7-2019

Pruneable sharding-based blockchain protocol

Xiaoqin FENG

Jianfeng MA

Yinbin MIAO

Qian MENG

Ximeng LIU
Singapore Management University, xmliu@smu.edu.sg

See next page for additional authors

Citation

FENG, Xiaoqin; MA, Jianfeng; MIAO, Yinbin; MENG, Qian; LIU, Ximeng; JIANG, Qi; and LI, Hui. Pruneable sharding-based blockchain protocol. (2019). *Peer-to-Peer Networking and Applications*. 12, (4), 934-950.
Available at: https://ink.library.smu.edu.sg/sis_research/5153

Peer-to-Peer Networking and Applications (2019) 12:934–950
https://doi.org/10.1007/s12083-018-0685-6

Pruneable sharding-based blockchain protocol

Xiaoqin Feng[1] · Jianfeng Ma[1] · Yinbin Miao[1] · Qian Meng[2] · Ximeng Liu[3] · Qi Jiang[1] · Hui Li[1]

Abstract

As a distributed ledger technology, the block-chain has gained much attention from both the industrical and academical fields, but most of the existing blockchain protocols still have the cubical dilatation problem. Although the latest Rollerchain has mitigated this issue by changing the blockheader's contents, the low efficiency, severe capacity expansion and non-scalability problems still hinder the adoption of Rollerchain in practice. To this end, we present the pruneable sharding-based blockchain protocol by utilizing the sharding technique and PBFT(Practical Byzantine Fault Tolerance) algorithm in the improved Rollerchain, which has high efficiency, slow cubical dilatation, small capacity expansion and high scalability. Moreover, the pruneable sharding-based blockchain protocol is certifiably secure and scalable. The experimental results show the protocol has good performance.

Keywords Cubical dilatation · Efficiency · Capacity expansion · Scalability · Sharding technique

1 Introduction

Since Satoshi Nakamoto proposed the concept of bitcoin [1], whose underlying core technology, the block-chain [2], has attached great importance to financial institutions, investment institutions, regulators departments and government departments. The blockchain is a distributed ledger, which is collectively maintained in a decentralized and trustless way. The main characteristics of blockchain are decentralized, trustless, collectively maintained, transparent, user-anonymous, time-sequential, tamper-resistant and traceable. However, th-ere are still many open problems with the existing block-chain protocols, which include the technical challenges in consensus mechanism [3], smart contract [4] and data security [5] as well as the practical performance [6, 7] in terms of efficiency, cubical dilatation and capacity expansion, and the scalability [8].

The practical performance and scalability are extremely crucial. For instance, in the bitcoin blockchain system, there are only 7 transactions handled per second. The participating nodes in the blockchain system must store the complete blockchain, so as more and more transactions are written into the blockchain, the nodes face enormous storage and computational pressures, which is what we call the cubical expansion. Lightweight nodes can partially solve this problem, but industrial scale solutions for larger scales are still to be studied. In addition, for the main blockchain itself, the transaction information in the existing blockchain protocols is pretty redundant, and the block data structure is cumbersome. As a result, each block takes up a lot of storage space, which leads to the capacity expansion problem, and the more transactions written into the blockchain, the more serious the capacity expansion problem. There are two solutions to this problem. The SegWit and other schemes are used to reduce the volume of transaction information, optimize the data structure and/or delete unnecessary information. The side chain and lightning network are used to take the transaction information away from the main blockchain. Furthermore, when the number of nodes in blockchain system dynamically increases, the system has poor adaptability as well as low throughput, which incurs low scalability of the system.

✉ Jianfeng Ma
jfma@mail.xidian.edu.cn

Xiaoqin Feng
fengxiaoqin@stu.xidian.edu.cn

[1] School of Cyber Engineering, Xidian University, Xi'an, China

[2] School of Telecommunication Engineering, Xidian University, Xi'an, China

[3] School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore, Singapore

The blockchain efficiency to a great extent determines the practical feasibility of the blockchain system. In blockchain, the PoW [9] (Proof-of-Work) mechanism is simple to implement, but its trading processing efficiency is pretty low. Aiming at the PoW's inefficiency problem, Sunny King and Scott Nada presented the PoS [10] (Proof-of-Stake) mechanism by using the system stake in place of computing. The PoS mechanism reduces the time to confirm a transaction, but introduces the question of accounter's non-profession. The DPoS [11] (Delegate-Proof-of-Stake) mechanism not only achieves professionalization of the accounter but also enormously lessens the number of nodes verifying and accounting for transactions, which makes the system a second-level consensus. Nevertheless, during the course of consensus, the DPoS mechanism relies on tokens that are infeasible in commercial applications. Although the classical BFT [12, 13] (Byzantine Fault Tolerance) algorithm does not depend on tokens and allows the fast and powerful trading settlement, it is burdened of redundant algorithm complexity. The PBFT [13, 14] (Practical Byzantine Fault Tolerance) algorithm lowers the algorithm complexity from the exponential order to polynomial, but its message complexity is quadratic order of the number of participants. Furthermore, all of these blockchain protocols are confronted with tremendous cubical dilatation, severe capacity expansion and low scalability issues.

There are scarcely any solutions for the cubical dilatation problem in the existing blockchain protocols. The Rollerchain system [15] was proposed to solve the problem, but it still has the slow trading processing efficiency, severe capacity expansion and low scalability problems. With regard to inherent scalability in the bitcoin blockchain, the Bitcoin-NG protocol [16] was designed to scale. However, it is possessed with the same deficiencies when treated as a BFT algorithm.

The PBFT algorithm has high efficiency and the sharding technique [17, 18] is usually used to improve the blockchain scalability. In this paper, we integrate the sharding technique into the PBFT algorithm to implement a new consensus mechanism, which is applied in the improved Rollerchain system to build the pruneable sharding-based blockchain protocol called the PSRB protocol. With the balance between the sharding technique and PBFT algorithm, the message complexity of PSRB protocol is reduced to the same order of the number of participants, and concurrently processing transactions further promotes the trading handling efficiency of PSRB protocol. We also complete the definition of the undefined transaction state change in the Rollerchain system. With the new consensus mechanism in the improved Rollerchain system, the PSRB protocol avoids serious cubical dilatation problem because each node only saves the blockheader chain and some blocks including the creation block. Furthermore, we verify that the PSRB protocol has smaller capacity expansion than the bitcoin blokchain. The main chain in the PSRB protocol contains the blockheader chain and the last few blocks which are used to deal with the blockchain forking. For the scalability of PSRB protocol, the sharding technique ameliorates system's adaptability when the number of participating nodes is increasing dynamically, which endows the system with better scalability. As a further step, we testify the protocol security and prove the performance and scalability of the proposed protocol. We also experiment on the consensus delay and cubical dilatation of the PSRB protocol. Table 1 reflects the comparison among some classical blockchain protocols and the PSRB protocol. Cd stands for the cubical dilatation, and Ce stands for the capacity expansion.

Table 1 Comparison among some classical blockchain protocols and the PSRB protocol

Schemes	Efficiency	Cd	Ce	Scalable
PoW	low	fast	large	low
PoS	low	fast	large	low
BFT	low	fast	large	low
PBFT	high	fast	large	low
Rollerchain	low	slow	large	low
PSRB	high	slow	small	high

The main contributions of this papre are as follows:

1) The PSRB protocol defines the transaction state change, and exemplifies the course to gain the transaction information via transaction state change. The sharding technique here enhances the trading processing efficiency, reduces the message complexity and promotes the scalability of the PSRB protocol.
2) With the sharding technique, the PSRB protocol enables the PoW consensus mechanism to achieve the concurrently administrating of transactions by the specific assigning rules of communities and certain sharding function of the network.
3) Different from the present blockchain protocols, the PSRB protocol can pledge the system security when each node and the main chain only store the blockheader chain and some blocks, which keeps the protocol from the cubical dilatation and capacity expansion problems.

2 Preliminaries

In this part we first explain the sharding technique employed in the PSRB protocol. Then we present the basic knowledge related to the blockchain security, performance and scalability.

Table 2 Notations in the employed sharding technique

Notations	Descriptions	Notations	Descriptions
IP	Address	(pk, sk)	Secret key
x_j	Node	c_f	Community
c	Community size	τ_k	Transaction
N	Total nodes	2^s	Total communities
τ^f	The sharding	K	Total transactions

2.1 The employed sharding technique

The sharding technique is usually implemented to ameliorate the blockchain scalability. In the sharding technique, different nodes are demanded to save discriminated blocks state and manage diverse transactions in the blockchain system. We apply the shading technique into our protocol for the sake of improving the protocol's efficiency and system's scalability. In addition, the proposed protocol reduces the message complexity in PBFT algorithm by reasonably designing the assigning rules of communities and sharding function of the network. The related notations about the employed sharding technique are demonstrated in Table 2.

The implementing steps of the employed sharding technique are introduced as followings.

1) **Identity building.** The system generates IP (Internet Protocol) and (pk, sk) for each node. The tuple (pk_j, sk_j) is the public and private key pair of x_j. IP_j stands for the identity of x_j.
2) **Community assigning.** Each node achieves the community assigning by specific assigning rules. Each node computes the PoW hard problem, and then is assigned to a certain community based on its nonce. In the case of x_j, suppose that its nonce is $(0 \ldots 0 b_1 \ldots b_{q-s} \ldots b_q)_j$, and the value of the last s bits is calculated by the function $Num_1((b_{q-s} \ldots b_q)_j) = f(f = 1, \ldots, 2^s)$. c_f is the community of x_j. Obviously, there are 2^s diverse communities and $N = c \times 2^s$. Each community internally executes the PBFT algorithm. To cut down the message complexity, the first community is selected to record the corresponding relationship between a community and its members, and the last community as a consensus community is responsible for collecting the processed transactions from others, verifying them and building the new block.
3) **Transactions sharding.** The shardings of transactions are obtained by the certain sharding function $Num_2(\tau_k) = \tau^f (k = 1, \ldots, K; f = 1, \ldots, 2^s - 1)$. Figure 1 features the corresponding connections between each sharding of transactions and each community, and different communities $c_f (f = 1, \ldots, 2^s - 1)$ concurrently handle the transactions in discriminated shardings.
4) **Transactions processing.** In the PSRB protocol, $c_f (f = 1, \ldots, 2^s - 1)$ internally executes the highly efficient PBFT algorithm to process τ^f. Then c_{2^s} collects and processes all transactions from $c_f (f = 1, \ldots, 2^s - 1)$ by the PBFT algorithm. All conformed transactions are written into the new block.

2.2 Security, performance and scalability

Next, we discuss the basic knowledge about protocol security, performance and scalability.

1). Security The blockchain security [19] mainly covers five aspects including the privacy protection [20] (legal privacy of individuals or institutions), data security(identity authentication, access control, data transmission, data reading, data writing, and private key), open and undefended network in the public blockchain system, physical security(the operational network and host of the block-chain system), and wind control mechanism(serious detection measures, damage assessments, technical remedies, security measures and tracing for illegal operations).

Mixed coins, ring signature, homomorphic encryption and zero-knowledge proof have being used to achieve the system privacy protection. For identity authentication, access control and private key security, plenty of mature research programs have been explored currently to strengthen them. Moreover, the identity authentication and other protections can be set to avoid troubles introduced by the open and undefended network. Furthermore, the physical security and wind control mechanism involve more sophisticated manipulations. Therefore, we require to take the data writing and reading safety into account to guarantee the security of PSRB protocol.

2). Performance and scalability Some attributes are used to judge for the performance of the blockchain protocols. They include the performance efficiency, resource consumption, reliability and compliance supervision of the consensus mechanism, the security, confidentiality and accidental situations of the smart contract, the rationality of the reward mechanism, furthermore, the performance and scalability of the blockchain, where the blockchain performance contains the efficiency, cubical dilatation and capacity expansion.

Being different from the traditional PoW mechanism, in the PSRB protocol, the application of PoW mechanism in community assigning involves a hard problem which is much more simple and introduces less resources consumption. This paper combines the sharding technique

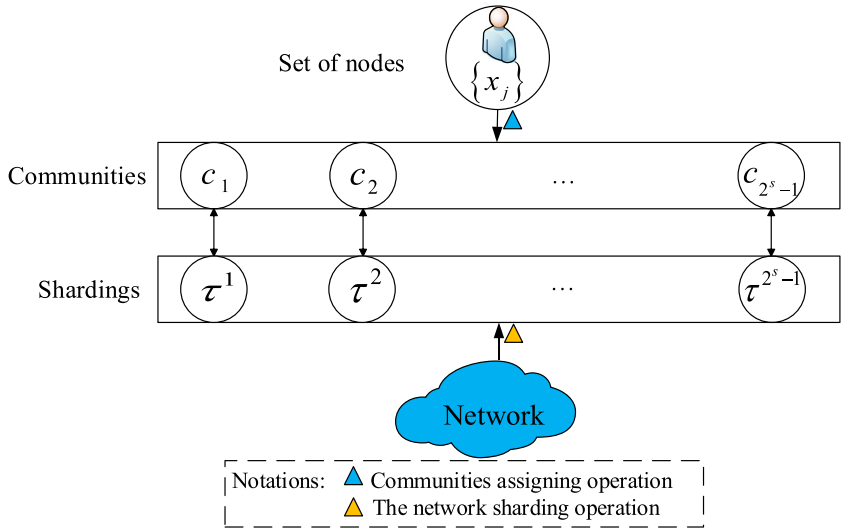

Fig. 1 Corresponding connections between each network sharding and each community

with PBFT algorithm to achieve the protocol's consensus function, and its reliability is the system robustness under various types of attacks, which is a branch of the protocol security. In addition, the compliance regulatory of consensus mechanism and the smart contract are related to deeper researches. Consequently, we are obliged to analyze the performance efficiency of the consensus mechanism(whether it can effectively choose the accounter), the blockchain performance and scalability to guarantee the protocol's integrity.

3 The PSRB protocol

Before dilating the PSRB protocol, we first discuss about the Rollerchain system and its deficiencies.

– **The Rollerchain**

Table 3 explains the Rollerchain related notations and Fig. 2 describes specific substance of the Rollerchain system.

Table 3 Notations in the Rollerchain system

Notations	Descriptions	Notations	Descriptions
S	Trading state	ΔS	Change of S
C	Consensus state	ΔC	Change of C
$*$	State application	box	State box
$\{q_k\}$	Set of box label	$q_k(box_k)$	box label function
(pb, sb)	Secret key of box	Z	Dictionary
s	Set of box	$ticket$	Unspent output
a_t	Proof of $ticket$	ctr	Nonce
a_s	Proof of block	a_τ	Proof of trading

The following procedures state the Rollerchain system with the example of the i-th epoch:

(1) The state change $(\Delta S_i, \Delta C_i)$ is put into the i-th blockheader. Given $(S_i, C_i) = (S_{i-1}, C_{i-1}) * (\Delta S_i, \Delta C_i)$, nodes can obtain the state (S_i, C_i) in the i-th block by applying $(\Delta S_i, \Delta C_i)$ into (S_{i-1}, C_{i-1}).

(2) The tuple $(\Delta S_i, \Delta C_i)_k (k = 1, \ldots, K)$ is separately stored in box_k. In Fig. 3, the system generates a set of $\{q_k\}$ to mark $\{box_k\}$ by the bijection $q_k(box_k)$. The system also generates a set of $\{(pb_k, sb_k)\}$ to encrypt $\{box_k\}$. The function $gen(q_k, (\Delta S_i, \Delta C_i)_k)$ outputs the proof π to reveal the connections between box_k and the k-th $(\Delta S_i, \Delta C_i)_k$. Then the function $mem(\pi)$ builds the relationship between each box_k and $(\Delta S_i, \Delta C_i)_k$. Z records the one-to-one correlations between box_k and $(\Delta S_i, \Delta C_i)_k$.

(3) In Fig. 2, the tuple $(s, a_t, a_\tau, a_s, ctr, Z)$ is put into the blockheader and the trading account is in the blockbody. Each node keeps the blockheader chain and some blocks including the creation block. The main chain contains the whole blockchain.

However, some deficiencies still remain with the Rollerchain system, which are described as follows:

(1) In Fig. 2, there is no concrete description of ΔS.

(2) The Rollerchain has slow trading processing efficiency and low scalability.

(3) The main chain in the Rollerchain system conserves the blockchain, and the tuple $(s, a_t, a_\tau, a_s, ctr, Z)$ is extraly preserved in the blockheader, which worsens the blockchain capacity expansion problem.

We make some improvements to make up for the deficiencies.

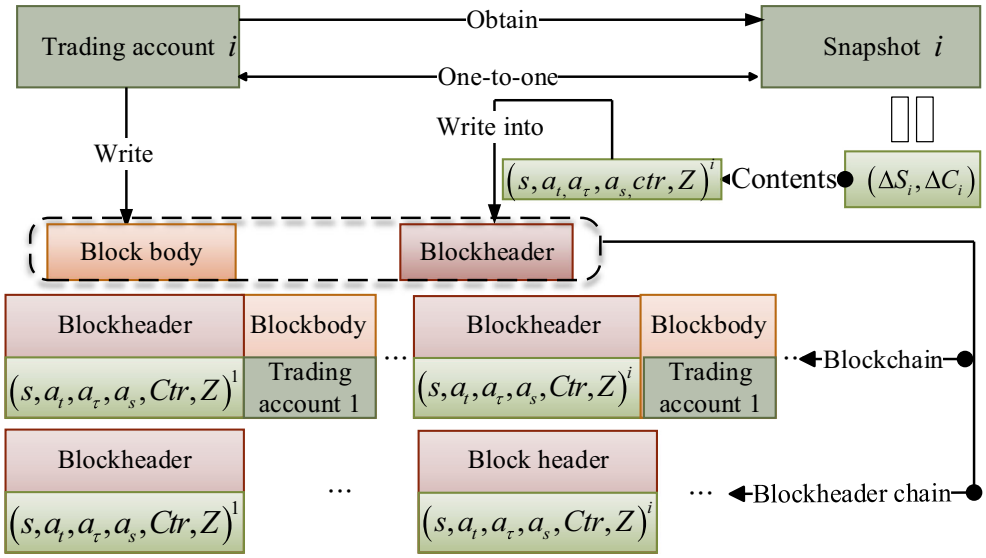

Fig. 2 Substance of the Rollerchain system

(1) Firstly, the concrete definition of ΔS is determined to make an improved Rollerchain system.

(2) With the balance between the sharding technique and PBFT algorithm in the improved Rollerchain system, we gain the PSRB protocol. The Sharding technique endows the PSRB protocol with rather higher efficiency and scalability, and the Rollerchain system with slow cubical dilatation. Moreover, uniting the PBFT algorithm with sharding technique not only maintains the advantages of the PBFT algorithm in terms of high trading processing efficiency but also refrains it from the defect of high message complexity.

(3) We also analyze that the main chain only needs to conserve the blockheader chain and a few last blocks used to handle the blockchain forking matter, which effectively solves the blockchain capacity expansion problem.

Then we start to define notations related to the PSRB protocol and sketch about it.

– **Notations and summary of the PSRB protocol**

Table 4 shows the notations in the PSRB protocol. Figure 4 shows the composition of a blockchain system. The PSRB protocol is devoted to make improvements on

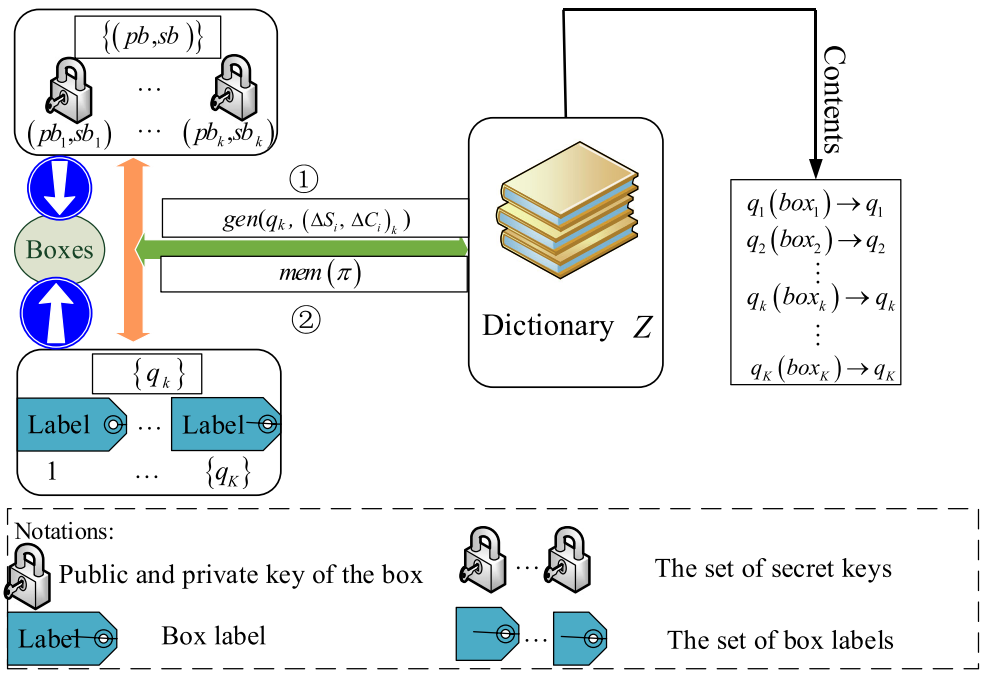

Fig. 3 Mapping between the state change and box, and the function of a dictionary

Table 4 Notations in the PSRB protocol

Notations	Descriptions	Notations	Descriptions
$\tau_1 \dots \tau_K$	Transactions	B_i	Block
hs^i	Root hash of B_i	D	Difficulty value
$\{x^f\}$	Nodes in c_f	M_C	Fund source
M_S	Payment amount	A_S	Address of payer
S_S	Signature of payer	W_M	Fund flows
A_R	Address of payee	S_R	Signature of payee
W_f	Accounter	A_f	Trading account
$whp_.$	Malicious nodes	$\{x^f\}_{hon}$	Honest nodes in c_f
λ	Security parameter	$\|B_{sum}\|$	Total blocks

block data of the data layer and consensus mechanism of the consensus layer. Figure 5 describes the PSRB protocol with three parts. The blockchain system first performs community assigning and network sharding. Then all nodes write the snapshots. Finally, $c_f(f = 1, \dots, 2^s)$ carries out the PBFT algorithm to handle $\tau_1 \dots \tau_K$.

Taking the i-th epoch as an example, we execute the PSRB protocol as the following steps:

(1) **Community assigning and network sharding.** Each node solves the PoW hard problem $SHA256(SHA256(version + hs^{i-1} + hs^i + timestamp + D + nonce)) \leq targetvalue$ to get a nonce. It inputs the acquired nonce and gets a f as an output from the function $Num_1((b_{q-s} \dots b_q)_j) = f(f = 1, \dots, 2^s)$ to assign the node to the community $c_f(f = 1, \dots, 2^s)$. Meanwhile, the network is divided into different sha-rdings by the function $Num_2(\tau_k) = \tau^f(f = 1, \dots, 2^s - 1)$. Each sharding contains some discriminated τ_k which are handled by different $c_f(f = 1, \dots, 2^s - 1)$.

(2) **Writing the snapshot.** To be convenient, we call the account of transaction state change as a snapshot. Contrary to the Rollerchain system, we define the transaction state change. $\{x^f\}(f = 1, \dots, 2^s - 1)$ write the trading accounts and snapshots once they take over τ^f. Moreover, $box_k(k = 1, \dots, K)$ separately preserves a certain snapshot, and relations between the τ_k and the box_k are recorded into Z.

(3) **Transactions processing and block establishing.** Firstly, $\{c_f\}(f = 1, \dots, 2^s - 1)$ respectively implement the PBFT algorithm to deal with τ^f. The conformed transactions then be transmitted to c_2^s. Secondly, c_2^s executes the PBFT algorithm to process the transactions and build the new block.

3.1 Community assigning and network sharding

In this part, each node solves the PoW hard problem to get itself assigned into a certain community. D [21] has to be adjusted to guarantee that the last s bits of *nonce* are not included in the leading zero bits, in this way, the s bits can be assured random. $(b_{q-s} \dots b_q)_j$ is the last s bits in the nonce of x_j. The function $Num_1((b_{q-s} \dots b_q)_j) = f(f = 1, \dots, 2^s)$ maps each $x_j(j = 1, \dots, N)$ to a certain $c_f(f = 1, \dots, 2^s)$, and it satisfies the following conditions:

(1) $\{x^1\}$ firstly works out the nonce and $|x^1| = c$, then for $\forall x_j \in \{x^1\}$, $Num_1((b_{q-s} \dots b_q)_j) = 1$.

(2) For the left nodes, there are two cases. In the first case, $\{x^2\}$ are the first c nodes satisfying $(b_{q-s} \dots b_q)_j = 1$ or $(b_{q-s} \dots b_q)_j = 2(x_j \in \{x^2\})$, then for $\forall x_j \in \{x^2\}$, $Num_1((b_{q-s} \dots b_q)_j) = 2$. For the left $\{x^{2'}\}$ where $(b_{q-s} \dots b_q)_j = 2(x_j \in \{x^{2'}\})$, there is $Num_1((b_{q-s} \dots b_q)_j) = 3$ for $\forall x_j \in \{x^{2'}\}$. In the second case, there are $(b_{q-s} \dots b_q)_j = 1$ or $(b_{q-s} \dots b_q)_j = 2(x_j \in \{x^2\})$ and $|x^2| < c$. Besides, $\forall x_j \in \{x^{2'}\}$ satisfies $(b_{q-s} \dots b_q)_j = 3$, and $|x^2 + x^{2'}| \geq c$. Then for $\forall x_j \in \{x^2\}$ and the first $c - |x^2|$ nodes $x_j \in \{x^{2'}\}$, $Num_1(b_{q-s} \dots b_q)_j = 2$.

(3) For the rest nodes, and so forth.

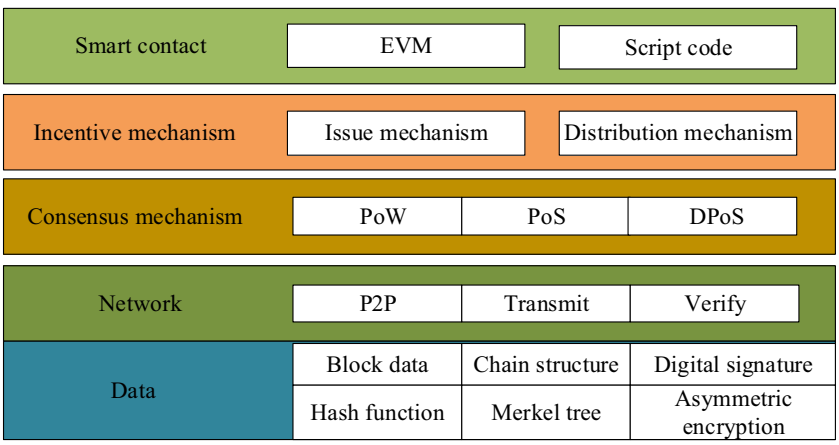

Fig. 4 Components of a complete blockchain system

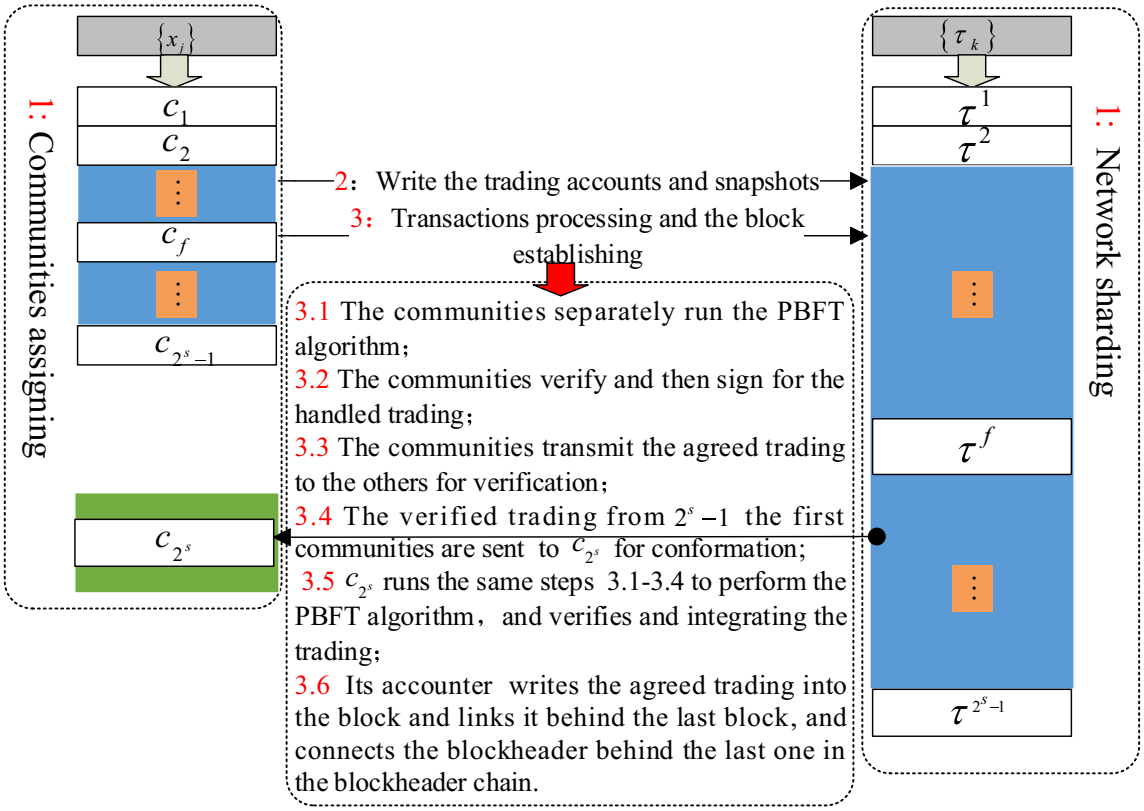

Fig. 5 Operational schemes of the PSRB protocol

Adhering to the above rule $Num_1((b_{q-s}\dots b_q)_j) = f(f = 1,\dots,2^s)$, the community assigning is done. During the process, the nodes join c_1 if the first community has not been built, or else they join $\{c_f\}(f = 2,\dots,2^s)$, and deliver their community labels to c_1. The lastly established consensus community is responsible for collecting, validating and handling the treated trading from the other communities. The function $Num_1((b_{q-s}\dots b_q)_j) = f$ can guarantee that the members of $\{c_f\}(f = 1,\dots,2^s-1)$ are averagely distributed. Besides, by rationally setting the size of c, c_{2^s} satisfies $c \le| x^{2^s} |\le c+\delta, \delta \to 0$. In addition, all nodes are allowed to ask c_1 about the identities of their members.

Analogous to the community assigning, the network shardings can be actualized by the function $Num_2(\tau_k) = \tau^f(f = 1,\dots,2^s-1)$, which builds one-to-one connections between $c_f(f = 1,\dots,2^s-1)$ and $\tau^f(f = 1,\dots,2^s-1)$. For $\forall\tau_k \in \tau_1\dots\tau_K$, $Num_2(\tau_k) = \tau^f(f = 1,\dots,2^s-1)$ satisfies the following properties:

(1) The domain is $\{\tau_1,\dots,\tau_K\}$ and $\{\tau^1,\dots,\tau^{2^s-1}\}$ is range.
(2) $Num_2(\tau_k) = \tau^f$ is a bijection from multi transactions to one sharding.
(3) Inputting $\forall\tau_k$, the function $Num_2(\tau_k) = \tau^f$ randomly outputs a value τ^f that has nothing to do with the τ_k. In this way, the $\tau_1,\dots,\tau_K$ can be randomly distributed.
(4) $|\tau^1| = |\tau^2| = \cdots = |\tau^f| = |\tau^{2^s-1}|$, that is, each sharding has the same number of transactions.

Different communities administer the discriminated shardings and concurrently handle the transactions among them. Figure 5 explains that all nodes write the trading accounts and snapshots based on the community assigning and network sharding.

3.2 Writing the snapshot

The trading account can be incarnated to the tuple $(M_C, M_S, A_S, S_S, W_M, A_R, S_R)$. The PSRB protocol demands the blockbody to keep the trading account. Figure 6 narrates the contacts among different trading accounts and shows their specific contents. Optimizing the trading account we acquire the snapshot, which occupies much less storage space. The snapshot can be equivalent to the tuple (M_C, W_M). Figure 7 presents the contacts among different snapshots and shows their specific contents. Table 5 explains the terms in a snapshot.

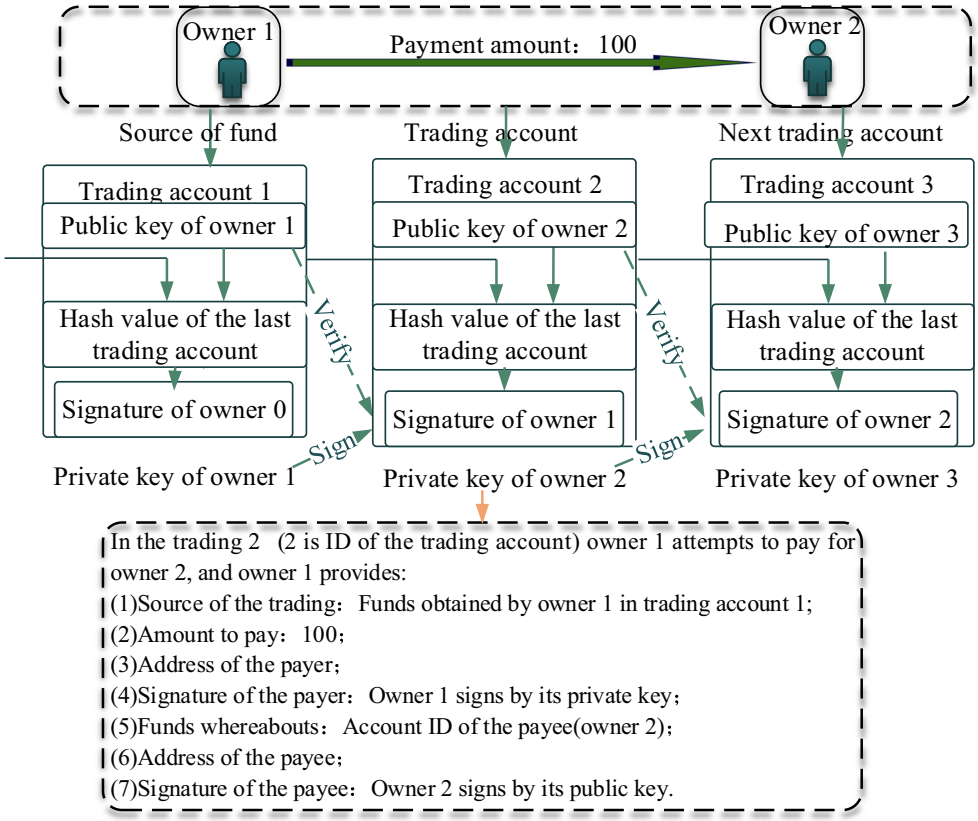

Fig. 6 Contacts among different trading accounts and their specific contents

The ticket is logged into a snapshot, which is saved in a box_k based on Fig. 3. The blockheader contains the tuple $(hs^{i-1}, hs^i, Z, s, others)$, and $others$ refers to the $versionnumber$, $timestamp$, D, and W_{2^s}. With memorizing the blockheader chain and a few of blocks containing the creation block, the nodes can participate in the PSRB protocol. In addition, new nodes join the system with downloading the blockheader chain from the main chain and a few blocks from other nodes. Exerting s in B_i into that in B_{i-1}, we obtain the trading information in the i-th epoch. With the same manner, sequentially applying them into the trading account of the creation block, all trading information up to date can be gained. In Fig. 8 we depict an instance demonstrating the process to obtain the trading information by snapshots, and we take the transaction 2 as an example.

Taking advantage of snapshots 1 and 2, information about the transaction 2 is obtained by the above steps.

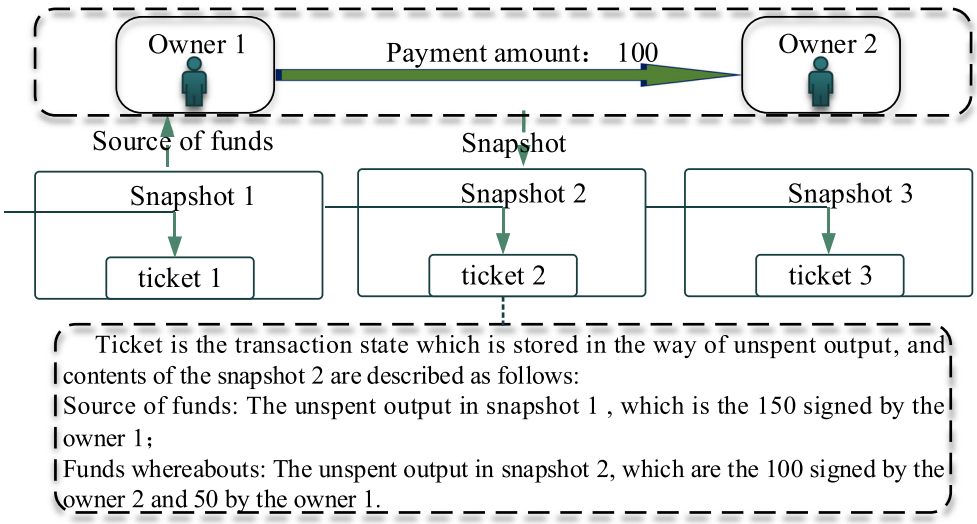

Fig. 7 Contacts among different snapshots and their specific contents

Table 5 Contents of a snapshot

Items	Contents
Funds source	Trading account ID of unspent output
Funds flows	Trading account ID of unspent output

(1) Deriving from the funds source of snapshot 2, we conclude that in the transaction 2, the funds come from the 150 signed by B in the snapshot 1;
(2) Deriving from the funds flows of snapshot 2, we conclude that funds of C increase by 100, and funds of B increase by 100 in transaction 2.

Therefore, in the transaction 2, B pays 100 for C and 50 for himself, and its last transaction is the transaction 1.

3.3 Transactions processing and block establishing

The section shows the process to handle the transactions. Like in Fig. 5, the trading processing includes two main steps. In steps 3.1-3.4, $c_f(f = 1, \ldots, 2^s - 1)$ executes the PBFT algorithm to handle the transactions. Furthermore, c_{2^s} performs the PBFT algorithm to dispose the processed trading from $c_f(f = 1, \ldots, 2^s - 1)$, and establishes the new block in steps 3.5-3.6. For minutely describing the trading handling process, we amply list the PSRB protocol in steps instead of the steps 3.1-3.6.

At the first stage, $c_f(f = 1, \ldots 2^s - 1)$ internally executes the following steps to deal with the transactions:

(1) $\{x^f\}$ verifies the funds source, funds flows and the payment amount of transactions in τ^f.
(2) If τ^f are impactful, c_f severally executes the PBFT algorithm to choose W^f.
(3) W^f accounts for τ^f, and writes the A^f and the snapshot.
(4) A^f and the snapshot are sent to other nodes in c_f for verification, and they are valid once signed by $\mid x^f \mid /2 + 1$ nodes.
(5) $\{x^f\}$ transmits the signed A^f and the snapshot to c_{2^s}.

Next, c_{2^s} begins to collect, validate and account for all transactions, and builds the new block. All unreasonable nodes are demonstrated as whp..

(6) By querying to c_1, $\{x^{2^s}\}$ checks the correctness of nonce and the matching between communities and their members. Then $\{x^{2^s}\}$ performs the same PBFT algorithm as $c_f(f = 1, \ldots, 2^s - 1)$ to validate $W^f(f = 1, \ldots, 2^s - 1)$, $A_f(f = 1, \ldots, 2_s - 1)$ and the snapshot.
(7) $\{x^{2^s}\}$ runs the PBFT algorithm to selec W^{2^s} which deals with transactions with the correct $A^f(f = 1, \ldots, 2^s - 1)$ and snapshots;
(8) W^{2^s} logs the effective A^{2^s} into the blockbody and the snapshot tuple $(hs^{i-1}, hs^i, Z, s, others)$ into the blockheader to establish B_i.
(9) $\{x^{2^s}\}$ collates B_i and signs for it. The B_i with $\mid x^f \mid /2 + 1$ signatures is proper. Then $\{x^{2^s}\}$ transmit the signed B_i to the network to be examined. Finally, the B_i with $N/2 + 1$ signatures is linked behind the main chain. Moreover, each node adds the blockheader of B_i behind its blockgheader chain. The main chain includes the blockheader chain and the last few blocks.

In the PSRB protocol, the employment of sharding technique in the PBFT algorithm on one hand raises the transactions handling efficiency and disposes off the high message complexity matter in the PBFT algorithm, on the other hand enhances the blockchain scalability. Still, the application of PoW in community assigning consumes rather less resource. The snapshot in the blockheader is in favor of alleviating pressures of nodes in memorizing the blockchain. Moreover, there is no longer necessary for the main chain to keep the blockchain, which resolves the

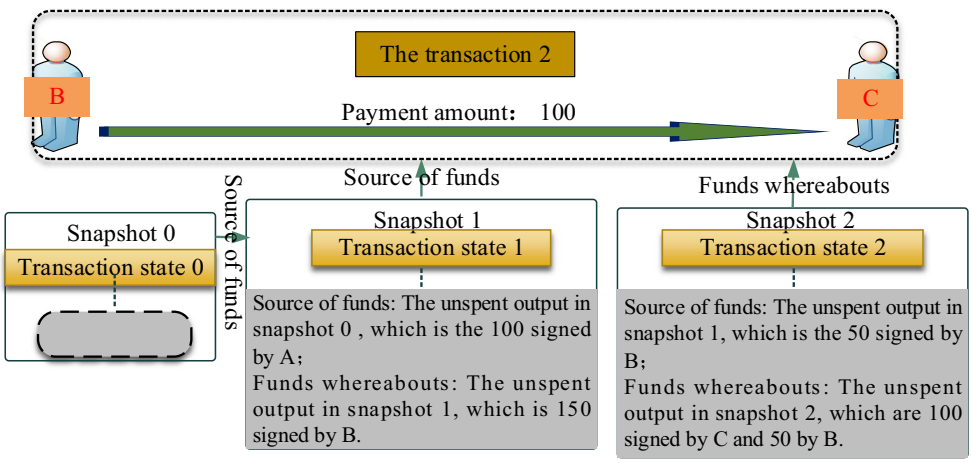

Fig. 8 The process to obtain the trading information by snapshots

capacity expansion problem. Most importantly, the double verifications on the trading account and snapshot make the PSRB protocol a more accurate consensus.

Next we analyze and prove the protocol security. The protocol performance and scalability will be proved in the next section either. Considering the protocol security theory in Section 2.2, security of data writing and reading are left to be proved. The writing security covers the transactions transmission security, consensus safety (safety of the participating nodes and consensus mechanism), and storage security. The reading security is mainly about the downloading security. We present these security proofs in Section 4. In addition, some attributes related to the PRSB protocol remain to be certified, which are performance efficiency of the consensus mechanism, the blockchain performance and scalability. We first sketch the performance efficiency of the PSRB protocol. Then its performance and scalability will be especially proved. The blockchain performance are comprised of efficiency, cubical dilatation and capacity expansion. Thereinto, efficiency indicates the transactions processing rate and message complexity. Cubical dilatation is the storage pressure of each single node, and capacity expansion is about the volume change of the main chain. For the blockchain scalability, it is relative to the adaptability and throughput of the system when there are increasing nodes. We present the relevant proofs about the four attributes in Section 5.

4 Security

The classical blockchain can guarantee the system security when all nodes store and new nodes download the blockchain. Considering the improvements in the PSRB protocol, the system security involves safety of participating nodes(the honest nodes in each community are the most), storage safety(the system security can be guaranteed as long as each node keeps the blockheader chain and a few blocks), downloading safety(the blockheader chain and partial blocks downloaded by new nodes are integrated). Due to the wide application of the PBFT algorithm, we will no longer prove the security of PBFT algorithm.

Proposition 1 The participating nodes are safe.

(1) The honest nodes in the first community are the most. *In each epoch, for a sufficiently large integer* $|\ x^1\ | > n_1$, *there are at most* $|\ x^1\ |/3 - 1\ whp.$ *in* c_1.

Proof The event "$x_j (j = 1, \ldots, N)$ is the u-th node that has calculated nonce" obeys the binomial distribution (N, p_j), and p_j is the computating proportion of x_j. Referring to the hypothesis in the Rollerchain system, $Ad = 1/3$ is the highest computating proportion of $whp.$. Force $Pr(|\ \{x^1\}_{hon}\ | \leq 2\ |\ x^1\ |/3) = P_0$, and it satisfies:

$$\begin{aligned} P_0 &= \Sigma_{u=1}^{\lceil 2|x^1|/3\rceil} Pr(x = u) \\ &= \Sigma_{u=1}^{\lceil 2|x^1|/3\rceil} \binom{|x^1|}{u} (1 - p_j)^u p_j{}^{|x^1|-u}. \end{aligned} \tag{1}$$

The probability in Eq. 1 exponentially decreases with the value of $|\ x^1\ |$. For a given λ, we can find an integer n_1 such that $\forall\ |\ x^1\ | > n_1$, $Pr(|\ \{x^1\}_{hon}\ | \leq 2d/3) \leq 2^{-\lambda}$. A community size is set to be at least n_1, which is to assure that the number of $whp.$ in a community is up to $|x_1|/3$ with the parameter λ.

(2) The honest nodes in $c_f (f = 2, \ldots, 2^s)$ are the most. In each epoch, for a sufficiently large integer $|\ x^f\ | > n_f$, there are at most $|\ x^f\ |/3 - 1\ whp.$ in the community c_f.

Proof The method of proof is same as above. □

Proposition 2 The storage is safe.

It is safe for all nodes to save the blockheader chain and some blocks. *We suppose that each node has to keep* $|B'|$ *different blocks. For given* $|B_{sum}|$ *and Ad,* $\exists |B_0| \in N^+$, *when* $|B'| \geq |B_0|$, $whp.$ *combine with each other to tamper blocks, which will not affect the blockchain integrity.*

Proof In the PSRB protocol each node stores the block-header chain. Analogous to the classical block-chain, each node stores the blockchain. Therefore, the system is secure even though $whp.$ tamper the blockheader. To keep the system security, the safety of blockbody must be guaranteed.

Considering the case where all malicious nodes jointly tamper the common block they store.

The number of blocks saved by $whp.$ is $2^s \cdot \varepsilon \cdot |B'| (0 < \varepsilon < c/3 - 1)$, and that of the honest nodes are $[N - 2^s \cdot \varepsilon] \cdot |B'|$.

Suppose that B_w is the common block of $whp.$ and they are planning to tamper it. Then the number of B_w stored by $whp.$ satisfies that $|(B_w)^{whp}| = 2^s \cdot \varepsilon$. To keep the blockchain integrity, B_w saved by the honest nodes must fulfil the following circumstance:

$$|(B_w)^{hon}| > 2^s \cdot \varepsilon. \tag{2}$$

From Eq. 2, we can gain that $min\{|(B_w)^{hon}|\} = 2^s \cdot \varepsilon + 1$. Then the total number of blocks saved by the honest nodes satisfies $\Sigma_{hon} = (2^s \cdot \varepsilon + 1) \cdot |B_{sum}|$. Considering that the honest nodes evenly save the blocks, that is, each of them saves $|(B_j)^{hon}|$ blocks, and $|(B_j)^{hon}|$ satisfies the Eq. 3.

$$|(B_j)^{hon}| = (2^s \cdot \varepsilon + 1) \cdot |B_{sum}|/(N - 2^s \cdot \varepsilon). \tag{3}$$

Let $|B_0| = (2^s \cdot \varepsilon + 1) \cdot |B_{sum}|/(N - 2^s \cdot \varepsilon)$, and the Proposition 2 gets proved. □

Proposition 3 The downloading process of new nodes is safe.

The blocks downloaded by new nodes are integrated. *Considering the condition where a new node x_{new} joins the blockchain network, we assume that there are multitudinous nodes in the network and x_{new} sees the last few blocks. Once x_{new} joins the network, it downloads $|B'|$ blocks from others and the blockheader chain from the main chain, and the downloading process is safe.*

Proof Each node stores the blockheader chain in the PSRB protocol. Analogous to the classical blockchain, it is secure for x_{new} to download the blockheader chain. Consequently, to guarantee the downloading safety, the $|B'|$ blocks downloaded by x_{new} must be integrated.

Taking the Proposition 3 as reference, we conclude:

(1) For $\forall B_i$, B_i that is kept by the honest nodes fulfils $|\ (B_i)^{hon}\ | \geq |\ (B_i)^{whp}\ | +1$;

(2) Assume that x_{new} downloads the B_i from $whp_.$, then $|(B_i)^{whp}|$ increases 1. For $N \gg 1$, there is

$$\begin{aligned} \Sigma_{hon}^{B^i} &= |(B_i)^{hon}| \cdot (N - 2^s \cdot \varepsilon) \\ &\geq (|\ (B_i)^{whp}\ | +1) \cdot (N - 2^s \cdot \varepsilon) \\ &\geq (|\ (B_i)^{whp}\ | +1) \cdot 2^s \cdot \varepsilon. \end{aligned} \tag{4}$$

$$\Sigma_{whp}^{B^i} = |\ (B_i)^{whp}\ | \cdot 2^s \cdot \varepsilon + 1. \tag{5}$$

Then, there is the following relation:

$$\begin{aligned} \Sigma_{hon}^{B^i} &\geq |\ (B_i)^{whp}\ | \cdot 2^s \cdot \varepsilon + 2^s \cdot \varepsilon \\ &\geq |\ (B_i)^{whp}\ | \cdot 2^s \cdot \varepsilon + 1 = \Sigma_{whp}^{B^i}. \end{aligned} \tag{6}$$

(3) If x_{new} downloads the B_i from the honest nodes, then $|(B_i)^{hon}|$ increases 1. For $N \gg 1$, there is

$$\begin{aligned} \Sigma_{hon}^{B^i} &= |(B_i)^{hon}| \cdot (N - 2^s \cdot \varepsilon) + 1 \\ &\geq (|\ (B_i)^{whp}\ | +1) \cdot (N - 2^s \cdot \varepsilon) + 1 \\ &= |\ (B_i)^{whp}\ | \cdot (N - 2^s \cdot \varepsilon) + (N - 2^s \cdot \varepsilon) + 1 \\ &\geq |\ (B_i)^{whp}\ | \cdot \frac{2}{3} \cdot 2^s \cdot \varepsilon + \frac{2}{3} \cdot 2^s \cdot \varepsilon + 1. \end{aligned} \tag{7}$$

$$\Sigma_{whp}^{B^i} = |\ (B_i)^{whp}\ | \cdot 2^s \cdot \varepsilon. \tag{8}$$

Comparing formulas (4), (5), (7) and (8), we know that $\Sigma_{hon}^{B^i} \geq \Sigma_{whp}^{B^i}$.

Taking the three aspects into account, we conclude that the downloading process of x_{new} is sufficiently secure.

As a further step, according to formulas (3) and (4), we can have

$$|B_0{}'| = (2^s \cdot \varepsilon + 1) \cdot (|B_{sum}| + 1)/[(N + 1) - 2^s \cdot \varepsilon]. \tag{9}$$

at the end of epoch. That is, $|B_0{}'| < |B_0|$.

To sum up, from formulas (4)–(8), we understand that the joining of x_{new} strengthens the blockchain integrity, and from the formula (9), x_{new} can share the storage pressure for the network node.

Propositions 1-3 have proved the protocol security. Next we illustrate the performance efficiency of the consensus, and then give emphasized proofs for the protocol performance and scalability. □

5 Performance and scalability

Before proving the protocol-related attributes, we first illustrate some advantages of the PSRB protocol compared with the PBFT algorithm.

- In the PSRB protocol, the PBFT algorithm is executed in a smaller community and more centralized network, which decreases the number of communicated messages and the network latency time during the consensus process. Therefore, the protocol efficiency and the consensus accuracy get promoted;
- The number of participants among a community is much less, which improves their successful probability to account. Assume that each node has equal capacity at beginning, then its successful probability to account is $1/N$. In the PSRB protocol, the size of a community is $|\ c_f\ |$. Suppose that $|\ c_1\ | = |\ c_2\ | = \cdots = |\ c_{2^s}\ | = c$, then successful probability of a node is $2^s/N$, and $2^s/N > 1/N$.
- In the PSRB protocol, the PBFT algorithm occurs in a smaller network and much fewer nodes participate in the PBFT algorithm, which promotes the consensus speed.

The performance efficiency is an attribute of consensus mechanism, and it indicates the probability to select the accounter and the robustness of consensus mechanism. The first aspect involves with Section 3.3 about determining the accounter. During the process, each community executes the PBFT algorithm to choose the accounter. Because the PBFT algorithm is widely used to achieve consensus, then the first aspect can be reached and the accounter can be selected. For the second aspect, from the advantages of PSRB protocol consensus, we conclude that the PSRB protocol is much more robust and greater than the PBFT algorithm.

We prove the protocol performance(efficiency, cubical dilatation and capacity expansion) and scalability next. The efficiency includes the transactions processing speed and message complexity of PSRB protocol. Cubical dilatation is about storage space needed by each node to save the blockchain. Capacity expansion indicates the change of main chain's storage space. Furthermore, scalability of the PSRB protocol can be concluded from the time to get the

trading conformed, the number of transactions and blocks conformed per unit time.

Although the PBFT algorithm is high in trading processing efficiency, its message complexity is quadratic level of the number of participants. Then we require to prove that the PSRB protocol is even more highly efficient and lower in message complexity than the PBFT algorithm.

Proposition 4 The high efficiency. *The PSRB protocol is rather more efficient in processing transactions than the PBFT algorithm and its message complexity is* $O(N)$.

Proof In the PSRB protocol, $c_f(f = 1, \ldots, 2^s - 1)$ concurrently runs the PBFT algorithm, which promotes the trading processing efficiency of traditional PBFT algorithm. Moreover, x_{new} does not require to download and manage all blockchain trading dating to the creation block, and they instead download the blockheader chain and some blocks. Therefore, a great deal of workload is abbreviated, which is conducive to improve the blockchain efficiency.

Then we explicitly attest that the message complexity of the PSRB protocol is $O(N)$.

The message transmission in the PSRB protocol is involved in two stages: the community assigning in Section 3.1 and transactions processing in Section 3.3. We discuss the message complexity during the two processes respectively.

(1) ***Community assigning and network sharding.***
$x_j(j = 1, \ldots, N)$ inquiries about its community members after finishing the community assigning. We assume that $| c_1 |=| c_2 |= \cdots =| c_{2^s} |= c$. If $x_j(j = 1, \ldots, N)$ communicates in a point-to-point manner among a community, it will convey $O(c^2)$ to $O(c^3)$ messages, which respectively correspond to the best case(the first node that is asked is the very its member) and the worst(the last is its member). Hence, the message complexity during the process is $O(Nc^3)$. In the PSRB protocol, however, the first community is selected as a directory to reduce the messages propagation. Each node asks c_1 about its community members, which conveys $O(c)$ messages. Then the message complexity is $O(Nc)$.

(2) ***Transactions processing and the block establishing.***
During $c_f(f = 1, \ldots, 2^s)$ implements the PBFT algorithm, there are at most c turns of c^2 multi-pointed transmissions. Besides, $\{x^f\}(f = 1, \ldots, 2^s - 1)$ send the transactions to c_{2^s}, and $\{x^{2^s}\}$ deliver the conformed blocks to all nodes, which severally convey $O(Nc)$ messages.

In conclusion, the message complexity of the PSRB protocol is $O(3Nc + c^2)$, which roughly is $O(N)$.

In the existing blockchain protocols, each node conserves the blockchain, which causes serious cubical dilatation. The PSRB protocol has solved this problem by making each node save the blockheader chain and a few of blocks. Next we certify that the PSRB protocol has slower cubical dilatation than the classical blockchain, and we can take the bitcoin blockchain as an example. □

Proposition 5 The slow cubical dilatation. *The PSRB protocol holds slower cubical dilatation than the bitcoin blockchain.*

Proof In classical blockchain, each node is demanded to preserve the blockchain. There are increasingly burdensome blocks data saved by each node with more and more transactions. Instead of preserving the blockchain, the PSRB protocol requires each node to save the blockheader chain and some blocks. We specify all storage items saved by each node in the condition of the bitcoin blockchain and PSRB protocol.

(1) Items in the bitcoin blockheader $(B_{head})^b$ include the version number Ver^b, hash value $(hs^{-1})^b$ of the last block, timestamp g^b, parameters in PoW (D^b, $nonce$), and merkel root hs^b. Items in the bitcoin blockbody $(B_{body})^b$ include the K and the trading account $(M_C, M_S, A_S, S_S, W_M, A_R, S_R)$.

(2) Items in the PSRB protocol blockheader $(B_{head})^{sr}$ include the version number Ver^{sr}, hash value $(hs^{-1})^{sr}$ of the last block, timestamp g^{sr}, parameters in PoW (D^{sr}), merkel root hs^{sr}, Z and snapshot (M_C, W_M). Items in the blockbody $(B_{body})^{sr}$ include the K and the trading account $(M_C, M_S, A_S, S_S, W_M, A_R, S_R)$.

In the bitcoin blockchain, each node saves the blockchain which includes all $(B_{head})^b$ and $(B_{body})^b$. The PSRB protocol requires each node to save the blockheader chain including all $(B_{head})^{sr}$ and $|B'|$ different blocks. Because the blockchain becomes more and more longer, and there are increasing nodes, the $|B'|$ blocks can be ignored when compared with the blockheader chain and the blockchain. Table 6 shows the items saved by each node in the bitcoin blockchain and in the PSRB protocol.

In the PSRB protocol, Z contains the items $q_k(box_k) \rightarrow q_k(k = 1, \ldots, K)$, and $nonce$ includes all the proper answers of the traditional PoW hard problem. Therefor, $V_6 \geq V_7$. In addition, in the bitcoin blockchain, M_S, A_S, S_S, A_R and S_R are also saved by each node. We suppose that their space size are $V_{11}, V_{12}, V_{13}, V_{14}$ and V_{15}. M_S, A_S, S_S, A_R and S_R occupy most of $(M_C, M_S, A_S, S_S, W_M, A_R, S_R)$, therefore, $V_6 + V_{11} + V_{12} + V_{13} + V_{14} + V_{15} \gg V_7$.

Table 6 Comparison between the bitcoin blockchain storage items and those of the PSRB protocol

Items in bitcoin	Space	Items in the PSRB prococol	Space
Ver^b	V_1	Ver^{sr}	V_1
$(hs^{-1})^b$	V_2	$(hs^{-1})^{sr}$	V_2
g^b	V_3	g^{sr}	V_3
D^b	V_4	D^{sr}	V_4
hs^b	V_5	hs^{sr}	V_5
$nonce$	V_6	Z	V_7
M_C	V_8	M_C	V_8
W_M	V_9	W_M	V_9
K	V_{10}	K	V_{10}

In conclusion, the nodes cubical dilatation of the PSRB protocol is much slower than that of the bitcoin blockchain. As the number of transactions increases, the growing rate of occupied space of each node in the PSRB protocol becomes more slower than that in the bitcoin blockchain.

Similar to the cubical dilatation, the main chain stores the blockchain in the existing blockchain protocols, which causes the main chin severe capacity expansion. The PSRB protocol makes improvement on this issue. We then prove that the PSRB protocol has smaller capacity expansion, and we take the bitcoin blockchain as an example. □

Proposition 6 The small capacity expansion. *The PSRB protocol holds smaller capacity expansion than the bitcoin blockchain.*

Proof The blockchain space linearly increases with the number of transactions. The PSRB protocol optimizes the structure of trading account in classical block-chain, and obtains the snapshot which is stored in the blockheader. In the PSRB protocol the main chain conserves the blockheader chain and some last blocks. The number of blocks used to handle the blockchain forking remains unchanged, and for a long term, they can be neglected when compared with the blockheader chain and the blockchain. The same as the Proposition 5, the blockheader chain occupies much smaller space than the blockchain. Hence the main chain of the PSRB protocol has smaller capacity than the bitcoin blockchain.

From Propositions 4-6, we certify that the PSRB protocol possesses a better performance. The following part proves the scalability of PSRB protocol. The exiting blockchain protocols have low scalability, and the PSRB protocol is devoted to solve the issue. Taking the bitcoin blockchain as example, we prove that the PSRB protocol is higher scalable than the bitcoin blockchain. We divide the proof into three parts including the time to reach the transactions' consensus, the number of processed transactions and formed blocks per unit time. □

Proposition 7 The high scalability. *With the number of network nodes linearly increasing, in the PSRB protocol, the time to reach the transactions consensus decreases linearly, the number of transactions conformed and blocks built per unit time grows linearly.*

Proof Assume that the number of network nodes increases with the function $x = ag + b$, a and b are constants, and $a > 0$. The time spent to reach the transactions consensus contains the time in community assigning and transactions processing. To solve the PoW hard problem occupies much time of the community assigning. We consider the spent time from four aspects:

(1) The time used to compute the PoW hard problem is independent of the number of nodes, and it depends on the value of D;
(2) $\{x_j\}(j = 1, \ldots, N)$ concurrently compute the PoW hard problem, therefore, the total time spent is decided by the slowest node. Assume the time is $max\ \{g_{sum}\}$ and it grows linearly with the time g;
(3) By the function $Num_1(b_{q-s} \ldots b_q) = f(f = 1, \ldots, 2^s)$, once the first nonce is calculated the community assigning begins. With the time g goes on, the number of nodes that have finished the community assigning increases linearly;
(4) The nodes start to process transactions once they achieve the community assigning. In Section 3.3, the transactions with $Num(sign) = c/2+1$ signatures get conformed.

Assume that the i-th epoch begins at the time g_b, and ends at g_e. At g_b the number of network nodes is N^i. In Section 3.3, the process for $c_f(f = 1, \ldots, 2^s - 1)$ to execute the PBFT algorithm is same as that of c_{2^s}. We then consider them as one process of executing the PBFT algorithm. By the function $x = ag + b(g \in (g_b, g_e))$, at g_e there are $(g_e - g_b)a + b$ newly added nodes.

At $g_b + max\{g_{sum}\}$, the number of nodes are $N^i + a(max\{g_{sum}\}) + b$. Consider the physical condition where $N^i \gg a(max\{g_{sum}\}) + b$, therefore, $N^i/2 + 1 \gg a(max\{g_{sum}\}) + b$. In Section 3.3, the transactions with $Num(sign) = c/2 + 1$ signatures can be conformed. Then at this time, N^i nodes have been starting running the PBFT algorithm and the transactions can be conformed.

When the N^i nodes run the PBFT algorithm, new nodes constantly join the network to solve the PoW hard problem, and then run the PBFT algorithm either. Therefore, the N_i nodes executing the PBFT algorithm will not be affected by the new nodes. Besides, the new nodes runing the PBFT

algorithm linearly shorten the total time to finish the PBFT algorithm. In conclusion, time used for the transactions consensus decreases linearly when new nodes continuously join the network.

Because the time used to get the transactions consensus decreases linearly, the number of transactions conformed per unit time increases linearly. In addition, the processing speed of transactions increases, thus, the number of blocks established per unit time grows linearly. □

6 Experimental evaluation

Based on the bitcoin blockchain, we design our procedure with JAVA, and we implement all components of the PSRB protocol on the procedure and the PBFT algorithm on the popular hyperledger fabric [22] with the computer attribute of Intel(R)core(TM) i5-4590 CPU @ 3.30GHZ and RAM:8GB. The number of network nodes is fixed and it is 1000. We implement the PSRB protocol and empirically evaluate the performance of the PSRB protocol and PBFT algorithm. The goals of our evaluation are two fold. We first measure the consensus delay of the PSRB protocol when the transactions increase. We aim to establish that the time to achieve consensus of the PSRB protocol matches its theoretical analysis in Section 5. The second goal is to compare the cubical dilatation of each node in the PSRB protocol and bitcoin blockchain with the blocks increase.

6.1 Consensus delay

Experimental setup We run several experiments with different number of the network communities to measure the consensus delay of PSRB protocol and that of PBFT algorithm with same implements. We vary the number of communities from 3 to 5, and the number of transactions from 1 to 1000. The experimental results shows that the consensus delay of the PSRB protocol is much less than that of the PBFT algorithm, and the consensus rate multiplies with the number of communities growing, which agrees with the theoretical analysis.

Consensus delay The sharding technique makes the PSRB protocol concurrently executed by some number of communities, however, in the PBFT algorithm the whole network corporately execute the transactions. We separately measure the consensus delay of PSRB protocol and PBFT algorithm when the number of communities are 3 and 5, where one community is the consensus community, and the number of transactions are 1, 10, 100, and 1000. The results are plotted in Fig. 9.

Figure 9 demonstrates that the consensus delay of the PSRB protocol decreases by a multiplied rate as we increase the number of communities, and it grows exponentially with the number of transactions. However, at the same condition of the number of nodes and transactions, the PBFT algorithm delivers multiplied consensus delay than the PSRB protocol with different number of communities. With increasing the number of transactions, the difference of consensus delay between the PSRB protocol and the PBFT algorithm becomes more obvious. In addition, we observe that the time to establish the communities keeps constant and is rather shorter than the consensus delay. For example, the consensus delay is 246.6s in the condition of 100 transactions and 3 communities in the PSRB protocol, however, it wastes 511.635s to reach consensus in the PBFT algorithm. As the same, the time to reach consensus is 122.54s in the PSRB protocol when the number of communities is increased to 5. When the number of transactions grows to 1000, the consensus delay between the PSRB protocol and PBFT algorithm has the difference of 3873.42M when there are 5 communities. In summary, our experiments confirm the expected consensus speed of the PSRB protocol.

6.2 Cubical dilatation

In the PSRB protocol, each node saves the blockheader chain and some blocks. While in the bitcoin blockchain,

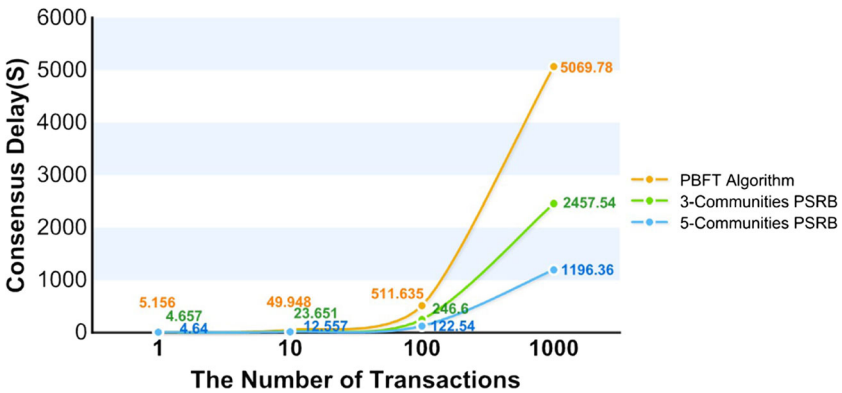

Fig. 9 The consensus delay of the PSRB and PBFT

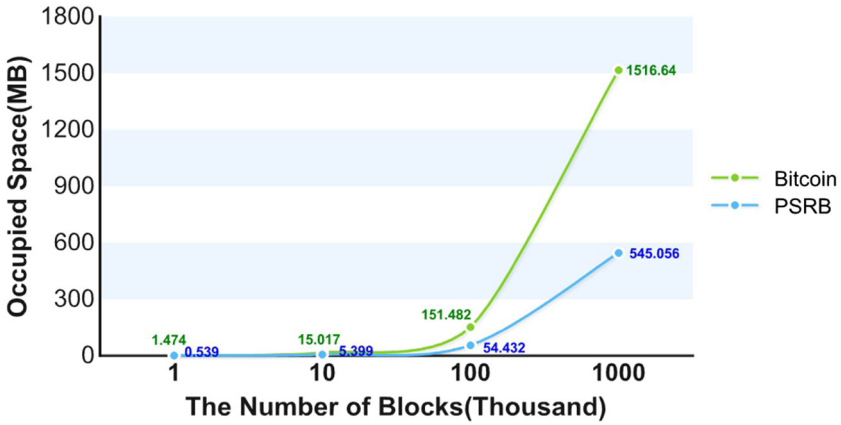

Fig. 10 The occupied space of each node in the PSRB and bitcoin blockchain

each node saves the whole blockchain, which causes the node cubical dilatation problem. We show how the PSRB protocol outperforms the bitcoin blockchain in the cubical dilatation of each network node. We experiment on the occupied space of each node in the PSRB protocol and bitcoin blockchain when the number of blocks varies from 1000 to 1000000. The results shows that in the PSRB protocol the node occupies much smaller space than that of the bitcoin blockchain. Longer of the blockchain, more slower cubical dilatation of the PSRB protocol than that of bitcoin blockchain. Figure 10 indicates the occupied space of each node in the PSRB protocol and bitcoin blockchain.

Figure 10 shows that in the PSRB protocol the occupied space of each node is much less than that in the bitcoin blockchain. At the same length of blockchain, the occupied space of each node in the PSRB protocol is about $1/5$ to $1/2$ of that in the PBFT algorithm. For example, when the length of blockchain is 10000, the occupied space of each node separately are $5.3996115M$ and $15.017048M$ in the PSRB protocol and bitcoin blockchain, and the difference is about $10M$. When the length of blockchain is 1000000, the occupied space of each node is $545.0563M$ in the PSRB protocol, however, $1516.6411M$ in the bitcoin blockchain, which is about $1000M$ more than the PSRB protocol. In summary, the PSRB protocol has slower cubical dilatation than bitcoin blockchain, and our experiment confirm the expected dilatation of each node in the PSRB protocol.

7 Conclusions

The pruneable sharding-based protocol has high efficiency, small cubical dilatation, slow capacity expansion and better scalability. Except for the high efficiency brought by the sharding technique for concurrently processing transactions, the PSRB protocol has low message complexity and it is the same order with the number of participants. Besides, each node just conserves and the new nodes download the blockheader chain and some blocks, which can still guarantee the system security. The main blockchain saves the blockheader chain and a few of last blocks involved with the blockchain forking, which meliorates the main chain of tremendous storage pressure, and relives the system of large capacity expansion problem. Furthermore, from three aspects we also analyze the protocol scalability. With the number of nodes growing, the PSRB protocol can achieve the transactions consensus in a linearly decreasing time. Moreover, the number of conformed transactions and established blocks per unit time increase linearly. The experiments in Section 6 on one hand verify that the performance of the PSRB protocol matches its theoretical analysis, on the other hand experiment the practical performance of the PSRB protocol in more detail, which guarantees its practicability.

Acknowledgements This work was supported by the Major Nature Science Foundation of China (No. 61370078, No. 61309016), the National Natural Science Foundation of China (No. 61702404), the China Postdoctoral Science Foundation Funded Project (No. 2017M613080), the Fundamental Research Funds for the Central Universities (No. JB171504).

References

1. Judmayer A, Stifter N, Krombholz K, Weippl E (2017) Blocks and chains: introduction to bitcoin, cryptocurrencies, and their consensus mechanisms. Synthesis Lectures on Information Security Privacy, & Trust 9(1):1–123
2. Anjum A, Sporny M, Sill A (2017) Blockchain standards for compliance and trust. IEEE Cloud Computing 4(4):84–90
3. Bruggeman J (2018) Consensus, cohesion and connectivity. Soc Netw 52:115–119
4. Guo B, Zhang D, Yang D (2011) Read more from business cards: Toward a smart social contact management system. In: IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology, IEEE computer society, pp 384–387
5. Lin I-C, Liao T-C (2017) A survey of blockchain security issues and challenges. IJ Netw Secur 19(5):653–659

6. Gervais A, Karame GO, Wüst K, Glykantzis V, Ritzdorf H, Capkun S (2016) On the security and performance of proof of work blockchains. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM, pp 3–16
7. Kreku J, Vallivaara VA, Halunen K, Suomalainen J (2017) Evaluating the efficiency of blockchains in iot with simulations, In: Proceedings of the 2nd international conference on internet of things, Lotbds, pp 216–223
8. Ruta M, Scioscia F, Ieva S, Capurso G, Di Sciascio E (2017) Semantic blockchain to improve scalability in the internet of things. Open Journal of Internet Of Things (OJIOT) 3(1):46–61
9. Chepurnoy A, Duong T, Fan L, Zhou H-S (2017) Twinscoin: a cryptocurrency via proof-of-work and proof-of-stake. IACR Cryptology ePrint Archive 2017:232
10. Kiayias A, Konstantinou I, Russell A, David B, Oliynykov R (2016) A provably secure proof-of-stake blockchain protocol. IACR Cryptology ePrint Archive 2016:889
11. Evans JD, Kessler RR (1992) Dpos: a metalanguage and programming environment for parallel processing. Lisp and symbolic computation 5(1-2):105–125
12. Nakamura J, Araragi T, Masuzawa T, Masuyama S (2014) A method of parallelizing consensuses for accelerating byzantine fault tolerance. IEICE Trans Inf Syst 97(1):53–64
13. Driscoll K, Hall B, Sivencrona H, Zumsteg P (2003) Byzantine fault tolerance, from theory to reality. In: Computer safety, reliability, and security, 22nd international conference, vol 3. Springer, pp 235–248
14. Oom Temudo de Castro M (2002) Practical byzantine fault tolerance. ACM Trans Comput Syst 20(4):398–461
15. Chepurnoy A, Larangeira M, Ojiganov A (2016) Rollerchain, a blockchain with safely pruneable full blocks. arXiv:1603.07926
16. Eyal I, Gencer AE, Sirer EG, van Renesse R (2016) Bitcoin-ng: a scalable blockchain protocol. In: 13th USENIX symposium on networked systems design and implementation, pp 45–59
17. Cattell R (2011) Scalable sql and nosql data stores. ACM Sigmod Record 39:12–27
18. Glendenning L, Beschastnikh I, Krishnamurthy A, Anderson T (2011) Scalable consistency in scatter. In: Proceedings of the 23rd ACM symposium on operating systems principles. ACM, pp 15–28
19. Park JH, Park JH (2017) Blockchain security in cloud computing: use cases, challenges, and solutions. Symmetry 9(8):164
20. Chen D, Zhao H (2012) Data security and privacy protection issues in cloud computing. In: Computer science and electronics engineering (ICCSEE), vol 1, IEEE, pp 647–651
21. Kraft D (2016) Difficulty control for blockchain-based consensus systems. Peer-to-Peer Netw Appl 9(2):397–413
22. IBM (2017) Hyperledger https://www.hyperledger.org/projects/fabric/

Xiaoqin Feng received the B.S. degree in information and computing science from Xidian University, Xi'an, China, in 2016. She is currently studying for the Ph.D. degree in cyberspace security, Xidian University. Her research interests include blockchain, consensus mechanism, and blockchain applications.

Jianfeng Ma received the B.S. and M.S. degrees in department of Mathematics from Shaanxi Normal University in 1985 and in department of Computer Science from Xidian University in 1988, respectively, and the Ph.D. degree in computer software and telecommunication engineering from Xidian University, Xi'an, China, in 1995. From 1999 to 2001, he was a research fellow with Nanyang Technological University of Singapore. He is currently a Full Professor and a Ph.D. Supervisor with the Department of Computer Science and Technology, Xidian University, Xi'an, China. His current research interests include information and network security, wireless and mobile computing systems, computer networks, coding theory, and cryptography. He is a member of the China Computer Federation.

Yinbin Miao received the B.E. degree with the Department of Telecommunication Engineering from Jilin University, Changchun, China, in 2011, and Ph.D. degree with the Department of Telecommunication Engineering from xidian university, Xi'an, China, in 2016. He is currently a lecturer with the Department of Cyber Engineering in Xidian University, Xi'an, China. His research interests include information security and applied cryptography. Email: ybmiao@xidian.edu.cn

Qian Meng received the M.S. degree from the School of Science, Hangzhou Normal University, Hangzhou, China, in 2016. She is currently pursuing the Ph.D. degree with the Department of Telecommunication Engineering, Xidian University, Xi'an, China. Her research interests include information security and applied cryptography.

Ximeng Liu received the B.Sc. degree in electronic engineering from Xidian University, Xi'an, China, in 2010 and Ph.D. degrees in Cryptography from Xidian University, China, in 2015. Now, he is a full professor at College of Mathematics and Computer Science, Fuzhou University, China. Also, he is a research fellow at School of Information System, Singapore Management University, Singapore. He has published over 100 research articles include IEEE TIFS, IEEE TDSC, IEEE TC, IEEE TII TSC and IEEE TCC. His research interests include cloud security, applied cryptography and big data security. Email: snbnix@gmail.com

Qi Jiang received the B.S. degree in computer science from Shaanxi Normal University in 2005 and Ph.D. degree in computer science from Xidian University in 2011. He is now an associate professor at School of Cyber Engineering, Xidian University. His research interests include security protocols and wireless network security, cloud security, etc.

Hui Li received the B.Eng. from Harbin Institute of Technology in 2005 and Ph.D degree from Nanyang Technological University, Singapore in 2012, respectively. He is an Associate Professor in School of Cyber Engineering, Xidian University, China. His research interests include data mining, knowledge management and discovery, privacy preserving query and analysis in big data.