Dizar: An Architecture of Distributed Public Key Infrastructure Based on Permissoned Blockchain

Qianyi Dai[1], Kaiyong Xu[1], Leyu Dai[1], and Song Guo[1]

Zhengzhou Information Science and Technology Institute, 450001 Zhengzhou, China

Abstract. With the current blockchain-based Public Key Infrastructure (PKI) being in its early stage of R&D, it is suffering from many shortcomings, such as its reliance on the centralized Certificate Authority (CA), the faulty identity registration and verification mechanism, and the difficulty in certificate management. As a result, the existing blockchain based PKI has trouble in adapting to a distributed network. Therefore, we have proposed Dizar: A distributed PKI architecture based on permissoned blockchain. Dizar architecture is designed with a distributed ledger operation system that can verify security. Based on no certificate authentication, electronic certificates with legal identities in the network are registered in a secure and verifiable permissioned blockchain, thus realizing the full-cycle management of the issued electronic certificates. The performance of Dizar is analyzed and compared with previous protocols. The results show that the Dizar architecture has better adaptability to a distributed network.

Keywords: Distributed PKI · Permissioned blockchain · Distributed ledger · No-certificate authentication.

1 Introduction

PKI refers to an infrastructure system based on public key theory. It manages terminals or applying corresponding public and private key pairs, and is used to verify the legal relationship between the subject and the corresponding key pairs. The PKI model widely used now issues legal electronic certificates to public key subjects through a Certificate Authority (CA) based on a Certificate Policy (CP) and a Certificate Practice Statement (CPS), providing authentic, integral and confidential cryptographic services to the public key subjects. However, given the frequent incidents in recent years (e.g., Stuxnet [10], Comodo [21], DigiNotar [7] and Trustwave [8]) , the fragility of the traditional CA-based centralized PKI is fully demonstrated. Besides, the applications of PGP-based certificate information model in the distributed network still needs to be improved due to its short trust chain and coarse granularity in measuring trust levels. Therefore, a new distributed PKI architecture is needed to solve the flaws of existing PKI and PGP, so as to adapt it to the dynamically changing distributed networks.

Originally proposed by Nakamoto [19], a blockchain aims at building a tamper-proofing and traceable block data structure in an open Peer to Peer (P2P) network through transparent and credible rules, to realize data sharing, auditing and management in a network composed of multiple sites or institutions. Any node can query the transaction history through blockchain transactions. Since a P2P network is a distributed system among peers, each peer node has the same access on its network, and has both client and server identities. A permissioned blockchain is an upgrade of the blockchain represented by Bitcoin, which only allows nodes with high credibility to verify transactions [2]. A permissioned blockchain is only open to specific groups. Each group is composed of a plurality of highly credible nodes and independently maintains block data within its own group domain, avoiding the risk of data disclosure caused by transparency across the network and offering high transaction throughputs and identity authentication efficiency.

A blockchain offers a credible distributed data storage platform based on multiple nodes in a distributed environment. And, it has no center. To address the over-reliance of PKI on a centralized CA, the blockchain and the PKI system are combined to realize a PKI based on blockchain, which has become a new research direction. At present, the main research protocols mainly include the following.

NameCoin [18] is an encrypted currency derived from Bitcoin. It is designed as a decentralized DNS and named after the ".bit" address. The self-signed certificate of Transport Layer Security (TLS) protocol of the node domain in NameCoin is written into the DNS address as auxiliary information, and is recorded into the blockchain after being verified by all nodes in their mining process. During a TLS handshake, a TLS client can publicly query the TLS self-signed certificate to verify the identity of the node domain. [11] proposed a CA-free distributed PKI based on NameCoin: Certcoin. Certcoin retains the identities of the nodes, which register both online and offline key pairs for their identities. A node uses the public key to register its identity, searches, verifies and revoks the public key of the given identity, thus realizing the basic operation of a traditional PKI. [4] proposed a privacy-aware PKI model: PB- PKI. This model does not directly link a user's real identity through the public key, but protects the online key through the offline one, thus securing the user's real identity, conducting multi-party authentication the registered nodes by all nodes. Meanwhile, PB-PKI divides the user's privacy level into global privacy and proximal privacy, and discloses different degrees of privacy for different application scenarios, thus reducing the risk of disclosure. However, Namecoin, Certcoin and PB-PKI suffer the same issues: none has an authentication mechanism. Any node that first applies for an identity will own it. A malicious node, being aware of the identity registration policy of a legitimate node, can impersonate a legitimate node to cheat other users by registering the identity, and the legitimate one will not be allowed to revoke it.

The IKP protocol (a platform for automated response of unauthorized certificates), proposed by Matsumoto et al [17], addresses the improper behaviors

of a centralized CA to issue unauthorized certificates. It designs an intelligent contract based on multi-point responses to stimulate CA to issue certificates correctly, with multiple nodes jointly supervising the legitimacy of CA's behabiors of issuing certificates and actively rejecting unauthorized issuance. The IKP protocol legally issues certificates through providing economic incentives to the CA, imposes penalties on unauthorized CA issuance, and rewards whistleblowers who report unauthorized issuance. However, this protocol still uses a small number of CA to issue certificates, and still has inherent defects of a traditional PKI based on centralized CA, e.g., high centralization, single failure point, performance bottleneck and high cost.

[16] proposes a certificate-based PKI authentication system based on Ethernet. By defining multi-point authentication contracts, it addresses the excessive communication loads in traditional PKI certificate management, Certificate Revocation List (CRL) and Online Certificate Status Protocol (OCSP). The current X.509 Certificate Standard only issues certificates for user identities, yet unable to sign certificates for fined-grained identity attributes. To Address this issue, Al-Bassam [1] improved it based on intelligent contracts, and improved its authentication of attribute information. The attributes corresponding to the user's identity are trustworthy should the identity be authenticated, and the transfer of trust between the user's identity and attributes is realized. However, the certificate management protocols proposed by [16, 1] are complicated, and the whole cycle management process is not perfect: the process of certificate revocation and recovery is not defined.

By constructing a certificate management platform based on blockchain in cloud, [9] ensured the security and reliability of certificate authentication with the security and reliability consistency of blockchain technology. By doing this, it solved the issues of identity and certificate management in personal cloud, and improved the efficiency and security of authentication before interoperability among different clouds. [27] proposed an efficient cross-domain authentication protocol based on blockchain technology, increasing the efficiency of the existing protocols. This protocol is designed with a trust model based on blockchain CA to improve the efficiency of certificate authentication. However, [9] and [27] are also based on B/S authentication mechanism. Such reliance will cause performance bottlenecks and distributed denial of service (DDoS).

In view of the above-mentioned defects in a PKI based on blockchain, a distributed PKI architecture based on permissioned blockchain is designed: Dizar. The Dizar architecture is designed with a non-certificate authentication process, and runs the permissioned blockchain in the consensus mode of the ledger designed in this article. By combining the blockchain with collective mining of block data, it offers a complete PKI platform in a distributed network. The main contributions of this article are as follows:

The design is based on non-certificate authentication, which enables the registration node to authorize the release of part of the registration information without revealing the identity. In this way, the multi-party credibility authenti-

cation of the certificate is achieved and securely registered in the permissioned blockchain.

- With the high transaction throughputs and high identity authentication efficiency of permissioned blockchain in a distributed network, the PKI function based on private blockchain is realized.
- Facilitate the multi-party credibility and multi-party maintenance of PKI service contents; traceable, revocable and recoverable node identities, keys and certificates; realizing dynamic security protection of node identities.
- This article designs a permissioned blockchain operation model of verifiable security based on the ledger model. This model treats nodes with different processing efficiency indiscriminately, making them participate collectively in the multi-point authentication process of the blocks and maintaining the reliability of transaction data. However, for different services of PKI, all nodes are classified and applied to comprehensively improve the application efficiency of Dizar architecture.
- The theoretical analysis and practical tests of Dizar architecture in this article show that the platform has good operation efficiency and scalability.

2 Relevant Researches on Blockchain

A blockchain is composed of a series of data blocks generated by cryptology associations. A data block comprises of a Block Header and a Block Body. The block header contains block height, current version code (version), Previous Block's Address (Prev-block), Target Hash Value (Bits) of the current block, random number (nonce), Hash Target, Merkle Root, Timestamp, etc. The block body records all previous transaction records and economic reward value (Gas), and all previous transaction information values are recorded in the Merkle Tree in the form of root nodes. The change of any root node recorded will affect the value of the whole Merkle Tree. The legal existence of transaction information can be verified as per the binary tree. A typical blockchain network is composed of nodes based on P2P networks. Each node maintains the consistency of the ledger data by executing a consensus algorithm.

The blockchain is operated as follows: when user A is making a transaction with user B, user B will generate his private key and address (that is, the result of the public key encoded by Base58) through the address generation algorithm. By digital signature, user B sends the wallet address to user A. User A initiates a transaction to user B's address. Meanwhile, user A will generate a corresponding block, and send the transaction with his own digital signature authorization to user B. Then, user A broadcasts the transaction to all nodes in the whole network. Miner nodes in the network begin to compete for bookkeeping rights until the confirmation authorization time of the block expires. The process of competing for bookkeeping rights is as follows: first, the hash value of Merkle root node is calculated; second, the nonce in the block header is continuously adjusted so that the SHA-256^2 value is smaller than the target difficulty value. The process of competing for bookkeeping rights is also called "mining". The

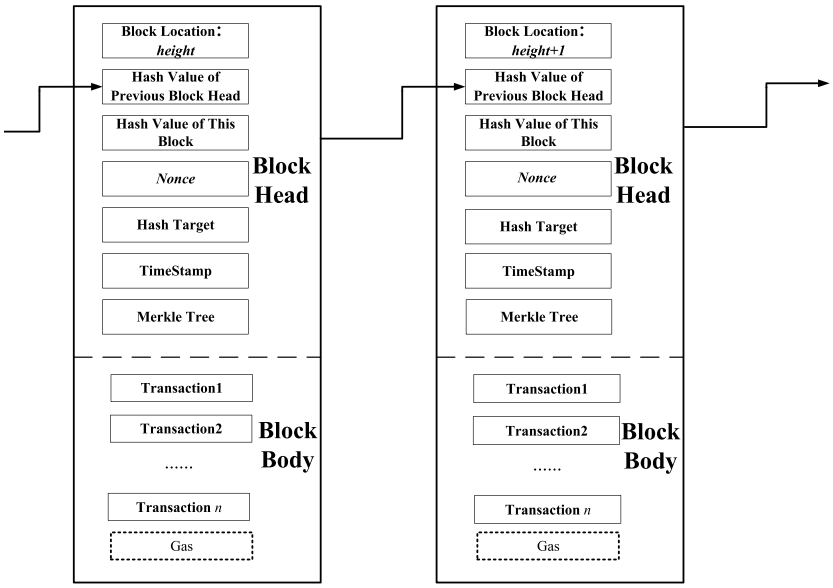

Fig. 1. Structure of a Blockchain

calculation process requires corresponding calculation powers. Therefore, this working mechanism is called the Proof of Work (POW). The node that first calculated and released the correct nonce will be rewarded with Gas.

Based on the hashing principle of the blockchain structure, the credibility of the blockchain will increase with the increasing of the blocks so long as most nodes in the blockchain are credible. If an adversary tries to tamper with the blockchain that has been saved by the nodes, he must construct a chain longer than the recognized main chain. The adversary needs to recalculate all blocks after the very block with more calculation powers than all nodes combined in the blockchain network. For a increasing blockchain, it is almost impossible for the adversary to complete the modification of the corresponding block information [12, 5].

3 Blockchain Node Ledger Operation System Based on Verifiable Security

This article uses a hybrid model based on proxy and pure peer-to-peer architecture between nodes to provide routing, security and topology information. It should be noted that the status of proxy nodes in this article are equal to other terminal nodes. That is, they have the same access as others, and only play servicing and auxiliary functions in the network, instead of acting as traditional centralized management nodes. In order to achieve global credibility of

block data, we designed a corresponding distributed node architecture and a blockchain operation system based on the ledger system.

3.1 Network Node Architecture

In this article, a private blockchain architecture is adopted, and the bitcoin system [23, 15] designed by Nakamoto is utilized. By default, the following basic facts exist in the network system:

- The nodes are linked through insecure channels;
- Any node in the network has access to the complete ledger records in the blockchain;
- There may be delays in publishing results on the blockchain between participating nodes of the protocol;
- The as-confirmed ledger records on the blockchain cannot be tampered with. The nodes are classified and their functions and characteristics are defined, as shown in Figure 2.

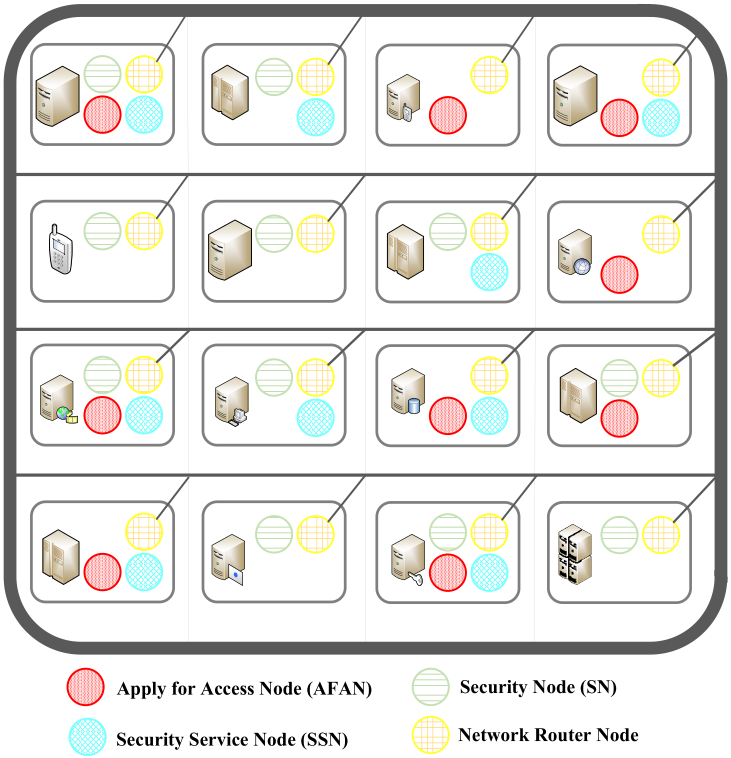

Fig. 2. PKI Service Architecture Node Architecture Based on Permissioned blockchain

Security Node (SN): a node whose network has been verified secure, usually refers to a mobile terminal node. If the security nodes want to authenticate each other's identities, they can issue certificate authentication requests to a security service node. An SN dynamically chooses whether to participate in the bookkeeping right competition process or whether to adopt the light node strategy according to its own resource processing capacity.

Apply for Access Node (AFAN): A node that applies for access in the network system. It may be a good node or a malicious node. Therefore, the node applying for access needs to send relevant authentication information to the Registration Nodes. It can only be recorded into the blockchain and obtain its identify access permission after its identity legitimacy are verified by all nodes.

Registration Node (RN): responsible for user proxy node registration in the network, verifying the legitimacy of AFAN, and providing partial registration private key information for legitimate AFAN. A RN has strong data processing capability, especially in terms of bilinear pairings. It should be noted that the RN blockchain's physical infrastructure, which acts as proxy service nodes, does not have an authorization function and is not a centralized management node.

Security Service Node (SSN): records all the published blockchain records in the network, and forms a blockchain physical infrastructure group with RN. It has strong data backup, disaster tolerance and fast blockchain retrieval capabilities. The network is composed jointly by RN and SSN, which are equal to each other. They jointly participate in the identity authentication of AFAN and in the competition of bookkeeping rights of all blocks. The legal basis of SSN data is determined by the PoW mechanism in the blockchain.

Miner Nodes (MN): the blockchain includes RN and SSN, as well as some security nodes actively join in. All the nodes in the miners' nodes adopt the same consensus mechanism and authentication protocol to jointly participate in the identity authentication of the AFAN. The generated blockchains are identical and jointly responsible for the legality and security of the data.

According to the functional features of nodes in the network, AFAN and SSN are credible nodes serving as permissioned blockchains responsible for authentication and proxy of the corresponding network access information. Based on the nature of P2P networks, miners or SSN are routing nodes, who will use Gnutella protocol to jointly participate in message broadcasting. RN can be functionally implemented as a distributed credible CA; SSN, as a credible data backup library, also participates in mutual identity authentication among SSN and keeps the same backup records as all nodes do. Besides, SSN is responsible for mutual authentication among common SSN to reduce the loads of secondary authentication on RN.

3.2 The Operating Mode of A Ledger-based Permissioned blockchain

The ledger-based licensing system designed in this article regards the blockchain as a credible public ledger maintained by all nodes. Each block is a set of transaction record ledgers. The Input Collection and Output Collection are recorded in the block body, and the PKI service information to be recorded in the blockchain is regarded as posting information requiring authentication by nodes across the network. During the operation, the legality of the information to be recorded is verified by proxy nodes, and is written into the input information of the block body and encapsulated in the block. Then, the block is sent to the miner nodes for multi-party authentication by competing for bookkeeping rights; when each node receives a new block, it verifies the legality of the block and the correctness of its contents. If the block is legal, it receives and stores the block and replies the correct receipt to the network. If the block is illegal, it discards the block and reply the error receipt to the network. When the node has received the correct receipts for more than half of the blocks, it writes the input records into the output, and link them to its own blockchain according to the time-stamp sequence, thus forming credible PKI service records (as shown in steps 1 to 6 in Fig. 3).

In order to further illustrate the process of a ledger-based blockchain system, we have defined, proved and analyzed its running mode.

Definition 1. *Similar to Bitcoin's trading process [23], the blockchain in this article is defined as follows:*

- ***Input Collection:*** *The information collection of the services initiated by proxy nodes in the network to the miner node group within a certain period of time.Iutput Collection is represented by "S". The types of input service elements defined in this article include node registration, public key update and key revocation, or S_0, S_1 and S_2, respectively. The specific input collection is as follows:*
  $S = \{Sx_y | Sx_y \in S$, where x represents the input service element type of the node and y represents the number of the node requesting services.$\}$
- ***Output Collection:*** *A collection of legal information that has been verified by the miner node group in the blockchain within a certain period of time. Output Collection is represented by "R": $R = \{R_0, R_1, R_2, \ldots, R_M\}$. The status of each output information will only be "Unspend" or "Spend".*
- ***Transaction:*** *The process during which the nodes responsible for the corresponding services in the miner node group construct the transaction input information into corresponding blocks, and send the blocks to all the nodes in the group in order to collectively compete for bookkeeping rights. The essence of this process is the mapping from the elements of input collection to the corresponding ones of output collection, with Tr representing the transaction function: $R_i = Tr(S_i), S_i \in S$. With the different service contents in the transaction, Tr_1, Tr_2 and Tr_3 represent node registration, key update, and key revocation operation, respectively.*

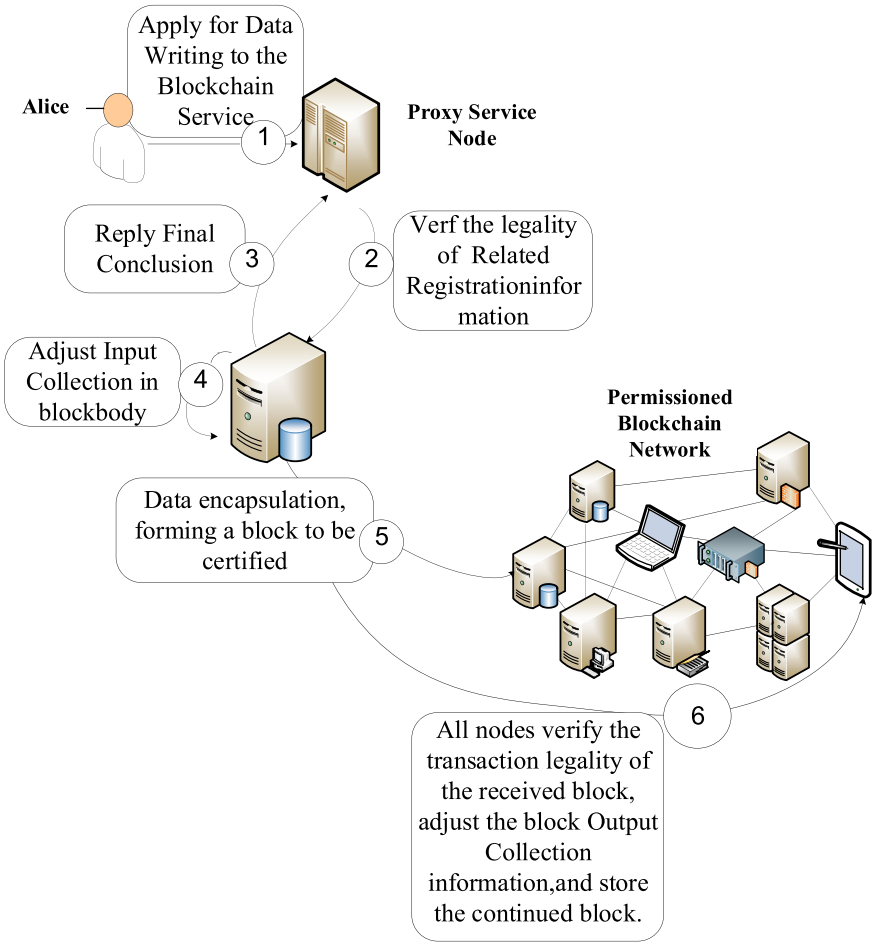

Fig. 3. Licensing Process Based on a Ledger System

– ***Round:*** *The number of times a miner node, during the process of updating its own transaction book, has received a correct receipt from all miners across the network.*

Definition 2. *The legal transaction behaviors on the blockchain (commandtype) defined in this article include: commandtype* $\rightarrow$ *{register, update, revoke}.*

Specifically, in the multi-party verification process by a miner's node group with v nodes, the j-th node performs the PoW for the PoW initiated by the i-th miner node. The corresponding proof value given by node j is: $y_{ij} = \mathcal{F}_{HashTarget}(nounce, x_{ij}) \rightarrow, \{0, 1\}$*, 0 and 1 represent Unspend and Spend respectively.*

y_i *is defined as the overall evaluation of the PoW initiated by all nodes in the network for the i-th miner node. To make a transaction legal, the Spend status of the information is proved only when it is approved by more than 51%*

of the nodes. Information whose security cannot be proved $y_i = \sum_{i=0}^{N} y_{ij} \geq \nu/2$ *is regarded as Unspend.*

Table 1. The corresponding ledge-based blocks designed in this article

Block Content	Block part	Note	Strlen/bit
ID	Block head	The user ID, which is the anonymous value of the public key	8
Prev Block Hash	Block head	Hash value of the previous block head	32
TimeStamp	Block head	Timestamp, based on user system generation	4
Version	Block head	Blockchain version number, 1.0 in this paper	4
Merkle Root Hash	Block head	Merkle Root value for transaction content	32
Block Height	Block head	Block height	4
Hash Target	Block head	Hash value of the previous block head	8
Nonce	Block head	Random value to adjust the block	4
Fulfillments	Block Body	Satisfy the list of conditions, the signature value and hash value of each book	72
Operation	Block Body	Operation type register,revoke,update	2
Data	Block Body	Transaction data	-
Input Collection	Data	The data of all pending transactions	-
Output Collection	Data	Transactions that have been certified by multiple parties as legitimate	-
Data_Hash	Data	Hash value of the Data	32
Gas	Data	Competitive accounting awards	4
Payload	Input Collection	Data load, related data values	-
Declarative Signature	Output Collection	Miner node has the right to sign the accouting information after successful mining	-

The miner node group operates on a consensus system based on private blockchain ledgers, and the process of competing for bookkeeping rights is essentially a process of constantly adjusting their own output collections. Referring to the traditional block format [22], we designed the corresponding block based on the ledger system, as shown in Table 1. When any proxy node sets up a corresponding input collection and generates a corresponding block, the block will be sent to all the nodes in the network for authentication. During network-wide multi-node authentication, the legality of input information of each block received by the miner node will be verified. If the information is legitimate, it is

written into the output collection, while the corresponding local block's storage copy is adjusted and the output information is written therein; If it is illegal, the blocks will be discarded. When the correct receipts of a node exceed half of the nodes, the block can be written into its own blockchain. The consensus algorithm based on blockchain ledgers of miner nodes can be described by Algorithm 1.

Algorithm 1 Improved POW consensus base on ledgers system

Input: *Blockchain (height-1)*, S_i, R_i, round, *nonce*, HashTarget
Output: Blockchain (height)

Start:
The proxy node constructs the block to be authenticated: block (blockhead, S_i, R_i, nonce), $S_i = \{Tr_1, Tr_2, \ldots, Tr_n\}$, R_i = null, nonce = null,
round = 1
decided = false
Broadcast myReport (block (blockhead, S_i, R_i, nonce))
while true **do**
    The network node verifies the legitimacy of the transaction records of S_i, Write legal transaction records to $R_i = \{Tr_1, Tr_2, \ldots, Tr_m\}, m < n$
    Enumerate the nonce in SHA-2562(blockhead, nonce) < HashTarget
    Reconstructs the block: block (blockhead, S_i, R_i', nonce')
    The miners node broadcasts the block information after the restructuring
    Continue waiting for network replies until updates are received myReport (block (blockhead, S_i, R_i', nonce'))times nonce $> v/2$
    $Blockchain(height) = Blockchain(height-1) + block(blockhead, S_i, R_i', nonce')$
    if all the myReport hasthe same R_i', **then**
        Broadcast Propose (block (blockhead, S_i, R_i', nonce'), round);
    else
        Broadcast Propose ($\perp$, round);
    end if
    if (decided == false) **then**
        Report myReport (block (blockhead, S_i, R_i', nonce'), round + 1)
        Determine decision value $R_i, R_i' = S_i$, writes the R_i into Output, broadcasts update block
    end if
    Adjust the local Block copy value
    Continue to wait for, until half round have the same Propose information
    if if (all receive block blocks are the same) **then**
        $R_i' = S_i$
        decide = true
    else if (at least one block proposal contains the same as the local copy R_i') **then**
        $R_i' = Si$,
    else
        Random choose R_i, $\Pr[S_i = 1] = p$, $\Pr[S_i = 0] = 1 - p$
    end if

32: $round = round + 1$
33: Broadcast myReport (block (blockhead, S_i, R_i', nonce'), round)
34: **end while**

Algorithm 1 is analyzed below.

Effectiveness analysis: Note that the effectiveness of consensus is equivalent to: if the block inputs of all nodes are S_i, it must be the last decision value assigned to R_i; Otherwise, R_i will only approve or reject, so the effectiveness will be verified automatically.

Assume that all nodes have input ledger proposals of $R_i' = S_i$, in which case all nodes start proposing S_i as the final decision value of Ri in the first round. Since all nodes can only receive the ledger proposal of $R_i' = S_i$, they will also take S_i as R_i' decision value (Line 17) and exit the loop in the next round.

Consistency analysis: A node only sends a reply supporting the input proposal when it receives a message that more than half of the nodes contain S_i (Line 8). Therefore, it is not possible for the proposal to be approved and rejected at the same time in the same round.

Assuming T is the node whose first decision value is $R_i' = S_i$ in the r-th round, it must have received proposals from a majority of ledgers recommending r (Line 17) in the r-th round. If a node receives more than half of the ledger proposals recommending the same value, it will accept the ledger proposal (changing the local R_i to R_i') and exit in the next round. Since there is no ledger proposal recommending other results in r-th round, it can be inferred that no node will select another different ledger proposal in the same round.

Any node with $T' \neq T$ may experience the following two situations: 1. It receives more than half of the ledger proposals recommending $R_i' = S_i$ in r-th round and decides to adopt R_i'. In this case, the requirement to reach consensus is satisfied directly, and the nodes will not get stuck; 2. This node does not receive unanimous recommendation from more than half of the proposals. Since node T receives most of the ledger proposals recommending S, each node must receive messages from more than half the nodes before ending. This means that each node receives at least one ledger proposal recommending S (Line 16).

Therefore, all the nodes set their respective R_i as $R_i' = S_i$ (Line 17, 18) in r-th round. All nodes will broadcast S at the end of r-th round, so all will propose $R_i' = S_i$ at $r + 1$ round. The nodes already decide to adopt this proposal in r-th round will terminate running in round $r + 1$ and send an additional myReport message (Line 13).

All other nodes will receive S recommended by more than half of the nodes in round $r + l$, and decide to adopt in this round. Meanwhile, they also send a myReport message. In this way, some nodes would have already terminated running in round $r + 2$, and the remaining nodes would have received enough myReport messages (Line 6). These nodes send a Propose message and a myReport message, and decide to adopt the ledger proposal in round $r+2$ to terminate running.

Termination analysis: We already know from the proof of consistency that, if a node receives more than half of the ledger proposals recommending r, all nodes will terminate their running with maximum two rounds.

Let's assume that none of the nodes has received more than half of the proposals recommending same value. In such a round, some nodes may update their values to R_i' based on a received proposal. Other nodes randomly select 0 or 1, with the probability of selecting the same value being at least p_n. A Boolean consensus can be reached within the expected time $O(2^n)$ if less than $v/2$ nodes collapse.

3.3 Security Proof of Private Blockchain Based on Ledger System

Theorem 1. *Each output entry must correspond to the corresponding input entry, and the number of elements in the output collection is smaller than number of elements in the input collection; During transactions, there are individual inputs and Unspent Transaction Outputs (UTXO). That is,* $m \leq n$.

Proof. $\because R = Tr(S)$, R can only be 0 or 1, which is the unit price decision value (only Spend or Unspend)

$\therefore S \to R$, if the transaction mapping Tr is an injective mapping, the number of image set elements must be less than the number of source set elements.

$\therefore m \leq n$.

Proof is complete.

During blockchain transactions, a miner node concludes that the transaction information contained in the block is verified and the block can be written into the blockchain if it receives correct receipts when more than half of all nodes are submitting the transactions to the network. Proof is complete.

From a functional point of view, the process of combining UTXO transaction set with PKI will be regarded as CRL in a traditional PKI.

Theorem 2. *When the node data is large enough, the blockchain mode system error probability Fp(S) (Failure Probability) based on the ledger system is close to 0.*

Proof. The multi-node authentication process of the ledger system shows that this proof is equivalent to proving that the decision group composed of more than half of the nodes has error 0.

Assuming that each node gives a fixed probability of error reply to the j-th node's authentication request as p_j, a Chernoff bound is formed as all nodes in the network adopt identical protocol for the same information authenticated. The following conclusions are reached: $y_1 \cdot, \ldots, y_n$, are all variables with independent and same distribution, and obey binomial distribution, i.e., $P_r[y_{\cdot j} = 1] = p_j$, $P_r[y_{\cdot j} = 0] = 1 - p_j$ and for $Y := n_j = 1y_{\cdot j}$.

And: $\mu : E[Y] \sum_{j-1}^{n} p_j, \mu = np_j$. Here, Y is the number of nodes with correct receipts.

In particular, $0 < \delta < 1$: $P_r[Y \leq (1-\delta)\mu] \leq \mathrm{e}^{-\mu\delta^2/2}$.

In a network system, more than $\lfloor v/2 \rfloor + 1$ node decision groups verify the authentication tuples issued in the network. Each node with a cardinality of $\lfloor v/2 \rfloor + 1$ can form a decision group. If more than half of the decision groups are attacked by Eclipsed [26], only $\lfloor v/2 \rfloor$ nodes can work normally at most. Otherwise, there must be at least one decision group to complete the decision.In order to estimate the asymptotic failure probability of $\lfloor v/2 \rfloor + 1$ node, $F_p(S) = Pr[\lfloor Y \leq 2/v \rfloor] \leq Pr[Y \leq 2/v] == Pr[Y \leq (1-\delta)\mu]$ can be obtained based on Chernoff inequality.

$\because \delta = p_j - p_j^2$, and if the node is considered to fail with a high probability, $1/2 < p_j \leq 1$ is taken.

$\therefore 0 < \delta \leq 1/4$, according to Chernoff bound, $F_p(S) \leq \mathrm{e}^{-\mu\delta^2/2} \in \mathrm{e}^{-\Omega(v)}$ is obtained.

$\lim\limits_{v\to\infty} \mathrm{e}^{-\Omega(v)} = 0$ That is, when the number of nodes is large enough, the running error probability of blockchain based on ledger system tends to 0. Proof is complete.

4 Distributed PKI Services Based on Permissioned blockchain

The above platform node system's security analysis shows that a blockchain based on the ledger consensus system is capable of achieving global consistency, validity and termination of all node's ledger results. Combined with the specific implementation process of a ledger-based private blockchain mode, a node-based PKI full-cycle management strategy with identities at its core is realized by designing the corresponding block body structure in the account book, with the service information to be authenticated being issued by the proxy nodes, multi-party authentication being performed by miner nodes in the network, and the authenticated results being written into the blockchain (as shown in fig. 4). The following functions of a distributed PKI system are implemented: 1. The AFAN in that network can register the corresponding public-private key pairs and the certificates based on the identities; 2. Update the public and private keys and certificates corresponding to the previously registered identities; 3. Retrieve, update, and revoke public and private key pairs corresponding to a given identity.

The operations in Fig. 4 respectively describe the processes of AFAN and SN completing identity registration, certificate update, certificate revocation/key recovery, certificate search and key verification services through the proxy node groups. Among them, the first three are blockchain writing services and the last ones are data query services.

The process of blockchain writing services is as follows: AFAN and SN respectively send the information to be written into the blockchain to the proxy node group, which includes RN and SSN; the proxy node group checks the legality of the applicant data and generates transaction information Tx_n(Tx_1 and Tx_4 are registration applications of the nodes, Tx_2 and Tx_5 are key renewal applications, and Tx_3 and Tx_6 are key revocation applications); the proxy nodes construct

blocks and uniformly write the transaction applications during a period of time into the input collection of the block body, and submit them to all the miners, who compete for bookkeeping rights. Miners throughout the network check the legality of the contents in the receiving blocks and the block input collection, and exhaustively calculate nonce. Then, they write legal transaction information into the output collection and generate corresponding certificate data. Illegal information is recorded as $\times$ (Tx_1, Tx_2 and Tx_3 are written into the block output collection, and Tx_4, Tx_5 and Tx_6 are recorded as illegal information). After the calculation is completed, the blocks are connected into the local blockchain to complete the process of data writing.

The process of data search services is as follows: SN send out data query applications to the data service nodes through the security nodes; the data service nodes check the legality of the applicant information and retrieve the blockchain information if it is legal; the data service nodes query and verify the information retrieved and reply the corresponding conclusions to the SN; the service process is finished.

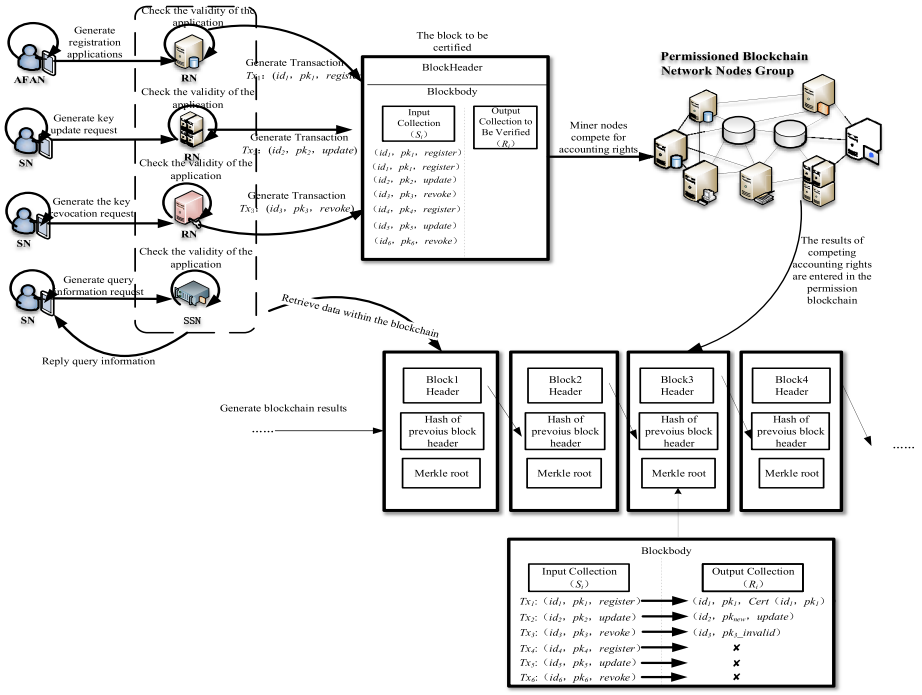

Fig. 4. PKI service system based on Private Blockchain

4.1 Description of Relevant Cryptographic Algorithms

Definition 3. *this paper adopts the relevant cryptographic algorithms in the private blockchain, which are defined as follows:*

- $KFG(1^k) \rightarrow (sk, pk)$ *Key Generator Function: input security parameter* k *and output corresponding public key* pk *and private key* sk*.*
- $Sig(sk, \mu) \rightarrow \sigma$*, Signature Algorithm: The information* μ *is digitally signed with the private key* sk*, and the result is* σ*.*
- $Vrfy(pk, \sigma, \mu) \rightarrow b \in \{0, 1\}$*, Signature Verification Algorithm: Verify through the public key* pk *whether the digital signature* σ *corresponds to the signature information* σ *of the corresponding private key* sk*.*
- $Rand() \rightarrow \mathbb{R}$*, Random Number Generation Algorithm: generate random number* $\mathbb{R}$*. the randam numbers generated in any two times are completely random.*
- $GenCert(id, pk, T) \rightarrow Cert(id, pk)$*, Certificate Generation Algorithm: the corresponding digital certificate* $Cert(id, pk)$ *is generated through the corresponding id, public key* pk *and timestamp* T*.*

The certificate version refers to the X.509 version certificate [21].

$$pk_n = f_1(pk_{n-1}, sk', N) = pk_{n-1} \times sk'(modN)$$
$$sk_n = f_2(sk_{n-1}, sk', N) = sk_{n-1}/sk'(modN)$$

Chain Key Pair Updating Algorithm: f_1 generates new public key pkn through original public key pk_{n-1}, reserved private key sk', and random number N; f_2 generates new public key skn through original private key sk_{n-1}, reserved private key sk', and random number N. In particular, according to the construction features of a blockchain, the crypto-system is based on the secp 256 k1elliptic curve of ECDSA.

And, below facts are recognized by default:

Privacy security: In the entire permissioned blockchain network, the computing environment of each default node is secure, i.e. an adversary cannot obtain the private key information of any node.

The private key cannot be forged: the digital signatures cannot be forged. Even if the adversary possesses the public key pk, he cannot obtain the corresponding private key sk. The adversary cannot generate the signature message pair (μ, σ) in polynomial time to satisfy $Vrfy(pk, \sigma, \mu) = 1$.

Initial security: this article believes that the miner node groups are the initial security node groups, which have the initial security by default.

4.2 Identity Registration

OnlineKeyPair: $(pk_{online}, sk_{online})$ and *OfflineKeyPair:* $(pk_{offline}, sk_{offline})$ are generated by the AFAN and corresponding certificates are generated. The two sets of key pairs can be encrypted and decrypted in different application scenarios. AFAN send identity authentication information to RN, who issue partial

registrated private keys to legitimate AFAN. After the AFAN generate registered private keys, the public keys, certificates and related registration information corresponding to the *OnlineKeyPair* and *OfflineKeyPair* are signed without certificate and packaged into blocks, and then released to MN node groups. After receiving the blocks, all MN verify the legality of the block contents and the non-certificate signature information and compete for bookkeeping rights. The legal blocks will be written into the main chain.

Algorithm 2 Identity Registration

Start Operation
Step 0 RN Initialization Operation
(1) RN Selects safety parameters k, p plane number circulation group G_1 and G_2. RN constructs bilinear mappings on elliptic curves: $\hat{e} : G_1 \times G_1 \to G_2$, P is the point on the elliptic curve, and G_1 is the Generator.
(2) RN random selects $\lambda \in Z_p^*$ and $g_2 \in G_1$, thereinto g_2^{λ} is selected as as the miner node group manager key. $Rand() \to s \in Z_p^*, sk_{RN} = s$ as a private key and stored securely, $pk_{RN} = s \cdot P \in Z_p^*$ is choosen based on multiplied point production generation public offering.
(3) Meanwhile, RN random selects $g \in G_1, g_1 = g^{\alpha}$, select four hash functions identified $H_1 \sim H_5$, $\boldsymbol{v} = (u_i)$, $\boldsymbol{\phi} = (f_i)$, $\boldsymbol{\gamma} = (g_i)$, $\boldsymbol{\omega} = (w_i)$ is a vector of character lengths as m, n, n, l. Miners node groups expose system parameters $params : = \{G_1, G_2, p, e, g, g_1, g_2, H_1 \sim H_5, \boldsymbol{v}, \boldsymbol{\sigma}, \boldsymbol{\tau}, \boldsymbol{\omega}, B_I, pk_{RN}\}$, there in B_I is the base name of group.
Step 1 AFAN Executes Operation
(1) AFAN generats random number: $Rand() \to \theta_1 \in Z_p^*, Rand() \to \theta_2 \in Z_p^*$, securely stores θ_1 and θ_2.
(2) AFAN generats the public-private key pair: $KGF(\theta_1) \to g^{\theta_1}, KGF(\theta_2) \to g^{\theta_2}, pk_1 = g^{\theta_1}, pk_2 = g^{\theta_2}$
(3) AFAN generates sequence random number and challenge random number: $Rand() \to seq_1, Rand() \to chlg$
(4) AFAN generates temporary anonymity: $\rho = H_1(ID_{AFAN}), PID_{AFAN} = B_I^{\rho} (\mathrm{mod} p)$
(5) AFAN randomly accesses an RN node from the RN list, denotes as RN', AFAN sends RegistrationApplication ($PID_{AFAN}, register, T_0, seq_1, pk_1, pk_2, chlg$) to RN'
Step 2 RN' Generats Partial Private Key
(1)RN' generats random number: $Rand() \to \mathcal{K} \in Z_p^*$
(2)RN' generats partial private key for AFAN: $\boldsymbol{v} = H_1(PID_{AFAN}, pk_1) = (v_1, v_2, v_3, \ldots, v_m), DID_{AFAN} = (D_1, D_2) = (g_2 U^k, g^k)$,
thereinto $U = \prod_{j=1}^{m} \boldsymbol{u}_j^{\boldsymbol{v}_j}$
(3) RN' generates a verification signature for AFAN: $Sig(PID_{AFAN} || chlg || T_0, sk_{RN}) \to \sigma_{RN'}$
(4) RN' sends RegistrationApplicationReply ($PID_{AFAN}, T_0, seq_1, DID_{AFAN}, \sigma_{RN'}$) to AFAN via secure channel
Step 3 AFAN Generates Private Key Signature and Certificate

(1) AFAN verifies the RegistrationApplicationReply legitimacy published by RN'
if $(Vrfy(pk_{RN}, \sigma_{RN'}, PID_{AFAN}||chlg||T_0, sk_{RN}) == 1)$ **then**
    accept RegistrationApplicationReply $(PID_{AFAN}, T_0, seq_1, DID_{AFAN}, \sigma_{RN'})$ generated by RN'
elseReturns to Step 2
end if
(2) AFAN calculates: $\boldsymbol{v} = H_1(ID_{AFAN}, pk_1) = (v_1, v_2, v_2, \ldots, v_m)$, $\vartheta = \prod_{j=1}^{m} (\phi_j)^{v_j}$
(3) AFAN verifies partial private key thatpublished by RN':
if $(\hat{e}(g_1, D_1) = \hat{e}(g_1, g_2)\hat{e}(D2, U))$ **then**
    Accept partial private key generated by RN', generates AFAN's own user private $keysk = \{(\theta_1, \theta_2), (D_1, D_2)\}$
else
    Returns to Step 2
end if
(4) AFAN generats random number: $Rand() \to \Psi \in Z_p^*$
(5) AFAN generates own certificate: $Rand() \to \mathbb{R}, Rand() \to \mathbb{Z}, KGF(1^{\mathbb{R}}) \to (sk_{online}, pk_{online}), KGF(1^{\mathbb{Z}}) \to (pk_{offline}, sk_{offline})$
securely stores sk_{online} and $sk_{offline}$, securely deletes $\mathbb{R}$ and $\mathbb{Z}$
$GenCert(ID_{AFAN}, pk_{online}, T_0) : = Cert(ID_{AFAN}, pk_{online}, T_0)$,
$GenCert(ID_{AFAN}, pk_{offline}, T_0) : = Cert(ID_{AFAN}, pk_{offline}, T_0)$
(6) AFAN calculates: $M = Cert(ID_{AFAN}, pk_{online}, T_0)||Cert(ID_{AFAN}, pk_{offline}, T_0)$,
$\alpha = H_4(ID_{AFAN}) = (\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_l)$, $\Omega = \prod_{k=1}^{l} (\boldsymbol{\omega}_k)^{\alpha_k}$, $\eta = H_5(M||g^{\Psi}||pk_1||pk_2)$, $\beta = H_2(M||g^{\Psi}||pk_1||pk_2) = (\beta_i)$,
$\Phi = \prod_{i=1}^{n} (\boldsymbol{\phi}_j)^{\beta_j}$, $\boldsymbol{\chi} = H_5(M||g^{\Psi}||pk_1||pk_2) = (\boldsymbol{\chi}_i, \Gamma = \prod_{i=1}^{n} (\gamma_j)^{\chi_j}$, the signature is $\sigma_{Register}(\sigma_1, \sigma_2, \sigma_3) = (D_1^{\eta}(\Phi)^{\theta_1 \eta} \Gamma^{\theta_2} \Omega^{\Psi}, D_2^{\eta}, g^{\Psi})$
(7) AFAN constructs Input information, issues registered blocks to the miners node group: $S_i : = \{PID_{AFAN}, Register, online||offline, T_1, height, Cert(ID_{AFAN}, pk_{online}, T_0), Cert(ID_{AFAN}, pk_{offline}, T_0), VerificationValues = (Cert(ID_{AFAN}, pk_{online}, T_0)||Cert(ID_{AFAN}, pk_{offline}, T_0), \sigma_{Register}, ID_{AFAN})), (pk_1, pk_2), (D_1, D_2)\}$
Step 4 MN Competes for Accounting Rights
(1) MN verify the legitimacy of AFAN identity
if (PID_{AFAN} has been registered) **then**
    The transaction fails, discards the block, broadcast to the network bulitinboard report ERROR: (PID_{AFAN}, Registration), **End Operation**
    (2) MN extracts VerificationValues from S_i, calculates $\vartheta = \prod_{j=1}^{m} (\phi_j)^{v_j}$, $\Omega = \prod_{k=1}^{l} (\boldsymbol{\omega}_k)^{\alpha_k}$, $\Phi = \prod_{j=1}^{n} (\boldsymbol{\phi}_j)^{\beta_j}$, $\Gamma = \prod_{j=1}^{n} (\gamma_j)^{\chi_j}$

end if
(3) MN verifies the legality of the signature
if $(\hat{e}(g,\sigma_1) == \hat{e}(g_1,g_2)^{\eta}\hat{e}(\sigma_2,\vartheta)\hat{e}(pk_1,\Phi)^{\eta}\hat{e}(pk_2,\Gamma)\hat{e}(\sigma_3,\Omega))$ **then**
    Writes $S_i := \{PID_{AFAN}, Register, T_1, height, Cert(ID_{AFAN}, pk_1, T_0), Cert(ID_{AFAN}, pk_2, T_0), VerificationValues = (Cert(ID_{AFAN}, pk_{online}, T_0) || Cert(ID_{AFAN}, pk_{offline}, T_0), \sigma_{Register}, ID_{AFAN})), (pk_1, pk_2), (D_1, D_2)\}$ into output information of R_i, rebuilds and stores the block, broadcasts $VerificationSuccess(nonce)$ and myReport($block(blockhead, S_i, R_i', nonce')$, round) to the network.
else
    The transaction fails, discards the block, **End Operation**
end if
(4) MN process copy statistics
if (round $> 2/\boldsymbol{v}$) **then**
    Continue the update block onto the blockchain
    End Operation
end if

4.3 Certificate Update

As the online public key certificate mainly used in the network, the nodes update the same type of public and private key pairs and certificates through updating operations when SN and MN in the network need to update the online key pairs and corresponding certificates to which the subject identities belong due to specific security requirements (e.g. online public and private key pair disclosure, online public key certificate's regular updates, etc.). In this case, the nodes do not need to re-register their identities, but only use the new and old public key pairs to jointly publish the updated transaction to ensure the security of the published information, thus realizing the binding of the new online keys with the old offline keys and the node identities.

Algorithm 3 Certificate Update

Start Operation
Step 1 AFAN Executes Operation
(1) AFAN generates sequence random number: $Rand() \rightarrow seq_2$
(2) AFAN randomly accesses an RN node from the RN list, denotes as RN", AFAN sends CertRenewalApplication ($PID_{AFAN}, certrenewal, keytype, T_1, seq_2$) to RN"
Step 2 RN" Executes Operation
(1) RN" verifies CertRenewalApplication legality, return to Step 1 if is not valid
(2) RN" generates validation success signatures for illegal CertRenewalApplication: $Sig(PID_{AFAN}||CertRenewalApplicationSuccess||T_1, sk_{RN}) \rightarrow \sigma_{RN''}$
(3) RN" Sends CertRenewalApplicationReply($PID_{AFAN}, T_1, seq_1, \sigma_{RN''}$) to AFAN via secure channel
Step 3 AFAN Update the Certificate

(1) AFAN Verifies an legitimacy published by RN" CertRenewalApplicationReply
if $(Vrfy(pk_{RN''}, \sigma_{RN''}, PID_{AFAN}||CertRenewalApplicationSuccess||T_1) == 1)$ **then**
    Accept the CertRenewalApplicationReply$(PID_{AFAN}, T_1, seq_1, \sigma_{RN'})$ generated by RN"
else
    Returns to Step 1
end if
(2) AFAN generates update key: $Rand() \rightarrow \mathbb{N}, KGF(1^{\mathbb{N}}) \rightarrow (sk_{keytype}^{new}, pk_{keytype}^{new})$
(3) AFAN generates update certificate: GenCert$(ID_{AFAN}, pk_{keytype}^{new}, T_0)$: $= Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0)$
(4) AFAN calculates $M' = Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0), \eta' = H_5(M||g^{\Psi}||pk_1||pk_2)$, generates authentication value signatures
$\sigma_{CertRenewal} = (\sigma_1, \sigma_2, \sigma_3) = (D_1^{\eta}(\Phi)^{\theta_1\eta'}\Gamma^{\theta_2}\Omega^{\Psi}, D_2^{\eta}, g^{\Psi})$
(5) AFAN constructs Input information, issues the blocks to the MN group: $S_i : = \{PID_{AFAN}, CertRenewal, keytype, T_1, height', M' = Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0), VerificationValues = (M' = Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0), \sigma_{CertRenewal}, ID_{AFAN})), (pk_1, pk_2), (D_1, D_2)\}$
Step 4 MN Competes for Accounting Rights
(1) MN extracts $VerificationValues$ from S_i, calculates $\vartheta = \prod_{j=1}^{m}(\phi_j)^{v_j}, \Omega = \prod_{k=1}^{l}(\boldsymbol{\omega}_k)^{\alpha_k}, \Phi = \prod_{j=1}^{n}(\boldsymbol{\phi}_j)^{\beta_j}, \Gamma = \prod_{j=1}^{n}(\gamma_j)^{\chi_j}$
(2) MN verifies the legality of the signature
if $(\hat{e}(g, \sigma_1) == \hat{e}(g_1, g_2)^{\eta'}\hat{e}(\sigma_2, \vartheta)\hat{e}(pk_1, \Phi)^{\eta'}\hat{e}(pk_2, \Gamma)\hat{e}(\sigma_3, \Omega))$ **then**
    Writes $S_i : = \{PID_{AFAN}, CertRenewal, keytype, T_1, height', M' = Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0), VerificationValues = (M' = Cert(ID_{AFAN}, pk_{keytype}^{new}, T_0), \sigma_{CertRenewal}, ID_{AFAN})), (pk_1, pk_2), (D_1, D_2)\}$ into Outout information Ri, rebuilds and stores the block, broadcasts VerificationSuccess (nonce) and myReport$(block(blockhead, S_i, R_i', nonce')$, round) to the network.
else
    The transaction fails, discards the block, **End Operation**
end if
(3) MN process copy statistics
if (round $> 2/v$) **then**
    Continue the update block onto the blockchain
    End Operation
end if

4.4 Certificate Revocation and Key Recovery

Certificate revocation and key recovery refer to the fact that SN or MN need to revoke original registration information, regenerate new identities and public and

private keys and generate new certificates due to specific security requirements (e.g. users discover that offline public and private key pairs are disclosed at the same time). The execution process is as follows.

Algorithm 4 Certificate Invalidation and Key Recovery

Start Operation
Step 1 SN
(1) From RN randomly accesses an RN node from the RN list, denotes as RN"'
(2) SN generats random number: $Rand() \rightarrow t \in Z_p^*$
(3) SN generates sequence random number and challenge random number: $Rand() \rightarrow seq_3, Rand() \rightarrow chlg'$
(4) SN regenerates temporary anonymity: $\rho' = H_1(ID_{SN}||t), PID'_{SN} = B_I^{\rho'}(modp)$
(5) SN regenerates the original private key signature: $Sig(PID_{SN}||PID'_{SN}), sk_{online}) \rightarrow \sigma_{online}, Sig(PID_{SN}||PID'_{SN}),$
$sk_{offline}) \rightarrow \sigma_{offline}$
(6) SN sends KeyRevokeApplication($PID_{SN}, PID'_{SN}, keyrevoke, T_3, seq_3,$ $pk_1, pk_2, chlg', \sigma_{online}, \sigma_{offline}$) to RN"'.
Step 2 RN" Executes Operation
(1) RN"' verifies the legality of KeyRevokeApplication
if $(Vrfy(pk_{online}, \sigma_{online}, PID_{SN}||PID'_{SN}) == 1\&\&(pk_{offline}, \sigma_{offline},$ $PID_{SN}||PID'_{SN}) == 1)$ **then**
    Accept the KeyRevokeApplication generated by SN
else
    Returns to Step 1
end if
(2) RN"' generats the new partial private key for SN: $\boldsymbol{v} = H_1(PID_{SN}, pk_1) = (v_1, v_2, v_3, \ldots, v_m), D'_{ID_{SN}} = (D'_1, D_2) = (g_2 U'^{\mathcal{K}}, g^{\mathcal{K}})$, there into $U' = \prod_{j=1}^{m} u_j^{v_j}$
(3) RN"' generates validation signatures for SN: $Sig(PID'_{SN}||chlg||T_3,$ $sk_{RN'''}) \rightarrow \sigma'_{RN'''}$
(4) RN"' sends KeyRevokeReply($PID'_{SN}, T_3, seq_3, D'_{ID_{SN}}, \sigma'_{RN'''}$) through the security channel to SN
Step 3 SN Generates Private Key Signatures and Certificates
(1) SN verifies the KeyRevokeReply legitimacy published by RN"'
if $(Vrfy(pk_{RN}, \sigma'_{RN'''}, PID'_{SN}||chlg||T_3, sk_{RN}) == 1)$ **then**
    Accepts the RegistrationApplicationReply generated by RN"'
else
    Returns to Step 2
end if
(2) SN calculates: $\boldsymbol{v} = H_1(PID_{SN}, pk_1) = (v_1, v_2, v_3, \ldots, v_m), \vartheta' = \prod_{j=1}^{m} (\varphi_j)^{v_j}$
(3) SN verifies the partial private key generated by RN"':
if $(\hat{e}(g_1, D'_1) = \hat{e}(g_1, g_2)\hat{e}(D_2, U))$ **then**

Accept partial private key generated by RN"' generates SN's own user's renew private key $sk = \{(\theta_1, \theta_2), (D_1', D_2)\}$
else
Returns to Step 2
end if
(5) SN generates update certificate: $Rand() \to \mathbb{P}, Rand() \to \mathbb{Q}, KGF(1^{\mathbb{P}}) \to (sk_{online}^{update}, pk_{online}^{update}), KGF(1^{\mathbb{Q}}) \to sk_{online}^{update}, pk_{online}^{update})$
securely stores sk_{online}^{update} and sk_{online}^{update}, securely deletes $\mathbb{P}$ and $\mathbb{Q}$
$GenCert(ID_{SN}, pk_{online}^{update}, T_3) : = Cert(ID_{SN}, pk_{online}^{update}, T_3, update)$
$GenCert(ID_{SN}, pk_{online}^{update}, T_3) : = Cert(ID_{SN}, pk_{online}^{update}, T_3, update)$
(6) SN calculates: $M'' = Cert(ID_{SN}, pk_{online}^{update}, T_3, update)||Cert(ID_{SN}, pk_{online}^{update}, T_3, update)$
$\alpha' = H_4(ID_{SN}) = (\alpha_1', \alpha_2', \alpha_3', \ldots, \alpha_l'), \Omega' = \prod_{k=1}^{l} (\boldsymbol{\omega}_k)^{\alpha' k}, \eta'' H_5(M''||g^{\Psi}||pk_1||pk_2), \beta' = H_2(M''||g^{\Psi}||pk_1||pk_2) = (\beta_i'),$
$\Phi' = \prod_{j=1}^{n} (\boldsymbol{\phi}_j)^{\beta_j'}, \boldsymbol{\chi}' = H_5(M''||g^{\Psi}||pk_1||pk_2) = (\boldsymbol{\chi}_i'), \Gamma' = \prod_{j=1}^{n} (\gamma_j)^{\chi_j'},$ the signature is $\sigma_{Update} = (\sigma_1', \sigma_2, \sigma_3) = ((D_1')^{\eta''}(\Phi)^{\theta_1 \eta''}(\Gamma')^{\theta_2}(\Omega')^{\Psi}, D_2^{\eta''}, g^{\Psi})$
(7) SN reconstructs the Input information, sends the Block to the MN: $S_i : = \{PID_{SN}, revoke, online||offline, T_3, height, Cert(ID_{SN}, pk_{online}, T_0), Cert(ID_{SN}, pk_{offline}, T_0))\}$ and $S_i : = \{PID'_{SN}, update, online||offline, T_3, height, Cert(ID_{SN}, pk_{online}^{update}, T_3, update), Cert(ID_{SN}, pk_{online}^{update}, T_3, update), VerificationValues = (Cert(ID_{SN}, pk_{online}^{update}, T_3, update)||Cert(ID_{SN}, pk_{online}^{update}, T_3, update), \sigma_{Update}, ID_{SN})), (pk_1, pk_2), (D_1', D2)\}$
Step 4 MN Competes for Accounting Rights
(1) MN verify the legitimacy of SN identity
(2) MN extracts VerificationValues from S_i, calculates $\vartheta' = \prod_{j=1}^{m} (\varphi_j)^{v_j}, \Omega' = \prod_{k=1}^{l} (\boldsymbol{\omega}_k)^{\alpha_k}, \Phi' = \prod_{j=1}^{n} (\boldsymbol{\phi}_j)^{\beta_j}, \Gamma' = \prod_{j=1}^{n} (\gamma_j)^{\chi_j}$
(3) MN verifies the legality of the signature
if $(\hat{e}(g, \sigma_1') == \hat{e}(g_1, g_2)^{\eta''} \hat{e}(\sigma_2, \vartheta) \hat{e}(pk_1, \Phi)^{\eta''} \hat{e}(pk_2, \Gamma') \hat{e}(\sigma_3, \Omega))$ **then**
MN publish the $S_i : = \{PID_{SN}, revoke, online||offline, T_3, height, Cert(ID_{SN}, pk_{online}, T_0), Cert(ID_{SN}, pk_{offline}, T_0))\}$ and $S_i : = \{PID'_{SN}, Update, online||offline, T_3, height, Cert(ID_{SN}, pk_{online}^{update}, T_3, update), Cert(ID_{SN}, pk_{online}^{update}, T_3, update)$, write $VerificationValues = (Cert(ID_{SN}, pk_{online}^{update}, T_3, update)||Cert(ID_{SN}, pk_{online}^{update}, T_3, update), \sigma_{Update}, ID_{SN})), (pk_1, pk_2), (D_1', D2)\}$ into Ri,reconstruct and store the block, publish $VerificationSuccess(nonce)$ and myReport($block(blockhead, S_i, R_i', nonce')$, round) to the network.
else
The transaction fails, discards the block, **End Operation**
end if

(4) MN process copy statistics
if (round $> 2/v$) **then** Continue the update block onto the blockchain
    End Operation
end if

4.5 Certificate Retrieval and Key Verification

Certificate retrieval and key verification are to find and verify the corresponding keys according to the nodes' identities and key types. SSN reply the latest key information currentpublickey according to the corresponding historical state of the keys. This operation is initiated by MN and SN to SSN to verify the relevant information of the nodes to be retrieved (NTBR), retrieve all transaction records in the blockchain, and output the corresponding keys. If no corresponding key type exists, the output $\perp$ process is as follows.

Algorithm 5 Certificate Retrieval and Key Verification

Start Operation
Step 1 SN
(1) SN randomly accesses an SSN node from the SSN list, denotes as SSN'
(2) SN generates sequence random number and challenge random number: $Rand() \rightarrow seq_4$
(3) SN generates query validation values: $Sig(PID_{SN}||PID'_{NTBR}||(Cert(ID_{NTBR}, pk_{NTBR}^{ketype}, T', keytype)), sk_{online}) \rightarrow \sigma_{lookup}$
(4) SN sends LookupRequest $(PID_{SN}, T_4, height, seq_4, \sigma_{lookup}, PID'_{NTBR})$ to SSN'
Step 2 RN" Executes Operation
SSN' validates LookupRequest, extracts the Si from the Block at height, search and parse Blockbody information
if $(keytype == register \&\& Vrfy(pk_{SN}, \sigma_{lookup}, PID_{SN}||PID'_{NTBR}||(Cert(ID_{NTBR}, pk_{NTBR}^{ketype}, T', keytype)) == 1)$ **then**
    Replies $(currentpublickey = pk^*, Cert(ID_{NTBR}, pk_N^*TBR, T', register))$, **End Operation**
else if $(commandtype == update \&\& Vrfy(pk_{SN}, \sigma_{lookup}, PID_{SN}||PID'_{NTBR}||(Cert(ID_{NTBR}, pk_{NTBR}^{ketype}, T', keytype)) == 1))$ **then**
    Replies $(currentpublickey = pk^{new}, Cert(ID_{NTBR}, pk_{NTBR}^{new}, T', updated))$, **End Operation**
else if $(renewalkeytype == revoke \&\& Vrfy(pk_{SN}, \sigma_{lookup}, PID_{SN}||PID'_{NTBR}||(Cert(ID_{NTBR}, pk_{NTBR}^{ketype}, T', keytype)) == 1))$ **then**
    Replies $(currentpublickey = \perp, Cert(ID_{NTBR}, pk_{NTBR}^{revoked}, T', revoked))$, **End Operation**
else
    Replies $(currentpublickey = \perp, CertNotExisted)$, **End Operation**
end if

5 Protocol analysis

5.1 Security Analysis

Platform Authority Platform authority: the existence of any transaction record in the blockchain is determined by all parties in the process of consensus ledgers and cannot be denied by any node. Therefore, the blockchain results have corresponding authority. Based on the private blockchain system, each node, when performing block calculations, stores the blockchain's historical transaction information like a crypto accumulator in the form of Merkle Tree, and uses SHA-2562 to calculate. As SHA-256 algorithm has unidirectionality and collision resistance [6, 14], it is capable of recording the information of all previous transactions of blockchain nodes in security. With the Merkle Tree and timestamp in the blockchain, the existence of ledger transaction records is proved based on the PoW mechanism. That is, the more trusted nodes in the network and the more nodes authenticating transaction values, the higher the results' reliability [13, 26].

Unforgeability Unforgeability means that, without obtaining the identity of a node and online and offline key pairs, an adversary cannot forge the legal identity of a node and impersonate it to access to a network based on a permissioned blockchain. In the absence of key pairs without node identities, even if the adversary uses a legitimate mobile device to send an authorization request to the RN, the identity information cannot be written into the blockchain as long as he cannot provide the corresponding two sets of public keys and the corresponding signature values. In this case, the adversary cannot access the network.

Key Security The platform in this article verifies the identity of each node and the corresponding public key. The legal correlation between the public key and the node's identity is recorded by the ledger in the blockchain, the results of which cannot be changed. SSN are responsible for verifying the public keys, the generation of which must be subject to the process of identity verification. If the verification fails, the public keys are not verified. Moreover, even if the adversary obtains the corresponding public keys after passing platform verification, he cannot update the keys without having grasped the corresponding two sets of private keys.

Revocation Transparency The revocation process: the emergency revocation of keys in an insecure state is supported by the platform. It can reduce the harms caused by key disclosure. Once a node discovers that the network may be attacked, or there is a vulnerability in the node's mobile application, or the node's offline private key may be compromised, a revocation operation will be triggered. The transparent revocation operation in this paper can be used either as an active defense measure or an emergency remedy measure to improve the security and controllability of the platform as a whole.

The whole revocation process in Dizar architecture is transparent and open. The revocation process in Dizar architecture is not a traditional PKI and is authorized by CA. When a valid revocation transaction is included in a new block, its validity is verified by all miner nodes. Only when all miner nodes have verified the legitimacy of the node can the status of the certificate be changed globally. Therefore, the revocation process in the Dizar architecture is ideally transparent and can be audited and verified. SN do not need CA to check the revocation status of certificates. All miners in Dizar can check and verify the final revocation status of certificates in the block at any time, thus eliminating the problems caused by relying on CA to ensure the reliability and transparency of the results.

Resistance to Aggression The attack methods against permissioned blockchains mainly include Split-world attack, 51% attack, Eclipse attack and DDoS attack. This article copes with DDoS attacks with a permissioned blockchain implemented on a P2P network. The chain itself is a distributed architecture and is featured by decentralization and multiple redundancies. Even if some nodes in the network fail due to attaches, other nodes are not affected and there is no failure of single point. In this paper, the private blockchain architecture is adopted. The node information in the network is stored in a distributed manner. The network system will not be paralyzed when several nodes in the network are attacked. Split-world attacks, 51% attacks and Eclipse attacks are applicable to the situation where the adversary is capable of providing different ledgers to the miner nodes. In a traditional PKI system, such situation only exists when there is a credible log operator and the target nodes do not exchange log views through a gossip protocol. The Dizar architecture has eliminated such attacks because it does not have an independent third-party verification authority. All miners are verification parties, all have the same and unique ledger data, and can only be updated based on the consensus of all miners.

The credibility of Dizar architecture is essentially based on its consensus mechanism. Under the assumption that the basic consensus protocol of blockchain architecture is secure, all miner nodes will have the final state of blockchain data containing certificate status. However, based on the PoW consensus mechanism, it is impossible to attack the blockchain node data. Therefore, the Dizar architecture has strong resistance to attack.

Verification of Information Rationality In fact, we take the authentication process of the identity registration as an example to prove the mathematical rationality of the node authentication process.

Proof.

$$\begin{aligned}\hat{e}(g,\sigma_1) &= \hat{e}(g, D_1^{\eta}(\Phi)^{\theta_1\eta}\Gamma^{\theta_2}\Omega^{\Psi}) \\ &= \hat{e}(g,(g_2^{\lambda}\vartheta^{\mathcal{K}})^{\eta})\hat{e}(g,(\Phi)^{\eta\theta_1})\hat{e}(g,(\Gamma)^{\theta_2})\hat{e}(g,(\Omega)^{\Psi}) \\ &= \hat{e}(g^{\lambda},g_2)\hat{e}(g^{\eta\mathcal{K}},\vartheta)\hat{e}(g^{\theta_1},\Phi)^{\eta}\hat{e}(g^{\theta_2},\Gamma)\hat{e}(g^{\Psi},\Omega) \\ &= \hat{e}(g_1,g_2)^{\eta}\hat{e}(\sigma_2,\vartheta)\hat{e}(pk_1,\Phi)^{\eta}\hat{e}(pk_2,\Gamma)\hat{e}(\sigma_3,\Omega)\end{aligned}$$

Proof is complete.

Therefore, based on the rationality of the design verification algorithm and combined with the non-certificate authentication process, the global credibility authentication of the information submitted by members can be realized without the need for a third-party authentication institution and without exposing the member's relevant privacy information. The Dizar architecture has eliminated the requirement for external auditors to check their encryption consistency and behaviors, as new blocks are verified and attached to the blockchain only if all miners agree.

Decentralized Storage The Dizar architecture has eliminated the storage problem of certificate/key centralization. We believe it is a burden for X509 V3 to manage credible CA root certificates and public keys in a centralized manner, which is also one of the main sources of security threats [20, 24]: it stores CA root certificates and public keys in a centralized certificate database. However, there is no need for a separate certificate database to manage the certificates and public keys of credible CA under the Dizar architecture, which stores and maintains the certificates and public keys in distributed databases of blockchains composed of all nodes. The distributed chain storage structure of a permissioned blockchain ensures the credibility and integrity of these certificates and keys.

5.2 Platform Efficiency Test

In this section, the cost of competition for bookkeeping rights by multi-node is tested in an experiment. To not lose generality, the running of nodes is at the same level of complexity as other literature. The efficiency test is conducted on ARM FastModels(ARM,2011) in Ubuntu 10.04 environment. The hardware is Intel(R) Core(TM)i7-4510CPU2.60g Hz, 16G memory. The efficiency test uses Docker virtual technology to build private blockchain nodes and uses Zeppelin to design the intelligent contract architecture of the blockchain. Each blockchain node runs in a separate Docker container with 2G memory. The contract codes to be run by each node are packaged in the Docker container, and the programs are isolated from each other with high security. The resource consumptions of the Docker containers are also independent of each other. The CPU consumption is at most about 3% and the memory consumption is not more than 100MB to test the average running efficiency of each node. The number of protocol nodes in this article is 1024, the block size is 1MB, and the ledger records are single records. Table2 shows the efficiency comparison, in which public and private key

encryption and decryption, digital signature and verification are compared, and the sum of single steps is counted.

Table 2. Comparison of Efficiency of Various Protocols

Protocols	Average Number of Public and Private Key Encryption /times	Average Digital Signature and Verification /times	Average Hash Calculation times /times	Consensus Mechanism	Degree of decentralization	Average Registration time /s
Literature [4]	5	6	4	PoW	Distributed	65
Literature [11]	8	6	4	PoS	Decentralized	58
Literature [16]	6	4	2	PoS	Distributed	64
Literature [17]	3	5	2	Multipoint Response	Decentralized	47
Dizar	4	4	3	Improved PoW	Distributed	10

Calculation Cost Analysis: Compared with [11] and [4], the public key encryption and decryption, signature verification and hash calculation of the protocol in this article have decreased, but some indexes have increased slightly compared with [17, 16]. [11] and [17] are based on the multi-center certificate distribution protocol, in which the computing loads of the center nodes will increase with the increase of nodes. This article is based on the distributed private chain. The increase of proxy nodes will not lead to an increase in the average number of public key algorithms used in multi-party competition for bookkeeping rights, and there are multiple proxy nodes, which can effectively reduce the burden of a single authentication node. In addition, the protocol in this article actively screens the ledgers to be authenticated in the process of multi-node competing for bookkeeping rights, thus reducing unnecessary expenses. Tests with mining machines with equivalent computing capacity show that the average registration time is obviously shortened. Therefore, the protocol in this article is more advantageous in writing data into blockchain in a multi-node mobile environment, thus maintaining a moderate calculation overhead on the network. Compared with PoS and multi-point collaboration, this paper adopts the POW consensus approach to minimize the overall mechanism unfairness caused by the fact that a few nodes have greater decision-making powers, making the process of competing for bookkeeping rights more reliable.

Average transaction time: the time required to detect that a block containing an input collection passes verification and is added into the blockchain (through

Algorithm 1). The running time of PKI based on blockchain and different working mechanisms is analyzed. The transaction verification time is the same order of magnitude for all transaction verification times. However, as shown in Table 2, the average transaction verification time of CertCoin protocol [11] is about 1.1 minutes per transaction, the one of IKP protocol [17] is 1 minute, while the one of PB-PKI is 45s [4]. The verification transaction time of the blockchain will change according to the status of the network and the upgrade of the protocol version. In the running of the ledger-based private blockchain in this article, the average block verification takes about 10 s. Compared with other schemes, the average running time is shortened and relatively reasonable, which helps to avoid potential false transactions from taking place in a short time and is more suitable for running in a dynamic network.

Increase in the size of the blockchain: the blockchain, acting as a distributed database, provides integrity and store security information to realize credibility. For the calculation, we adopt the elliptic curve encryption algorithm with smaller key size, higher performance and higher security. The block size is 256 bits. The size of Dizar's permissioned blockchain depends on the block size and the frequency of adding new blocks. The block's size increases with increase of transactions, and the block verification time depends on the consensus algorithm selected. In actual calculation, the target block time is selected to be 5 minutes considering the following aspects.

- The storage capacity required in a light node will increase as the block time decreases. However, most of the medium and small-sized nodes in the actual Dizar architecture have storage limitations and cannot be easily upgraded. With the increase of network mobility, we expect light nodes to become the main form of the most common Dizar architecture. Light nodes do not store blocks but only headers. The required disk space is proportional to the number of blocks. The light nodes in the Dizar architecture require only about 30 MB of disk space per year for storage.
- Full Node: The node where the whole miner nodes create new blocks. It is a part of the consensus and stores all blocks. Assume that the average certificate life cycle is one year, all network nodes need only 150 GB of disk space to store all updated transactions (the assumption is based on the 2017 VeriSign Domain Name Industry Briefing [25], or, about 3.3×108 registered domain names). These calculations are based on the assumption that certificates are stored for all. Considering the fact that the price of 1 GB disk storage is about US $ 0.02 [3], the cost of storing all Dizar block data for non-mining entire nodes is about US $ 5. The extra data needed to build a blockchain will not have significant impact on the overall storage space required.

Analysis of time consumption required by blockchain retrieval: the main challenge in PKI services in the protocol of this article is the time needed

to perform the retrieval in the blockchain. The complexity of this operation is theoretically $O(\ )$, v is the number of network nodes. The CertCoin [11] and the IKP [17] use a DHT and a password accumulator to reduce the retrieval complexity and time of the blockchain to $O(\log(t))$, where t represents the number of transactions in the blockchain. Therefore, proceeding from this point of view, the protocol in this article still has opportunity for improvement theoretically in terms of data retrieval. On the other hand, the actual time required to view the retrieval is as shown in Table 3.

Table 3. Average Operation Time of Retrievals in Blockchains of Different Sizes

	LookUp
250 MB	320 ms
500 MB	652 ms
1 G	1315 ms
2 G	2711 ms

Based on the different sizes of blockchains, we examine the time required by each transaction in all the blockchains. The worst case of the test certificate retrieval is that the transaction block is retrieved at the end of the blockchain. Our results in Table 3 show a linear increase in complexity. For the size of the representative blockchain, the certificate retrieval time is less than 3 seconds. However, in the actual applications, the data service nodes do not have to retrieve throughout the entire blockchain. The data service nodes usually use the latest updated information in the blockchain to improve the retrieval efficiency by adopting a fast retrieval method.

6 Conclusion

To address the existing challenges in current blockchain-based PKI, this article proposes the Dizar architecture, designs the blockchain running mode in a basic ledger system, realizes multi-party credibility of identity information based on the non-certificate authentication process, and defines the corresponding distributed PKI service process. The platform realizes distributed PKI results based on blockchain, which are featured by multi-party credibility and joint maintenance. The analysis of the simulation test results shows that its calculation cost makes it suitable for implementation in the distributed network. The platform has strong scalability and high network adaptability. Our next step is to study the fast synthesis and retrieval of block data to improve the adaptability of a distributed PKI in the network.

References

1. Al-Bassam, M.: Scpki: A smart contract-based pki and identity system. In: ACM Workshop on Blockchain, Cryptocurrencies and Contracts. pp. 35–40 (2017)
2. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y.: Hyperledger fabric: A distributed operating system for permissioned blockchains (2018)
3. Author: Disk storage cost. https://diskprices.com/, accessed November 28, 2017
4. Axon, L.: Privacy-awareness in blockchain-based pki (2015)
5. Camacho, P., Opazo, R., Opazo, R., Opazo, R.: Strong accumulators from collision-resistant hashing
6. Camacho, P., Opazo, R., Opazo, R., Opazo, R.: Strong accumulators from collision-resistant hashing. In: International Conference on Information Security. pp. 471–486 (2008)
7. Diginotar: Diginotar. https://en.wikipedia.org/wiki/DigiNotar/, accessed March 4, 2011
8. Eweek: Mozilla asked to revoke trustwave ca for allowing ssl eavesdropping. http://www.eweek.com/security/mozilla-askedto-revoke-trustwave-ca-for-allowing-ssleavesdropping/, february March 4, 2012
9. Faisca, J.G., Rogado, J.Q.: Personal cloud interoperability. In: World of Wireless, Mobile and Multimedia Networks. pp. 1–3 (2016)
10. Falliere, N., Murchu, L.O., Chien, E.: W32.stuxnet dossier. White Paper (2011)
11. Fromknecht C, V.D.: Certcoin: A namecoin based decentralized authentication system. technical report, 6.857 class (2014)
12. Gervais, A., Karame, G.O., Glykantzis, V., Ritzdorf, H., Capkun, S.: On the security and performance of proof of work blockchains. In: ACM Sigsac Conference on Computer and Communications Security. pp. 3–16 (2016)
13. Gervais, A., Karame, G.O., Glykantzis, V., Ritzdorf, H., Capkun, S.: On the security and performance of proof of work blockchains. In: ACM Sigsac Conference on Computer and Communications Security. pp. 3–16 (2016)
14. Heilman, E., Kendler, A., Zohar, A., Goldberg, S.: Eclipse attacks on bitcoin's peer-to-peer network. In: Usenix Conference on Security Symposium. pp. 129–144 (2015)
15. Lesueur, F., Me, L., Tong, V.V.T.: An efficient distributed pki for structured p2p networks. In: IEEE Ninth International Conference on Peer-To-Peer Computing. pp. 1–10 (2009)
16. Lewison K, C.F.: Backing rich credentials with a blockchain pki. technical report, pomian & corella, llc, 2016. (2016)
17. Matsumoto, S., Reischuk, R.M.: Ikp: Turning a pki around with decentralized automated incentives. In: Security and Privacy. pp. 410–426 (2017)
18. Melin, T., Vidhall, T.: Namecoin as authentication for public-key cryptography (2014)
19. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Consulted (2008)
20. Patrick Wardle, A.M.: Ca threats. https://objectivesee.com/blog/blog_0x26.html/, accessed April 4, 2017
21. Phillip: Comodo ssl affiliate the recent ra compromise. https://blog.comodo.com/other/therecent-ra-compromise/, accessed March 4, 2011
22. Satoshi: Bitcoin blockchain size. http://blockchain.info/charts/blocks-size/, accessed August 7, 2018

23. Shen, X., Pei, Q.Q., Liu, X.F.: Survey of block chain. Chinese Journal of Network & Information Security (2016)
24. Symantec, T.I.: Marketscore proxyserver certificate. https://www.symantec.com/security_response/ attacksignatures/detail.jsp?asid=20804./, accessed April 4, 2017
25. VerisignDECEMBER: The verisign domain name industry brief. https://www.verisign.com/en_US/domainnames/dnib/index.xhtml/, accessed April 4, 2017
26. Xu, J.J.: Are blockchains immune to all malicious attacks? Financial Innovation **2**(1), 25 (2016)
27. Zhicheng, Z., Lixin, L., Zuohui, L.: Efficient cross domain authentication scheme based on blockchain technology. Journal of Computer Applications **38**(2), 316–320 (2018)