a key distribution and management scheme for large customer networks was proposed by Kamto et al. [13], which is based on the Diffie-Hellman (DH) protocol [14] and a group ID-based mechanism [15]. They claimed that their scheme achieved authentication, privacy and data confidentiality. However, the scheme is very computationally costly, and it cannot prevent man-in-the-middle and desynchronization attacks. In the same year, Wu and Zhou [16] proposed a novel key distribution scheme for smart grids to prevent man-in-the-middle and replay attacks. The proposed scheme is based on the symmetric-key Needham-Schroeder authentication protocol and public key elliptic curve cryptography (ECC). However, they both used a trusted anchor and public key infrastructure (PKI) in their scheme. Furthermore, the scheme fails to avoid man-in-the-middle attacks, as pointed out by Xia and Wang [17], who used a lightweight directory access protocol (LDAP) server as the TTP and proposed a novel key distribution scheme for the smart grid. Although Xia and Wangs scheme reduces the operation cost, it has a single point failure, namely the LDAP server, and impersonation attacks and unknown key share attacks cannot be prevented [18]. In recent years, several identity-based authentication schemes have been presented, in Refs. [19–21].

He et al. [19] proposed a data aggregation scheme for a smart grid to protect against internal attacks. Further, Odelu et al. [20] analyzed Tsai-Los authentication scheme [3] and proposed a new, efficient, provably secure and authenticated key agreement scheme for smart grids, to generate credentials for SMs and SPs. However, their scheme requires the involvement of a trust anchor in the authentication process. In Ref. [21], Li et al. proposed a lightweight transmission scheme, which combined a one-time pad mechanism and quantum cryptography. They claimed that the scheme can ensure the security of power data transmission by a quantum random number generator. However, the process of the scheme is too complicated for actual practical deployment. In Ref. [22], Gope-Sikdar used a physically unclonable function (PUF) to propose an authenticated key agreement scheme. They claimed that the proposed scheme offered resilience to DoS. Braeken et al. [23] pointed out the weaknesses of [22] and developed a new provably secure key agreement model for SM communications, which offers the required security features for the smart grid. However, the proposed scheme requires a TTP. Furthermore, Gope-Sikdar [24] used lightweight cryptographic primitives to design a lightweight and privacy-friendly masking-based spatial data aggregation scheme. However, the transponder aggregator (TPA) of the proposed scheme needs to verify the legitimacy of the SMs before receiving messages, so the computational complexity increases linearly with the number of SMs [25]; a similar weakness was also discovered in Ref. [26]. Moreover, the above solutions are based on a centralized SP, which poses a challenge for maintaining the security of the system.

With these considerations in mind, we used a technology known as the blockchain [27] to achieve our security goals. The blockchain is a distributed peer-to-peer network in which users can interact with each other in a verifiable manner without a trusted intermediary. The blockchain has received much attention over recent years [28]. Central managers are removed from the blockchain structure and the public ledger is instead maintained by all of the network participants [29]. Messages are broadcast into the network for nodes to authenticate. Furthermore, the blockchain performs better in terms of robustness under a single point of failure. Some blockchain-based studies have been published recently, which proves the value of this subversive technology [29–31]. In our conception, as a decentralized, tamper-proof and trustless technology [32], blockchain is a disruptive innovation for key management of smart grid systems, and also paves the way for privacy-preserving communication between SMs and SPs. Furthermore, Merkle tree [33,34], which is used in blockchain, is well known for its secure and efficient signature, which can be used to provide authentication between SMs and SPs. We combine these two technologies to achieve an efficient, secure and low-cost key management scheme.

2.1. Blockchain

In this section, we discuss the blockchain that we employed in our scheme.

Over the past few years, blockchain technology has received widespread attention in the academic and industrial fields for its potential for decentralizing the management of distributed systems. The core idea of the blockchain is that it maintains a distributed, authenticated, and synchronized ledger of transactions without administration by the centralizing manager. It can be viewed as a distributed ledger [35], where stored data have strong guarantees of immutability. The blocks store a list of valid transactions that link into a chain by using the hash of the previous block. To provide assurances that the data are tamper-proof, the timestamp, as an essential application, is used to make a blockchain. It is hard to change the content of a previous block because all of the following ones have to be rehashed, as shown in Fig. 2.

Two characteristics are always mentioned in discussions of the blockchain, namely that they are: 1) distributed and 2) decentralized [36]. A new block is registered by a node in the network, which sends it to all of its peers, and each node will verify the new block and then further propagate it. Due to the decentralized characteristic of the system, and the fact that each node needs to save a copy of the block locally, the blockchain system needs to use a consensus algorithm, such as proof of work (PoW) or practical byzantine fault tolerance (PBFT) [37], to ensure that all of the nodes agree on the authenticity and global validity of the block. As shown in Fig. 3, a new block containing the newest transactions is generated by a

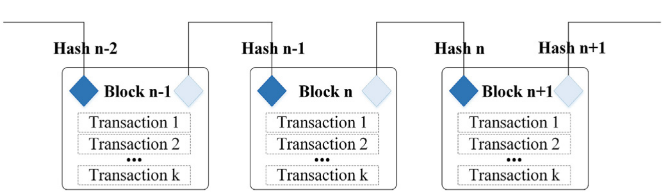

Fig. 2. Data chaining of blockchain.

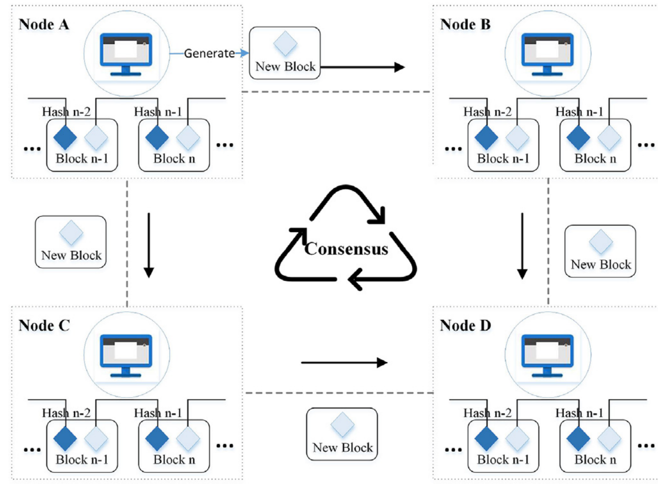

Fig. 3. Block distribution in blockchain network.

node; then, the block is broadcasted to all of the other nodes in the network to be verified and validated. If any incorrect data are detected, the last block will be dropped immediately.

As a disruptive, innovative technology, blockchain constitute a new approach to key management and message transmission in smart grids. We refer to the communications between SMs and SPs as transactions, and specifically identity authentication and message verification between SMs and SPs. These transactions are generated by the SM and sent over the network. The network periodically selects an SP node as a master node to interact with the SM, to validate the system via identity auditing and broadcasting of the transactions. All of the above transactions use their generated keys for secure communication. All of the transactions from the SMs are stored in a blockchain, which is in turn stored by the SPs.

3. Proposed framework

In this paper, we propose a blockchain-based keyless authentication architecture for secure communication and signature generation between SMs and SPs in the smart grid. The notations catalogued in Table 1 are used for the proposed scheme. First, we describe the system components in Section 3-A. Then, in Sections 3-B to 3-D, we introduce the scheme based on a detailed example of communication between an SM and its corresponding SP.

3.1. System components

We focus exclusively on the authentication of secure communication between SMs and SPs, especially when there are no TTPs in the smart grid, as shown in Fig. 4. Therefore, our scheme is composed of the following elements: **1) a set of remote SMs; 2) a peer to peer network connected by multiple SPs; and 3) a signature device infrastructure**. An SM is a solid-state programmable device that collects sensor data, such as electricity consumption, real-time electricity load, etc. for the corresponding SP. In our scheme, it is essential for the SM to send the ID of the corresponding SP to verify its manager. As a data aggregator, an SP is in charge of several SMs, and is responsible for maintaining the electric flows and other public information concerning these SMs, and for sending information to SMs in real time. In our scheme, the keys are supposed to be generated and stored in a dedicated facility with specific privacy and security functions, namely, the Signature Device Instruction (SDI). Hence, the SDI is accessed under two situations, as follows:

Table 1
Notations used in this paper.

Symbol	Description
SM, SP	The smart meter an the service provider in the proposed scheme
TTP, TA	the trusted third party for key management
SPN	The network composed of service providers by P2P
SP_i, ID_{SP_i}	The i-th service provider in BKSS network and its identity ID
SM_{ij}, $ID_{SM_{ij}}$	The j-th smart meter managed by SP_i and its identity ID
SDI, ID_{SDI}	The signature device instruction and its identity ID
PK_{SP_i}, SK_{SP_i}	The SP_i's public key and private key
$h(*)$	The secure one-way hash function
$Bin(*)$	The binary representation
m	The message in the communication between SP_i and SM_{ij}
Am	The prior license agreement between service providers
M_{ident}	The identity information sent by SP_i
$Sig(*)_{\#}$	The signature operation for $*$ using $\#$
T_s	The hash-tree time stamp created by a service provider
t_0	The validity time of the key
M_r	The root message generated by SM_{ij} for signature verification
Sig_m	The signature of the collecting message
SP_l	The leader selected by Algorithm 1
$Trans$	A transaction generated in SM_{ij}
$En(*)_{\#}$	The encryption operation for $*$ using $\#$
$\mathscr{A}$	The adversary
f	The fail nodes number controlled by $\mathscr{A}$
Num_{SP}	The number of service providers in SPN
Num_{SM}	The number of smart meters managed by SP_i
T_{inter}	The SMs' interaction cycle with the SPN (min)
T_{certi}	The certification cycle of hash chains in each SM with SDI (year)
S_{totle}	The whole storage cost of SP_i
l_*	The corresponding data length
Ψ_*	Computation time cost of $*$
ω	Time cost of executing a hash chain traversal algorithm
φ	Time cost of executing a hash operation
ζ	Time cost of signature generation operation
η	Time cost of transaction geration operation
Γ	Time cost of v_i certification
Υ	Time cost of root message geration operation
Θ	Time cost of transmission of consensus messages between SPs
Δ	Time cost of block verification algorithm
Δ_{Ts}	Time cost of timestamp geration operation
Δ_{vote}	Time cost of counting operation

Remind: The i in this paper has two types of ranges: $i = 2, 3, \ldots$ when referring to v_i and $i = 1, 2, \ldots$ in other cases.

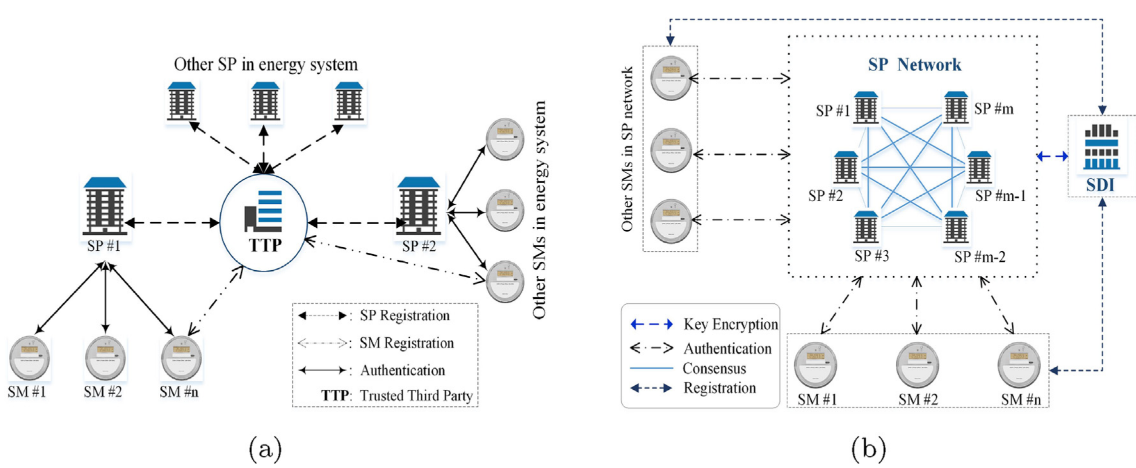

Figure 4. Different network structures:(a) Traditional key management instructure with TTP; (b) Blockchain keyless instructure without TTP.

1. *Initial Registration*: new SMs need to write the hash-chain for the initial registration when they join the network and first participate in the system.
2. *Change the hash-chain Information*: it is necessary for SMs to periodically change their hash-chains, so they need to contact the SDI to generate a new chain of one-time passwords.

Similar to blockchain applications in the digital currency field, such as Bitcoin, Ethereum, etc., the blockchain used in this paper mainly functions to provide a decentralized information sharing model between nodes of the network, namely the SPs. These are connected to each other by a peer-to-peer network. When an SM sends a message to the network, a node is selected to encapsulate it into a transaction. Multiple transactions within a specified time period are packaged into a block, which will then be broadcast into a network for consensus verification of other nodes. As a result, the selected node will issue the block on the blockchain after sanctioning the verification.

3.2. Key generation phase

We now describe SP_i and SM_{ij} key generation.The SP_i key generation proceeds according to the following steps:

Step 1: SP_i produces a random number larger than 256 bits by using the underlying operating systems random number generators and uses the hash algorithm $h(random\ number)$ to get a 256-bit number, here we use SHA-256 algorithm;
Step 2: Verify the result. If the result satisfies the condition $1 < the\ 256 - bit\ number < n$, where $n \approx 2^{256}$, it is specified as the private key SK_{SP_i}, otherwise, return to Step 1.
Step 3:SP_i produces the PK_{SP_i} by using secp256k1 elliptic curve multiplication, which is irreversible like Equation (1):

$$PK_{SP_i} = SK_{SP_i} \times G \tag{1}$$

where G is a constant point specified as part of the secp256k1 standard called the generator point.

The relationship between SK_{SP_i} and PK_{SP_i} is fixed, but can only be calculated in one way, from SK_{SP_i} to PK_{SP_i}. Each SP contains a series of key pairs, each consisting of SK_{SP_i} and PK_{SP_i}. The SK_{SP_i} is used to verify the signature for the node identity. The SK_{SP_i} must be kept secret at all times and must be backed up and protected so they are stored on SDI encrypted.

SDI generates a hash chain with reverse order of indices for SM_{ij} which is managed by SP_i as follows:

Step 1: SDI chooses a random number as the chain's seed using Equation (2) as follow:

$$v_n \leftarrow \{0,1\}^* \tag{2}$$

in which, n is the number of one-time passwords, i.e. the number of preset authentication sessions, at once, let:

$$v_0 = Bin(ID_{SP_i}) \tag{3}$$

Step 2: with Equation (3) in Step 1, SDI generates a key-hash chain $v_1, v_2, \ldots, v_n$ by $v_i \leftarrow h(v_{i+1})$ where $i \in \{1, 2, \ldots, n\}$, note that every hash value, as a one-time authentication key, just works at particular second of time.
Step 3: SDI then processes the chain into a secure shape with Merkle tree against premature disclosure of key. Specifically, let $v_3, v_5, \ldots$ with odd indices except v_1 be replaced respectively with Equation (4). The root r of the Merkle tree, which is an important element in the communication between SM_{ij} and SP_i, is also computed in this step, as shown in Fig. 5.

$$\begin{array}{l} r_i = h(v_i) = v_{i-1} \\ i \in \{1 < i \le n, i mod 2 \ne 0\} \end{array} \tag{4}$$

Due to the one-way feature of the hash algorithm, the key generation of the SPs is secure as long as the SK_{SP_i} is managed properly. Furthermore, we assume that the SDI is immune to cyber-attacks, and the generation of hash-chains is executed locally so it is also secure. Details of this phase are summarized in Fig. 6.

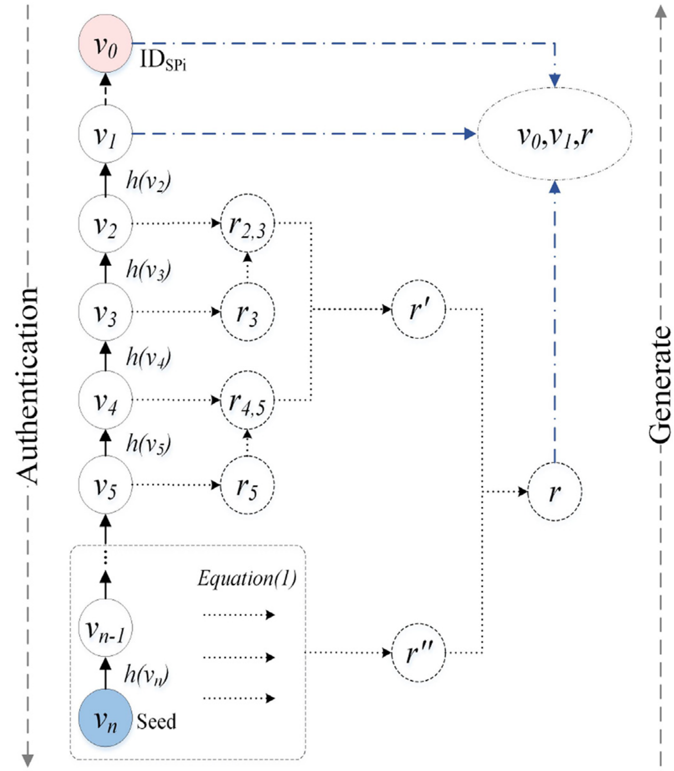

Fig. 5. Key hash-chain with Merkle tree.

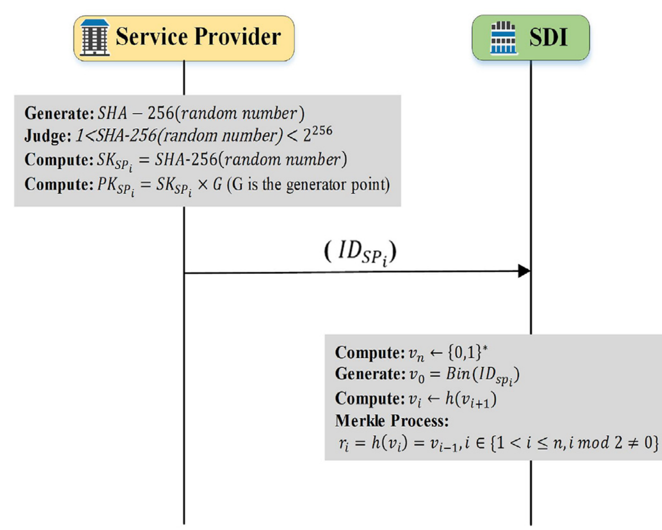

Fig. 6. Key generation phase.

3.3. Registration Phase

In this phase, we explain the registration process of SP_i and SM_{ij}, with which SP_i and SM_{ij} will mutually authenticate and establish a secure session without SDI. The detailed description of the process is as follows:

i) Service Provider Registration: Compared to the public blockchain for bitcoin, the proposed scheme uses consortium blockchain technology. In the architecture of a consortium blockchain, node failure or misuse has to be considered. Therefore, the number of system nodes is usually set in advance for data synchronization and

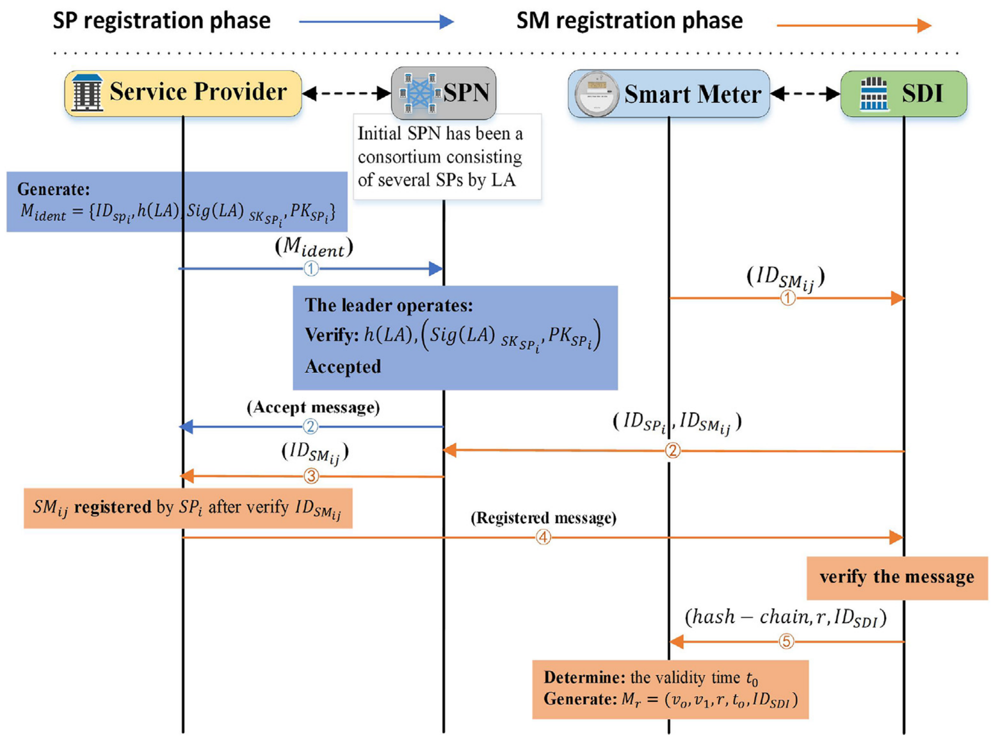

Fig. 7. Registration Phase of the proposed scheme.

is more than $3F+1$ (F is the number of faulty or malicious nodes), i.e., at least 4. In our scheme, several initial SPs (we assume that the number of SPs is no less than 4) form a consortium blockchain through a prior license agreement (denoted as LA). After this process, each new SP will be verified by the service provider network (SPN).

Step 1: SP_i sends its identity messages M_{ident} like Equation (5) to SPN:

$$M_{ident} = \left\{ID_{SP_i}, h(\mathrm{Am}), Sig(\mathrm{Am})_{SK_{SP_i}}, PK_{SP_i}\right\} \tag{5}$$

Step 2: The current leader (a master node for consensus) in the network is elected to compared $h(\mathrm{Am})$ and $Sig(\mathrm{Am})_{SK_{SP_i}}$ with its own agreement's hash and PK_{SP_i};
Step 3: Finally, the SP_i joins the SPN as a new node after the compliance verification.

ii) Smart Meter Registration: for the purpose of acquiring certification credentials and registering with SPN, the smart meter SM_{ij} executes the following steps:

Step 1: SM_{ij} sends $ID_{SM_{ij}}$ to SDI, then SDI sends $\{ID_{SP_i}, ID_{SM_{ij}}\}$ to SPN, SP_i registers $ID_{SM_{ij}}$ after all the nodes check the ID_{SP_i}. Then SP_i sends the registered message to SDI. Next the hash-chain generated in the above SDI setup phase and other parameters including r and ID_{SDI} is written into SM_{ij} from the SDI with physical medium.
Step 2: After written the information from SDI, SM_{ij} determines the validity time t_0 after which the certificate becomes valid. In other words, v_2 is intended to sign documents at time t_0+1, v_3 is for signing at t_0+2, etc.
Step 3: Upon the above steps, SM_{ij} generates the root message M_r ultimately, as shown in Equation (6):

$$M_r = \{v_0, v_1, r, t_0, ID_{SDI}\} \tag{6}$$

which is used for signature verification between SM_{ij} and SPN. Due to the regional management characteristics of the smart grid, v_0 is used to recognize the transaction of SM_{ij}. In this phase, the SP_i only requires verification from the leader of the SPN. Achieving this registration makes the communication time and cost acceptable, as well as the corresponding SM_{ij} of SP_i. This process is shown in Fig. 7.

3.4. Signing and verifying phase

in this phase, the SM_{ij} will send the collected sensor data to the SPN securely and frequently. In contrast to the traditional scheme, the data receiver is not the corresponding SP_i but rather the SPN, and each communication authentication will be permanently aggregated as a transaction on the distributed replicated ledger.

Step 1: SM_{ij} computes a hash of the collecting message $p = h(m)$, and then generates the message like Equation (7),

$$q = h(p, v_i) \tag{7}$$

At last, the message generated by SM_{ij} in Equation (8):

$$\left\{q, v_0, ID_{SM_{ij}}\right\} \tag{8}$$

is sent to SPN.

Step 2: Upon receiving the message in Equation (8), SPN selects a service provider SP_l using Algorithm 1 from the nodes as a leader, the leader checks if the $ID_{SM_{ij}}$ has been registered in SP_i and if so, the leader creates a Merkle tree time stamp T_s for $\{q, v_0, ID_{SM_{ij}}\}$ and sends $\{T_s, PK_{SP_l}\}$ to SM_{ij}.

Algorithm 1 Leader Election

1 $SP_i \rightarrow Follower(i \in 1,2,\ldots,n)$, where $n > 3f+1$, f is the fault nodes number;
2 Set the tenure number to 0, $TN_{SP_i} = 0(i \in 1,2,\ldots,n)$;
3 Set the original number of votes to 0, $N_v = 0$;
4 Start the Timer, set a random timeout T_{out};
5 **while** $Timer > T_{out}$ **do**
6 $Follower \rightarrow Candidate$;
7 $TN + 1$;
8 start the new Timer;
9 $N_v + 1$;
10 Send a request of voting to all other nodes and wait for the reply votes;
11 **if** Receive votes reply **then**
12 Computes the N_v again;
13 **if** $N_v > n/2+1$, where n is the nodes number **then**
14 $Candidate \rightarrow Leader$;
15 **end if**
16 **else** [Receive leader confirmed]
17 $Candidate \rightarrow Follower$;
18 **else**
19 Repeat steps 7–10 for a new election;
20 **end if**
21 **end while**

Step 3: After receiving $\{T_s, PK_{SP_l}\}$, SM_{ij} generates the signature of the collecting message:

$$Sig_m = \left\{ID_{SM_{ij}}, T_s, v_i, i, C_i\right\} \tag{9}$$

is the i-th element of the hash-chain. Then SM_{ij} sends a transaction to SP_l like Equation (10), and the transaction format is shown in Table 2.

$$Trans = \left\{En(m)_{PK_{SP_l}}, Sig_m, M_r\right\} \tag{10}$$

Step 4: the leader SP_l uses SK_{SP_l} to get the message m and generates a block, as shown in Table 3, in which contains the new transaction like Equation (10). Then SP_l broadcasts the block to other nodes for verifying with Algorithm 2.

Table 2
Transaction format.

Transaction Header
Hash result of the transaction
The root message M_r generated by SM_{ij}
The signature to ensure integrity and authentication $Sig_m = \{ID_{SM_{ij}}, T_s, v_i, i, C_i\}$
Payload: The encrypted message $En(m)_{PK_{SP_l}}$

Table 3
Block format.

Block Header	
Version	Block version number
Previous Block Hash	Hash of previous block in the chain
Merkle Tree Root	Root hash of the transactions merkle tree
Timestamp	Creation time of this block
Block Payload (Transactions)	
Transaction 1	
Transaction 2	
...	

Algorithm 2 Consistency Verification

Every follower receives the block from the leader $B = \{H_{pre}, BS_t, Root_M, Trans\}$ and verifies as follows:
for Each follower SP **do**
$ID_{SM_{ij}}$ in Sig_m coincides with in M_r;
ID_{SDI} in M_r coincides with the local;
Computes the root hash value: $r = v_i + C_i$;
Extract the time from T_s;
if $ID_{SM_{ij}}(Sig_m) = ID_{SM_{ij}}(M_r)$, $ID_{SDI}(M_r) = ID_{SDI}(Local)$, $r = v_i + C_i$ and $t = t_0 + i - 1$ **then**
Send Validated (block) to leader;
end if
end for
Leader initializes a parameter V to denote the Validated (block) from the followers, $V = 0$;
When a Validated (block) is received, $V = V + 1$;
if $V > 2f + 1$, where f is the fault nodes number **then**
Leader sends Committed to followers;
All the followers which receive the Committed Add the block to the blockchain.
end if

Step 5: Last but not least, each node that received the new block verifies the parameters (shown in Table 3)of the block and confirms the integrity of the block. Finally, the new block will be added to the chain's end to form the latest blockchain. Fig. 8 shows the particular process of signing and verifying phase.

Algorithm 2 ensures that verification of the block can be completed with most nodes, so that they can still interact with the meter even if some nodes fail [38]. When an attack occurs, the blockchain is protected and the message cannot be tampered with by the attacker. (1) All nodes must be deterministic. That is to say, in the case where the given state and parameters are the same, the result of the operation execution must be the same; (2) all nodes must be executed from the same state.

It is obvious that the PBFT algorithm only scales to a few tens of nodes, as it needs to exchange $O(n^2)$ messages to reach consensus on a single operation between n servers [39]. Thus, enhancing the scalability of the proposed scheme is essential for ensuring practical deployment with efficient and extensible abilities. To achieve this objective, we proposed a voting algorithm to elect some nodes as accounting nodes. These accounting nodes exercise block-generating and consensus rights. The voting algorithm is as follows.

Algorithm 3 Accounting Nodes Voting

Every node SP_i has some votes, i.e., the number of smart meters within its jurisdiction, denote as: N_i, the Voting process is as follows:
for Round i **do**
Each node set the votes N_i to the SPs it trusts (it could votes for multiple nodes except itself);
Each node computing the votes V_i;
Get 21 delegates sort by votes: $list_i$;
Shuffle the $list_i$;
Select k of the remaining nodes as alternative nodes:$Alist_i$, where $k \le 10$;
if A node in $list_i$ fails **then**
Select a node in $Alist_i$ Randomly and add to $list_i$;
Shuffle the $list_i$;
end if
end for

When the number of nodes in the proposed scheme is greater than 21, the system will execute Algorithm 3 to vote 21 nodes as accounting nodes. Other nodes are responsible for collecting and forwarding related data from the *SMs* that they manage. In Algorithm 3, a round indicates a cycle of voting, which can be adjusted according to the requirements of the system. Whenever we move to the next round, the system will repeat the voting algorithm for a new list to ensure the fairness and security of the system consensus. Furthermore, an alternative node mechanism is also designed to ensure that the number of accounting nodes remains constant. When an accounting node fails, it should be replaced by a random node from the k alternative nodes.

4. Security analysis

In this section, we describe the security analysis of the proposed scheme as follows.

4.1. Key generation

The SP keys are generated using a random number generator, a hash algorithm, and an elliptic curve function. Although the random number generator is initialized by a human source of randomness, which could be exploited by an attacker, the hash algorithm and elliptic curve function provide a more secure method, and because all of the SPs keys are random and independent, the SPN can provide further authentication. Hence, each SPs key is secure. An SM key is a chain created based on a Merkle tree and a timestamp using a hash function, and is written to the SMs storage by physical means. The SM key is secure.

4.2. Registion and key replacement

In the proposed scheme, the auto-refreshing of the SM key depends on the time t. If it's v_i's turn to be used for signing, and it is used immediately before t_0+i-1, the probability of abuse of v_i is 0. However, if v_i is used for a sufficiently long time before t_0+i-1, v_i can be abused by anyone who has the required signature. Hence, the security measures of the scheme needs to guarantee that it remains viable, namely that the other parties will not obtain v_i until the signer verifies the signature. Due to the condition that $t=t_0+i-1$, the addition to the blockchain occurs after t, it is secure to disclose v_i. The new keys are time-dependent on the old keys, which guarantees their security.

4.3. Message integrity and authentication

When an SM communicates with an SPN, it holds its own sessions keys. When the SPN receives requests, a leader will be selected first, and the leader will then verify the signature of the encrypted data using its private key. The leader decrypts the message only when it passes the authentication. Otherwise, the message will be discarded. The authentication and integrity of the message transmission is ensured by the freshness of the SM keys and the randomness of leader selection.

4.4. Block verification

The security of block verification in the proposed scheme is proven by the PBFT algorithm [38]. It is well known that a smart grid is an asynchronous distributed system, in which the failure of a single node is an independent event. We assume that there is an attacker A that can control several nodes (denoted f) in the SPN, thus enabling malicious consensus. In the proposed scheme, this means that the leader must make a judgment after communicating with the *nf* nodes, because the f fail nodes controlled by A are likely to send erroneous or responses, or no responses. However, the leader still needs a sufficient number of responses from non-failed

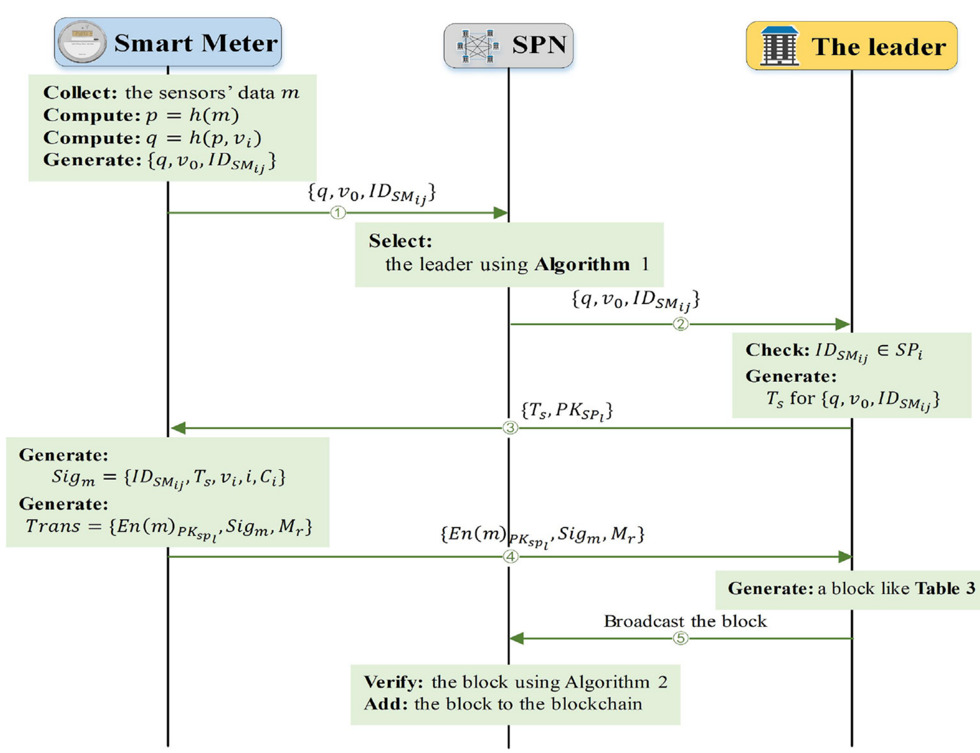

Fig. 8. Signing and verifying Phase of the proposed scheme.

nodes, and the number of responses from non-failed nodes must exceed those from failed nodes, i.e. $n2f > f$, thus resulting in $n > 3f$. It must be guaranteed that there are at least $3f + 1$ nodes, to ensure that the security and activity of the asynchronous system meet the requirements of the proposed scheme.

4.5. Comparison based on security features

To assess the security features of our scheme, we compared its performance to that of other schemes in terms of defending against well-known attacks in Table 4. The proposed scheme provides mutual authentication without the help of a TA, unlike Wu and Zhous scheme [16]. Abbasinezhad-Mood and Nikooghadam's scheme [12] is less computationally costly than our scheme, but it does not provide decentralized key management for strong privacy and security, or tamper-proof messages. These are the same weaknesses as in Refs. [20,22]. Furthermore, the scheme proposed in Ref. [20] cannot provide high privacy for SMs. Guan et al. [30] used blockchain technology to design a scheme for data aggregation in the smart grid. However, this requires a TA to generate keys. Compared to these existing schemes, our scheme includes tamper-proof messages and an efficient consensus-based solution for authentication and message transmission, as shown in Table 4.

Table 4
FEATURE-BASED comparison with the related schemes.

	F1	F2	F3	F4	F5	F6	F7	F8	F9
[16]	✓	×	✓	×	×	×	×	×	×
[20]	✓	✓	✓	✓	✓	✓	×	×	×
[12]	✓	✓	✓	✓	✓	✓	✓	×	×
[22]	✓	✓	✓	✓	✓	✓	✓	×	×
[30]	✓	✓	✓	×	×	✓	✓	✓	✓
Ours	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note:F1:Impersonation attack resistance [18]; F2:Man-in-the-middle attack resistance [17]; F3:Reply attack resistance [17]; F4:Providing mutual authentication without the help of TA [3]; **F5**:Providing perfect forward secrecy [40]; F6:Unknown key share attack resistance [18]; F7:Strong privacy of smart meters [20]; F8:No any centralizing manager [29]; F9:Providing messages tamper-proof [29].✓: The scheme supports that feature or it is secure. × : The scheme does not support that feature or it is insecure.

5. Performance evaluation and comparison

The proposed scheme was simulated on a PC running the Ubuntu 16.04 LTS operating system with an Intel Core 3.40 GHz i7-6700 CPU and 16 GB of RAM. A consortium blockchain network was deployed with Go Ethereum (Geth 1.7.2) on another machine with the same configuration, i.e., the nodes were all running on the same machine. Furthermore, we used the MIRACL library (a cryptographic library with many practical applications [41]) for computation.

5.1. Storage cost

In terms of communication within the proposed scheme, SMs and SPs should store data including authentication keys, timestamps and additional values. The data to be stored by SPs and SMs are summarized in Table 5.

Due to the different data storage requirement indicated in Table 2, the two types of communication modalities, SP_i and SM_{ij}, have their own methods for calculating the storage cost, as follows. For SP_i, the secp256k1 ECC algorithm (a type of asymmetric encryption algorithm) is used to generate PK_{SP_i} and SK_{SP_i}, The key has a length of 256 bits, which is shorter than RSA and DSA keys. The timestamp is set to 10 bits, and the LA is set to 32 bits. The transactions, which are an important part of the storage cost, depend on the number of SMs managed by SP_i (denoted as Num_{SM}) and the frequency of interactions with the SMs. Here, we assume that the SMs interact with the SPN every minute, denoted T_{inter}. As mentioned previously in Section 3, the length of the hash-chain hinges on T_{inter} and the certification cycle of the SDI (denoted T_{certi}). Then, the whole storage cost within S_{totle} is calculated using Equation (11):

$$S_{totle} = l_{PK_{SP_i}} + l_{SK_{SP_i}} + l_{timestamp} + l_{Am} + l_{Trans} \times Num_{SM} \times \left(T_{certi} / T_{inter} \times 2.628 \times 10^6 \right) \tag{11}$$

where l is the data length. For SPs, specialized data management servers can be used as storage for keys and other related data, whereas the storage ability of SMs is limited.

As for SM_{ij}, the maximum possible storage cost of each SM should be evaluated according to the length of the hash chain, as shown in Table 2. We issue a chain generated using Secure Hash Algorithm 256 (SHA-256) and the certification cycle of SMs is no more than 10 years. Fig. 9 shows the storage cost as a function of the interaction cycle of SM with the SPN and T_{certi}, which ranges from 1 to 10 years. We observed that the storage costs increased under the short interaction interval and longer certification interval. Another cost curve plotted against a different value of T_{inter} is shown in Fig. 10. We can easily see that the slope of the curve decreases with increasing T. Therefore, as long as T is within reasonable limits, the key chain of the SMs can be used for a certain period without certification. Furthermore, Table 6 shows the storage cost for SMs within T_{inter} and T_{certi}. we can see that the storage cost for each SM will increase with T_{certi} but decrease with T_{inter}. In a normal situation, the maximum storage cost of each SM is 0.3 KB. As mentioned previously [12,42], the SMs usually send usage reports at 9001,800 s time intervals. Thus, this result is acceptable.

5.2. Computational time cost

Due to the time limit of the message transmission, it is essential to analyze the time cost for maximum computation at a given time. We will describe the computational time cost calculation method and results separately, as follows.

1) Computational Time Cost of Each SM: Generally, the SM is always implemented by embedded systems. According to the processes of key management between the SMs and the SPN described in Section 3, the method for calculating the computational time cost of each SM can be obtained from Equation (12), Here, we denote the computational time cost of each SM as Ψ_{SM} :

$$\Psi_{SM} = \omega + 2\varphi + \zeta + \eta \tag{12}$$

where ω is based on $O(\log_2 l_{hash-chain})$. The operation rate for hash functions and hash sequence traversal is approximately 1050 Mb/s. The signature and transaction generation operation relies on simple calculations for data packing; the computational costs are so small that we will not consider them. We analyzed the computational time costs by setting different T_{inter} values, then calculating the computational time cost for each SM, which are listed in Table 7.

Table 5
Related data stored in the SP and SMS

	Service provider (SP_i)	Smart meter (SM_{ij})
Authentication Kyes	PK_{SP_i}, SK_{SP_i}	$v_0, v_1, \ldots, v_n$
Timestamp	T_s	T_s
Additional values	LA, $Trans$	r

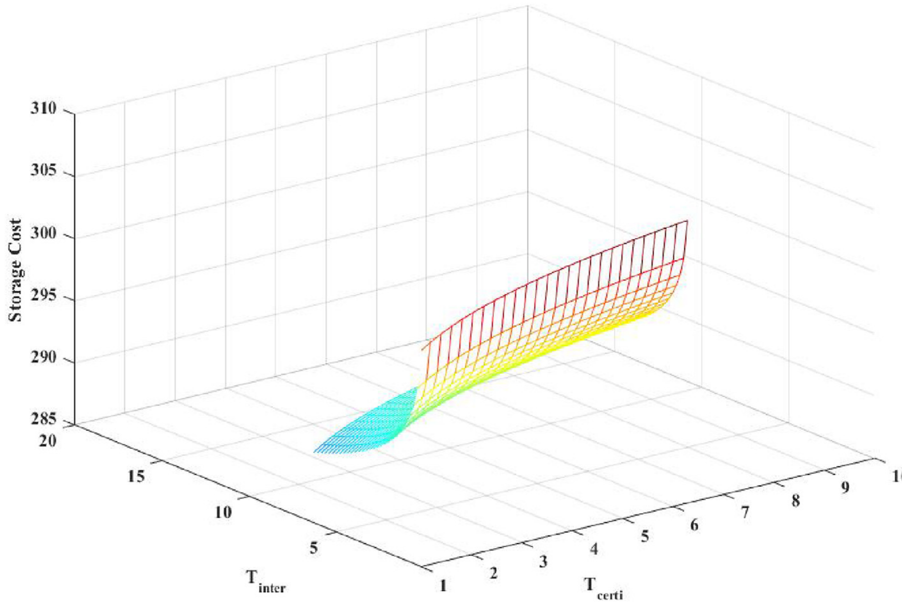

Fig. 9. Storage cost with the parameters.

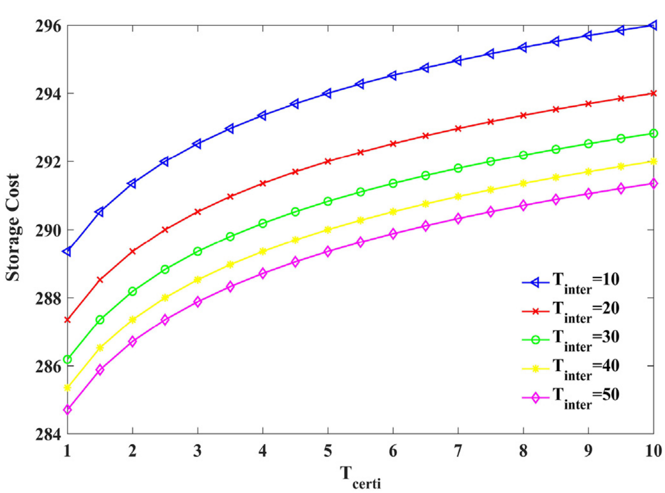

Fig. 10. Different curves of storage cost with different.T_{inter}

According to the results, the computational time cost for each SM is too small to analyze the variation between them, and the effect of message transmission time is marginal.

2) Computational Time Cost of Each Service Provider: the scheme in this paper uses SPN and forms a federated network with multiple SPs. All of the communication messages must be agreed through consensus among the nodes in the network, and we use the PBFT-based consensus algorithm. The computational time costs of the SPs include the leader election operation, communication between the leader and the SMs, and the follower verification operation. As all three of these operations are carried out by the leader, and the verification operations carried out by the followers are mainly used to verify the transaction, the time cost is much smaller as we only need to analyze the computational time cost of the leader. We must also make some assumptions:

1. The leader is already chosen before the block transaction starts, and does not change during the execution of the operation steps for a single block.
2. The rate of message processing by each follower is the same.
3. Followers do not fail at any time during the execution of a single block.
4. The rate of message transmission between all nodes is the same.

If needed, the mentioned assumptions can be relaxed. Equation (13) is used to calculate the computational time cost of the leader:

$$\Psi_{SP} = \Theta + \Upsilon + \Gamma + \Lambda \tag{13}$$

where

$$\Lambda = \varphi \times Num_{SM}$$
$$\Theta \geq \Delta_{vote} \times \left(\frac{Num_{SP}}{2} + 1\right) \tag{14}$$

In Equation (13), Λ and Θ should satisfy the condition specified in Equation (14). Furthermore, the rate of Υ can be ignored. And the rate of the asymmetric encryption keys can also be ignored because of the generation before the system transaction. The PCI cryptographic coprocessor can be used to execute the computations of the SP. The operation rate for hash functions is approximately 50 Mb1 Gb/s. Assuming that all of the nodes are equidistant from each other, we set some fixed parameters as follows: $T_{inter} = 30$ and

Table 6
Storage cost examples of each SM.

T_{inter} (min)		10	20	30	40	50
Storage Cost (KBytes)	$T_{certi} = 1$(year)	0.289	0.287	0.286	0.285	0.284
	$T_{certi} = 5$(year)	0.294	0.292	0.290	0.290	0.289
	$T_{certi} = 10$(year)	0.296	0.294	0.293	0.291	0.289

Table 7
Computational time cost in each smart meter.

T_{inter} (min)		10	30	50
Time cost in each SM (ms)	$T_{certi} = 1$	11.56~57.80	11.44~57.20	11.36~56.80
	$T_{certi} = 5$	11.76~58.80	11.60~58.00	11.56~57.80

Table 8
Computational time cost examples of the leader.

Num_{SM}		1000	2000	3000	4000
Time cost of the leader (ms)	$Num_{SP} = 5$	51.4	102.6	154.0	205.9
	$Num_{SP} = 10$	51.7	105.0	157.7	210.1
	$Num_{SP} = 15$	52.9	110.9	165.4	216.9
	$Num_{SP} = 22$	55.2	116.1	178.3	231.7
	$Num_{SP} = 40$	63.2	125.2	191.2	243.5
	$Num_{SP} = 50$	72.1	134.6	204.2	265.0

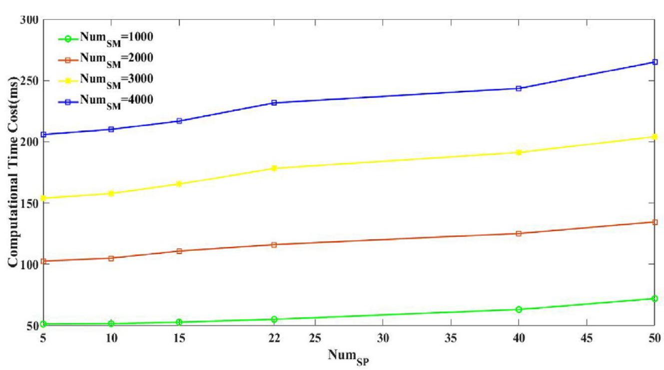

Fig. 11. Computational time cost with different.Num_{SM}

$T_{certi} = 5$. Based on this analysis, Fig. 11 shows the trend in the computational time cost under different Num_{SM}. We can easily see that the trends are steep when there are 22 nodes and became moderate after reaching a transition point. This is because the slowdown in message processing has a greater impact on the average time taken to reach a consensus than the slowdown of the selection leader. We calculated the computational time costs and summarize the results in Table 8.

Based on these results, we found that the time costs of SPs in the SPN depend on the values of Num_{SM} and Num_{SP}. Obviously, when the number of SPs is greater than 21, there will be an inflection point with the calculation time, as show in Fig. 11. As the system will execute Algorithm 3 to select 21 accounting nodes from among all of the SP nodes, all of these messages will be verified by the 21 nodes rather than all of the nodes in the network. Furthermore, the time complexity of the algorithm is also small and increases much more rapidly with the value of Num_{SM} than with Num_{SP}. Than with Num_{SM} is set to 10,000, the time complexity will not seriously affect the transmission of messages.

3) Total time complexity: in this paper, the total time complexity of the system includes the computational time of the SMs, the election of accounting nodes in SPN (if the number of all nodes is greater than 21), election of the leader in the SPN and the message forwarding and consensus processes. In other words, the total time complexity is closely related to the system performance, as we ignore the time effect of the communication delay. To compare the performance of the proposed scheme with that of other schemes, we used a case wherein Num_{SM} and Num_{SP}, where $Num_{SM} = 1000$., $Num_{SP} = 22$. As shown in Fig. 12, the total computation time for [22] is lower than that of the other schemes. However, it cannot guarantee most of the important security features listed in Table 4, which are important for smart grid security. Due to the verification operations, the computational time cost for [30] is much higher than that of other schemes. In contrast, our proposed scheme can ensure all of the important security features, including decentralized management and tamper-proof messages. Although the

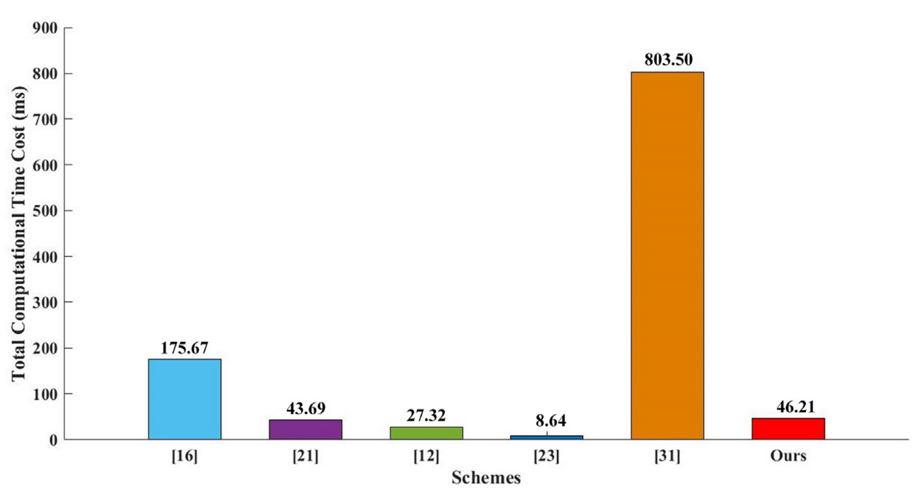

Fig. 12. Comparison based on computational time.

computation performance is a little higher than [20], our scheme has significantly lower computational costs than [16]. Hence, it is suitable for ensuring a secure service between SPs and users (SMs).

5.3. Other discussions

As a new blockchain-based key management scheme in a smart grid, it is necessary to further discuss the scheme in terms of the characteristics of smart grids, including accuracy, effectiveness and efficiency, and practical applications.

In terms of accuracy: we first assumed that the SDI is secure. In the case of SMs, hash-chains are designed based on a one-time password and timestamp. Meanwhile, they provide forward security through the irreversibility of the hash function. The SPs form a consortium network, which realizes identity and message verification through a decentralized consensus algorithm. Furthermore, the historical data can be protected by a backup blockchain. Tampering can only be achieved by simultaneous attacks on several nodes within the network (see the security analysis in Section 4. Hence, this solution is accurate.

In terms of effectiveness and efficiency: Ethereum is currently the most popular blockchain development platform. Thus, we used it for simulations and comparison experiments, as well as for comprehensive evaluation of the performance of the system. From our analysis of the experimental results, we can see that the proposed scheme is effective. Furthermore, the proposed scheme uses blockchain technology to realize decentralization, and includes a consensus mechanism to ensure the reliability and consistency of the system. The time complexity of the consensus algorithm has a significant impact on the system performance, as shown in Fig. 11. From Fig. 11, it is obvious that there is an inflection point at 22. It is obvious that there is an inflection point at 22 nodes. This is because, when the number of nodes exceeds 21, the main factor affecting the system performance is no longer the consensus authentication of the node, but the distribution of messages and the overall message volume. This issue needs to be analyzed and improved in our future work.

6. Conclusion

In this paper, we proposed a novel key management scheme for SMs and SPs in smart grid systems. To solve the centralization and data-tampering problems, we introduced the concept of the blockchain and optimized the performance of the SMs using a Merkle tree. The proposed blockchain structure allows messages to be transmitted securely within the decentralized SPN. We developed an effective data consensus method to reduce the message authentication time of the blockchain scheme. Two components are discussed: 1) a blockchain-based keyless signature scheme and 2) a dynamic transaction consensus scheme. First, we studied key management schemes in smart grids and analyzed the weaknesses of recently proposed schemes. Second, a more efficient and robust structure was presented. Furthermore, we compared our scheme to related schemes in terms of both communication and computational costs and concluded that the proposed scheme is feasible. In the future, we will extend our work to optimize the consensus algorithm and its efficiency, and improve message collection and distribution. Moreover, SMs may be able to choose to self-certify by considering the tradeoff between security and privacy.

Acknowledgment

We would like to express our gratitude to Tianjin University of Technology that funded our research. This work was in part supported by Key Program of Tianjin Municipal Natural Science Foundation (18JCZDJC30700) and National Natural Science Foundation Of China (61702369).

References

[1] Bayindir R, Colak I, Fulli G, Demirtas K. Smart grid technologies and applications. Renew Sustain Energy Rev 2016;66:499–516. https://doi.org/10.1016/j.rser.2016.08.002.
[2] NIST. Guidelines for smart grid cybersecurity, nISTIR 7628 Revision 1. 2014. http://nvlpubs.nist.gov/nistpubs/ir/2014/NIST.IR.7628r1.pdf.
[3] Tsai J-L, Lo N-W. Secure anonymous key distribution scheme for smart grid. IEEE Trans Smart Grid 2016;7(2):906–14. https://doi.org/10.1109/TSG.2015.2440658.
[4] Wang K, Ouyang Z, Krishnan R, Shu L, He L. A game theory-based energy management system using price elasticity for smart grids. IEEE Trans Ind Inf 2015;11(6):1607–16. https://doi.org/10.1109/TII.2015.2426015.
[5] Jalali MM, Kazemi A. Demand side management in a smart grid with multiple electricity suppliers. Energy 2015;81:766–76. https://doi.org/10.1016/j.energy.2015.01.027.
[6] Liu H, Ning H, Zhang Y, Xiong Q, Yang LT. Role-dependent privacy preservation for secure v2g networks in the smart grid. IEEE Trans Inf Forensics Secur 2014;9(2):208–20. https://doi.org/10.1109/TIFS.2013.2295032.
[7] Alagoz B, Kaygusuz A, Karabiber A. A user-mode distributed energy management architecture for smart grid applications. Energy 2012;44(1):167–77. https://doi.org/10.1016/j.energy.2012.06.051.
[8] I. A. Kamil, S. O. Ogundoyin, Epdas: Efficient privacy-preserving data analysis scheme for smart grid network, J King Saud Univ - Comput Inf Sci.doi:10.1016/j.jksuci.2018.12.009.
[9] Aitzhan NZ, Svetinovic D. Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams. IEEE Trans Dependable Secure Comput 2018;15(5):840–52. https://doi.org/10.1109/TDSC.2016.2616861.
[10] Kabalci Y. A survey on smart metering and smart grid communication. Renew Sustain Energy Rev 2016;57:302–18. https://doi.org/10.1016/j.rser.2015.12.114.
[11] X. Li, F. Wu, S. Kumari, L. Xu, A. K. Sangaiah, K.-K. R. Choo, A provably secure and anonymous message authentication scheme for smart grids, J Parallel Distrib Comput:10.1016/j.jpdc.2017.11.008.
[12] Abbasinezhad-Mood D, Nikooghadam M. An anonymous ecc-based self-certified key distribution scheme for the smart grid. IEEE Trans Ind Electron 2018;65(10):7996–8004. https://doi.org/10.1109/TIE.2018.2807383.
[13] Kamto J, Qian L, Fuller J, Attia J. Light-weight key distribution and management for advanced metering infrastructure. In: 2011 IEEE GLOBECOM Workshops (GC Wkshps); 2011. p. 1216–20. https://doi.org/10.1109/GLOCOMW.2011.6162375.
[14] Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inf Theory 1976;22(6):644–54. https://doi.org/10.1109/TIT.1976.1055638.
[15] S.A.. Identity-based cryptosystems and signature schemes. Proceedings of CRYPTO 84 on advances in cryptology, vol. 196; 1985. p. 47–53. https://doi.org/10.1007/3-540-39568-7_5.
[16] Wu D, Zhou C. Fault-tolerant and scalable key management for smart grid. IEEE Trans Smart Grid 2011;2(2):375–81. https://doi.org/10.1109/TSG.2011.2120634.
[17] Xia J, Wang Y. Secure key distribution for the smart grid. IEEE Trans Smart Grid 2012;3(3):1437–43. https://doi.org/10.1109/TSG.2012.2199141.
[18] Park JH, Kim M, Kwon D. Security weakness in the smart grid key distribution scheme proposed by xia and wang. IEEE Trans Smart Grid 2013;4(3):1613–4. https://doi.org/10.1109/TSG.2013.2258823.
[19] He D, Kumar N, Lee J-H. Privacy-preserving data aggregation scheme against internal attackers in smart grids. Wireless Network 2016;22(2):491–502. https://doi.org/10.1109/TIE.2018.2807383.
[20] Odelu V, Das AK, M, Conti M. Provably secure authenticated key agreement scheme for smart grid. IEEE Trans Smart Grid 2018;9(3):1900–10. https://doi.org/10.1109/TSG.2016.2602282.
[21] Li Y, Zhang P, Huang R. Lightweight quantum encryption for secure transmission of power data in smart grid. IEEE Access 2019:1. https://doi.org/10.1109/ACCESS.2019.2893056.
[22] P. Gope, B. Sikdar, Privacy-aware authenticated key agreement scheme for secure smart grid communication, IEEE Trans Smart Grid.doi:10.1109/TSG.2018.2844403.
[23] Braeken A, Kumar P, Martin A. Efficient and provably secure key agreement for modern smart metering communications. Energies 2018;11(10):2662. https://doi.org/10.3390/en11102662.
[24] Gope P, Sikdar B. Lightweight and privacy-friendly spatial data aggregation for secure power supply and demand management in smart grids. IEEE Trans Inf Forensics Secur 2019;14(6):1554–66. https://doi.org/10.1109/TIFS.2018.2881730.
[25] Gope P, Sikdar B. An efficient privacy-friendly hop-by-hop data aggregation scheme for smart grids. IEEE Syst J 2019:1. https://doi.org/10.1109/JSYST.2019.2899986.
[26] Gope P, Sikdar B. An efficient data aggregation scheme for privacy-friendly dynamic pricing-based billing and demand-response management in smart grids. IEEE Internet Things J 2018;5(4):3126–35. https://doi.org/10.1109/

JIOT.2018.2833863.
[27] S. Nakamoto, [Bitcoin: A peer-to-peer electronic cash system].
[28] Y. Lu, The blockchain: State-of-the-art and research challenges, J Ind Inf Integr.doi:10.1016/j.jii.2019.04.002.
[29] Lei A, Cruickshank H, Cao Y, Asuquo P, Ogah CPA, Sun Z. Blockchain-based dynamic key management for heterogeneous intelligent transportation systems. IEEE Internet Things J 2017;4(6):1832–43. https://doi.org/10.1109/JIOT.2017.2740569.
[30] Guan Z, Si G, Zhang X, Wu L, Guizani N, Du X, Ma Y. Privacy-preserving and efficient aggregation based on blockchain for power grid communications in smart communities. IEEE Commun Mag 2018;56(7):82–8. https://doi.org/10.1109/MCOM.2018.1700401.
[31] C. H. Liu, Q. Lin, S. Wen, Blockchain-enabled data collection and sharing for industrial iot with deep reinforcement learning, IEEE Trans Ind Inf.doi: 10.1109/TII.2018.2890203.
[32] Swan M. Blockchain: Blueprint for a new Economy. O'Reilly Media, Inc.; 2015.
[33] Fouda MM, Fadlullah ZM, Kato N, Lu R, Shen XS. A lightweight message authentication scheme for smart grid communications. IEEE Trans Smart Grid 2011;2(4):675–85. https://doi.org/10.1109/TSG.2011.2160661.
[34] Li H, Lu R, Zhou L, Yang B, Shen X. An efficient merkle-tree-based authentication scheme for smart grid. IEEE Syst J 2014;8(2):655–63. https://doi.org/10.1109/JSYST.2013.2271537.
[35] Muzammal M, Qu Q, Nasrulin B. Renovating blockchain with distributed databases: An open source system. Future Gener Comput Syst 2019;90:105–17. https://doi.org/10.1016/j.future.2018.07.042.
[36] Yuan Y, Wang F-Y. Blockchain: The state of the art and future trends. Acta Autom Sin 2016;42(4):481–94.
[37] Zheng Z, Xie S, Dai H, Chen X, Wang H. An overview of blockchain technology: Architecture, consensus, and future trends. In: 2017 IEEE International Congress on Big Data (BigData Congress); 2017. p. 557–64. https://doi.org/10.1109/BigDataCongress.2017.85.
[38] Castro M, Liskov B. Practical byzantine fault tolerance and proactive recovery. ACM Trans Comput Syst 2002;20(4):398–461. https://doi.org/10.1145/571637.571640.
[39] Liu J, Li W, Karame GO, Asokan N. Scalable byzantine consensus via hardware-assisted secret sharing. IEEE Trans Comput 2019;68(1):139–51. 0.1109/TC.2018.2860009.
[40] Oh H, Kim J, Shin JS. Forward-secure id based digital signature scheme with forward-secure private key generator. Inf Sci 2018;vols. 454–455:96–109. https://doi.org/10.1016/j.ins.2018.04.049.
[41] CertiVoxiMIRACL [Online] Available: https://github.com/CertiVox/MIRACL.
[42] Wazid M, Das AK, Kumar N, Rodrigues JJ. Secure three-factor user authentication scheme for renewable-energy-based smart grid environment. IEEE Trans Ind Inf 2017;13(6):3144–53. https://doi.org/10.1109/TII.2017.2732999.