Mechanism Design of Privacy Data Storage and Transaction with a Token-based Bookkeeping Method

Liangshun Wu[1[0000-0001-6183-4680]] and H. J. Cai[2[0000-0002-3477-8334]]

[1,2] School of Computer Science, Wuhan University, Wuhan, Hubei 430079 P. R. China
1471007431@qq.com

Abstract. The blockchain system with a token-based bookkeeping method has a universal token structure, a flexible authority management mechanism and a DPOS+BFT consensus mechanism. When transaction done, token's ownership will be updated. A traceable and non-tamperable privacy data block chain could pinned to a token. Each block consists of the ciphertext of privacy data, the timestamp, and the hash value of the previous block. Latest block will be append periodically. If someone is willing to obtain privacy data and its historical versions, an improved interactive zero-knowledge proof method based on discrete logarithm can be used to verify that the token owner does have the key to recover corresponding plaintext. When a pinned token transaction finished, the owner of the Token would be updated, the key to the privacy data handed over to the new owner, and the rewards denominated in fuel coins payed to the previous owner. Our mechanism design has several advantages: delivery versus payment, preventing verifier nodes from doing evil, shielding previous users by " changing lock " , traceable, and naturally supports parallelism. In addition, we also designed the reward program and deceit punishment mechanism.

Keywords: Token, Privacy Data, Block Chain, Zero knowledge proof, Transaction.

1 Introduction

Token is a derivative of the blockchain technology. It originated from Ethereum ERC20 standard [1]. Based on this standard, everyone is allowed to issue a token on the blockchain platform. Distinguished from the cryptocurrency on Bitcoin, token is a kind of proof of equity, encrypted and tradable. Token can also be issued in a centralized system, not specific to the blockchain, but the token circulated on the blockchain can guarantee uniqueness through encryption and the distributed ledger. A lot of token-based block chain systems and technologies have emerged afterwards, such as everiToken, EOS, etc. Distinguished from Bitcoin's UTXO model and Ethereum's account-based model, token-based bookkeeping method simply change the owner of the token to complete the transaction [2], and it is adopted by everiToken. We will improve this design to better accommodate privacy data's storage and transaction.

So far, neither the transaction of Bitcoin nor everyToken's is anonymous. The content of transaction(including transaction amount and transaction sides) can be queried and tracked. There are some blockchain systems adopted anonymous technology, such as Monroe, DASH, Zcash, etc. Anonymous transactions have the advantages of privacy protection, freedom, etc, but will inevitably bring about crimes such as money laundering, and is difficult to regulate effectively. Some countries' regulatory agencies have banned anonymous transactions of cryptocurrencies [3]. In view of this, non-anonymous way will be employed in our design to avoid regulatory risks on the one hand, and on the other hand can facilitate the implementation of the system reward program and the accumulation of user's credit.

Regarding the storage, the mainstream technology is centralized storage, such as data center, cloud storage,etc. Wang Zi (2017) [4] studied how to realize the security of cloud personal privacy data. HAO Kun, et al (2017) [5] designed a Decentralized Metadata Blockchain (DMB), save metadata in blocks and ensure the integrity by collaborative verification. DMB has traceability and integrity, and good concurrent processing capability.

Privacy data also requires liquidity, which is achieved by transaction. Before a transaction, the potential counterparty must clarify that the token owner does have the key to the privacy data, but the token holder won't tell the key. That is zero-knowledge proof (abbr as ZKP). ZKP methods are mainly divided into two categories, one is interactive, such as scheme based on discrete logarithm, scheme for deciphering RSA capability [6], etc. The other one is non-interactive,such as zk-SNARK [7] adopted by Zcash, and GSNIZK scheme proposed by Chen Chen[8], etc. In consideration of the tremendous amount of computing for non-interactive schemes, we employed an interactive scheme based on discrete logarithm and improve it.

2 Privacy Data Storage

2.1 Blockchain System with a Token-based Bookkeeping Method

Tianqi Cai, H.J.Cai, et al [2] defined a blockchain system with a toke-based bookkeeping method. Tokens share a universal data structure, which consists of three fields: domain, token name and owner.

In this paper, the structure is improved, and the "note" field is added. The improved universal token structure is shown in Fig.1.

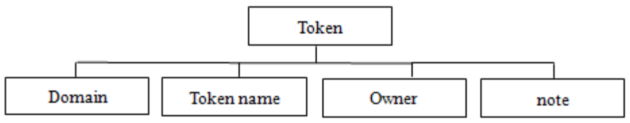

Fig. 1. Improved token structure

According to reference [2], the domain of each token defines the corresponding authorization management. Whenever performs an operation, the legality of the operation must be verified. All operations fall into three categories: issue, transfer, and manage.

We believe that the content of the manage includes not only setting the authorization, but also modifying the token (especially the "note" field).

When an operation is initiated, the group member associated with the operation must sign with the private key, and the verifier nodes authenticate with the public key. If the sum of the weights of the users who agree and sign on the operation reaches (i.e., is equal to or greater than) the threshold required for the certain token, then the operation is approved, and otherwise the user receives a rejection.

The system adopts the DPOS+BFT consensus mechanism. DPOS has been proven to be able to meet the needs of blockchain applications. In [2], 15 producers elected by voting are responsible for bookkeeping/produce, once more than 11 producers confirmed the operation, then the operation is verified and irreversible.

The blockchain system with a toke-based bookkeeping method is proved secure, with high TPS, naturally support parallelism and regulatory-friendly.

2.2 Privacy Data Block Chain Pinned to Token

In cryptocurrency applications such as Bitcoin, the blockchain holds the user's transaction data, and in smart contract applications such as Ethereum, the blockchain preserves the user's smart contract. If users have privacy data reluctantly revealed to others, they can save data on blockchain too.

Assuming that Alice has some sensitive privacy data (it could be the identity and purchase information of all customers of a foreign trade corporation, or the formula of Coca-Cola, etc.), mark the privacy data as *pdata*, which is encrypted to $E_M(pdata)$ by symmetric key.

If the privacy data is modified, Alice will combine the timestamp, the encrypted privacy data, and the hash value of the previous block, and then hash to form a new block. The PreHash value field of the initial block is marked as 0. So, we store the history versions through the blockchain. We call this chain an privacy data blockchain, abbreviated as PDBC.

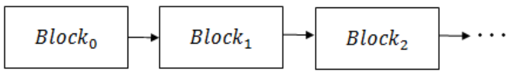

Fig. 2. PDBC structure

Where,

$$block_i = (TimeStamp, E_M(pdata_i), H_i)$$
$$H_i = HASH\,(TimeStamp, E_M(pdata_i), H_{i-1})$$
$$H_0 = HASH\,(TimeStamp, E_M(pdata_0), 0)$$

This block chain structure records the modification history of the privacy data, and if any of the privacy data in history is tampered with, then the values of all subsequent blocks will be unverifiable.

2.3 Upload Token and Periodically Data Block Append

Still the case mentioned above. Alice can create a token. Domain name is defined as: ABC Company, token name: customer information, owner: U_{Alice} , description: all customer information. To facilitate DLZK verification (as detailed in Section 3) and key correctness verification, Alice must publish the discrete logarithmic parameters $\left(Y_{E_M}, g_{E_M a}, n_{E_M}\right)$ and the hash value of E_M. Authorization for all operations is set to owner group.If the domain of ABC Company already exists, the system automatically matches the operation authorization. The token data can be stored in Json format as,

```
{
    "Domain": {
                "name":ABC Company,
                "Groups":{ "OwnerGroup": $U_{Alice}$},
                "authority":{
                        "Issue":{
                            "Threshold":1,
                          "Weight":{"OwnerGroup":1}
                            },
                        "Transfer":{
                            "Threshold":1,
                          "Weight":{"OwnerGroup":1}
                            },
                        "Manage":{
                            "Threshold":1,
                          "Weight":{"OwnerGroup":1}
                            }
                        }
          },
    "Token name": customer information,
    "Owner": $U_{Alice}$,
    "note": {
            "Description": customer information,
            "PDBC":PDBC,
            "DLZK": $\left(Y_{E_M}, g_{E_M}, n_{E_M}\right)$,
            "KeyHash":HASH($E_M$)
        }
}
```

Since the modification of privacy data can be frequent, or even real-time, there is no need to record every minor change of the data, as this can lead to numerous duplicate copies of metadata, taking up too much resources. We can periodically upload the version instead. When the fixed time interval arrives, we generate a new block using snapshot of current privacy data, and append it to the existing PDBC.

HAO Kun et al. [5] proposed a blockchain-based metadata distributed storage model: DMB. In this paper, the storage structure is distinguished from the DMB model. We encrypt the data and abandon the Merkel tree structure. By means of periodic uploading, redundancy is reduced, efficiency is improved. All historical versions can be traced, whereas only the hash value of the metadata Merkel root node exists in DMB, which can not be traced back to every historical metadata. However, in the same way as DMB , data synchronization is achieved in a coordinated manner.

2.4 PDBC Kit

In order to ensure the storage mechanism, the system generates and updates PDBC with a unified program, we call it PDBC Kit. The PDBC Kit is packaged in the form of dll/lib and is provided to the user with API.

The user only needs to input the key and upload the plaintext of the privacy data. The PDBC Kit automatically calculates the ciphertext of the private data according to the preset symmetric key algorithm (e.g. 3DES,IDEA,FEAL,BLOWFISH, etc.), records the timestamp, and adds the hash value of the previous block. The ciphertext, timestamp, and PreHash are hashed to form a new block. After the new block is generated, the PDBC Kit automatically sends a message to the verifier nodes, and the verifier nodes confirm the operation and complete the bookkeeping. The entire network nodes will proceed data synchronization with the user node.

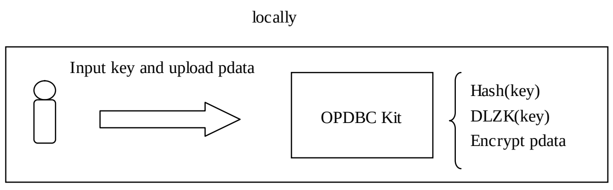

Fig. 3. PDBC kit

In order to ensure that the user always encrypts the data with the same key, the PDBC Kit performs a hash operation on the initial key, which is stored in the "KeyHash" field of the token. Each time the block is appended, the PDBC Kit compares the key hash values. If the key is changed, the system surely reject this operation. This restriction will not be released until the ownership of the token is transferred.

In addition, the PDBC Kit is responsible for calculating the DLZK parameters of the key when the key is first entered and when the key is changed, and the result is stored in the "DLZK" field of the token.

To facilitate understanding, Fig. 4 shows the block chain system described in this Section. Tokens of the same domain forms a branched chain, and the branched chain is part of the main chain.

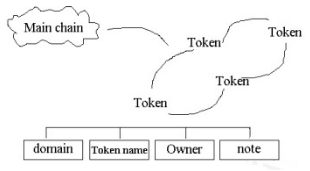

Fig. 4. Blockchain system based on token

3 Privacy Data Transaction

3.1 Potential Transaction and Key Verification by Improved DLZK method

Users can query token information through the a common system search tool. If a user Bob browses to Alice's token and is interested in the privacy data pinned, he will get them through transaction. To ensure a valid transaction, Bob must first confirm that Alice has the key E_M to recover corresponding plaintext of the privacy data. That is, Alice must convince Bob that she does have the key without disclosing the key.

This is a zero-knowledge proof (abbr as ZKP) question. There are many ways to achieve it. In reference [6], the "classic" scheme of ZKP based on discrete logarithm is described. The scheme combined with this example can be expressed as:

Converts the key E_M to a integer x (e.g. by ASCII code, etc.). Let $Y = g^x (mod\, n)$, where n is a prime and $(x, n-1) = 1$. Here, Y, g, n are public, x is secret. Discrete logarithms usually do not have very efficient ways for fast calculation, so it is impossible to get x quickly through Y, g, n.

The verification process is as follows.

Step 1: The verifier Bob generates a random number k, calculates $K = g^k (mod\, n)$, and sends it to Alice.

Step 2: The prover Alice calculates: $M = g^{kx} (mod\ n) = K^x (mod\ n)$, sent to Bob.

Step 3: The verifier Bob check $Y^k (mod\ n) = M$? If they are equal, it proves that Alice does know x.

Proof 1. $g^{kx}(mod\ n) = [g^k(mod\ n)]^x(mod\ n) = K^x(mod\ n)$
$g^{kx}(mod\ n) = [g^x(mod\ n)]^k(mod\ n) = Y^k(mod\ n)$

However, the verifier Bob will eventually get $K^x\ (mod\ n)$ regardless of which K is selected. This is what he should not get,since he may pretend that he knows x.

Bruce Schneier [6] also provides a solution by multiple interactions. And the more interactions, the stronger the proof. Obviously,it is expensive. We improve it as follows.

Step 1: Alice generates a random number j, calculate $(i \times j) = x(mod\, n-1)$ to get i. If i, j are both greater than 3, continue, otherwise re-select. Then calculate $Y_1 = g^i(mod\, n)$ and send Y_1 to Bob.

Step 2: Bob generates a random number r, calculates $Y_2 = g^r\, (mod\ n)$, and sends Y_2 to Alice.

Step 3: Alice calculates $Y_3 = (Y_2)^i(mod\ n)$ and sends Y_3 to Bob.

Step 4: Bob calculates $Y_4 = (Y_1)^r(mod\ n)$. If $Y_3 = Y_4$, it is believed that Y_1 sent by Alice is indeed $g^i\,(mod\ n)$;

Step 5: Bob generates another random number s, calculates $Y_5 = (Y_1)^s(mod\ n)$, and sends Y_5 to Alice;

Step 6: Alice calculates $Y_6 = (Y_5)^j\,(mod\ n)$ and sends Y_6 to Bob;

Step 7: Bob calculates $Y_7 = (Y)^s(mod\ n)$. If $Y_6 = Y_7$, Bob believes that Alice does have the privacy data key x.

This scheme masks x by i, j, which j is a random number, i is generated by j, so i is also a random number. Bob first verifies that Y_1 sent by Alice is indeed $g^i\,(mod\ n)$, then verifies that Alice knows j. Since $(i \times j) = x(mod\ n - 1)$. Therefore, Bob convince that Alice knows x. Far less interactions than Bruce Schneier's solution.

Fig. 5. Depict this process.

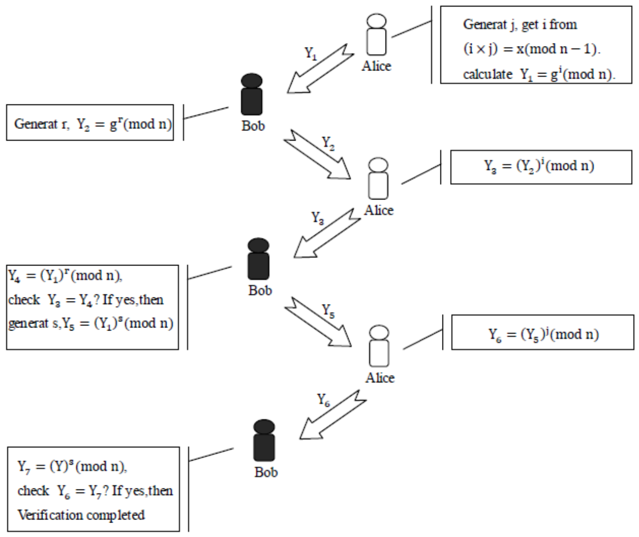

Fig. 5. Improved DLZK process

To facilitate marking, we call this improved ZKP method based on discrete logarithms as: DLZK.

The proof of Step 4 is as follows.

Proof 2. $Y_4 = (Y_1)^r(mod\ n) = (g^i(mod\ n))^r(mod\ n) = g^{i\times r}(mod\ n) = (Y_2)^i(mod\ n) = Y_3$

The proof of Step 7 is as follows.

Proof 3.

$$Y_6 = (Y_5)^j(mod\ n) = ((Y_1)^s(mod\ n))^j(mod\ n) = \left(\left(g^i(mod\ n)\right)^s(mod\ n)\right)^j(mod\ n)$$
$$= \left(g^{i\times s}(mod\ n)\right)^j(mod\ n) = g^{i\times j\times s}(mod\ n) = (Y)^s(mod\ n) = Y_7$$

Without doubt, other ZKP schemes are also possible, the method selected here is just one of the feasible schemes.

3.2 Transaction Process

For token without privacy data pinned, transaction means the change of ownership. However, for token with privacy data pinned, transaction means not only to change ownership but also to hand over the key of privacy data to new owner. In general, new owner are required to pay a certain amount of fuel coins as reward.

Step 1: Alice signs with her own private key SK, and then generates an asymmetric key pair (SK_{EM}, PK_{EM}), encrypts E_M with SK_{EM}.Then create a message as: Message =

["Sign": $SK(U_{Alice})$, "Operation Type": Transfer, "Content": $SK_{E_M}(E_M)$], and calculate message digest D=MD5(Message).

Step 2: Alice and Bob establish a channel, Alice sends PK_{E_M} and D to Bob.

Step 3: Alice sends Message to verifier nodes and change the ownership as Bob.

Step 4: Bob sends the pre-agreed fuel coins reward to verifier nodes for trusteeship.

Step 5: Verifier nodes authenticates Alice's identity with PK and check whether sum weights reach the threshold.

Step 6: If operation permitted, verifier nodes will forward Message to Bob, pay fuel reward to Alice, and finish bookkeeping.

Step 7: Bob uses D to verify the integrity of the Message. And E_M is then obtained by $E_M = PK_{E_M}(SK_{E_M}(E_M))$. Bob creates a new private data key E_M' (change lock).

Fig. 6. shows the transaction process.

3.3 Advantages

This transaction process has the following obvious advantages:

a. Delivery versus payment

The deliver of key encrypting privacy data and the transfer of token's ownership are simultaneous.

b. Prevent verifier nodes from doing evil

On the one hand, the verifier nodes cannot extract the key from the message; on the other hand, the receiver can verify the message integrity that fabrication is prevented.

c. Shielding previous users by " changing lock "

Even if the original owner retains the key, he cannot decrypt the latest privacy data because of the " changing lock " mechanism.

d. Traceable

The revision history of all privacy data cannot be tampered with.

e. Naturally supports parallelism, easy to cross-chain

The transfer of each token is independent. On multi-core CPUs, this greatly improves the transaction performance. In other words, the token-based design naturally supports parallelism and cross-chain transactions, and does not require additional introduction of third-party logic storage or other structures to assist in concurrent or cross-chain operations.

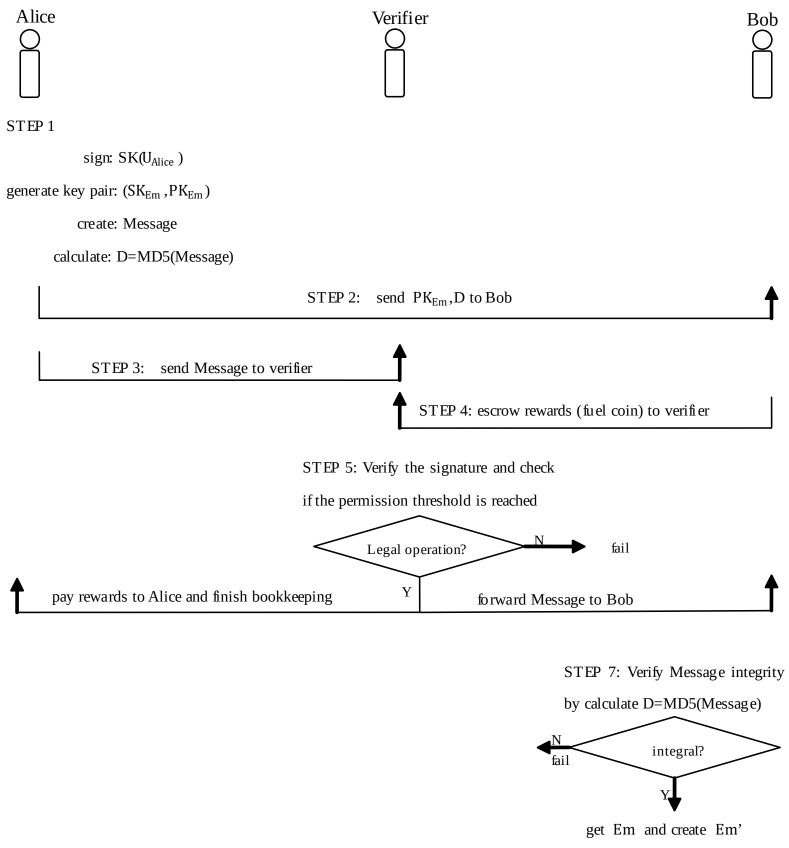
Alice
Verifier
Bob
STEP 1
sign: SK(U_Alice)
generate key pair: (SK_Em, PK_Em)
create: Message
calculate: D=MD5(Message)
STEP 2: send PK_Em, D to Bob
STEP 3: send Message to verifier
STEP 4: escrow rewards (fuel coin) to verifier
STEP 5: Verify the signature and check
if the permission threshold is reached
Legal operation?
N
fail
Y
pay rewards to Alice and finish bookkeeping
forward Message to Bob
STEP 7: Verify Message integrity
by calculate D=MD5(Message)
integral?
N
fail
Y
get Em and create Em'

Fig. 6. Transaction process

4 Reward and Punishment Mechanism

4.1 Reward Program

Fee: to avoid malicious attacks, such as DDos, and reward the resources provided by the block producer at the same time. Any operation consumes a certain amount of fuel as a service fee and as a reward to the block producer.

Decentralization as mining: to reward users who contribute to the ecosystem. The more tokens issued pinned to privacy data, increased real transactions, and increased participants, the more contribution to the ecosystem was believed, and the more rewards will be given. Thus, the rewarded users can earn more credits and gradually grow into an underlying verifier node in the future. The system fuel coins are issued annually in addition to the rewards for verifiers and dividends and are mainly used to support weak centers or users , and that is so called " Decentralization as Mining".

4.2 Deceit Punishment

After a token transaction, if the recipient finds that the key he received is wrong, the new owner can sued. The verifier nodes/producers are responsible for processing the lawsuit, they will get the key by $E_M = \mathrm{PK}_{E_M}(\mathrm{SK}_{E_M}(E_M))$, and simply hash E_M to check if the key is correct. If confirm that the sender sent the wrong key, the sender will be fined 2 times of the reward to receiver.

This operation consumes a certain amount of fuel coins and is paid to the producer node. This fee-based service will effectively prevent attackers from consuming system resources.

5 Probable Inadequacies

The design may be not perfect.

On the one hand, if the privacy data is large enough, and with tremendous nodes in this branch, then the append operation will take too much time in data synchronization. And, as the version continues to update, since the historical versions cannot be tampered with, the data will continue to accumulate and the storage space used will increase dramatically.

On the other hand, our design fully addresses the privacy security issue, but due to the excessive interaction and computational complexity, the transaction speed is much slower than token transaction without pinned privacy data.

In addition, the anonymity of the transaction is also a technical direction to break through.

In any case, this paper only proposes a feasible mechanism design, which has its own unique advantages, but it is still worth studying and perfecting.

References

1. ”What is ERC-20 and What Does it Mean for Ethereum?”, https://www.investopedia.com/news/what-erc20-and-what-does-it-mean-ethereum/, last accessed 2018/10/6.
2. Tianqi Cai, H. J. Cai, Hao Wang, Xiji Cheng and Linfeng Wang: Analysis of a Blockchain System with a Token-based Bookkeeping Method. In: IEEE ACCESS(2018). (in press).
3. JAPAN Virtual Currency Exchange Association prohibits cryptocurrency "anonymous tran sactions". https://www.walian.cn/news/3131.html. last accessed 2018/10/6.
4. Wang Zi: Privacy Preserving Cloud Storage. Cyberspace Security 6(8), 13-15 (2017).
5. HAO Kun, XIN Junchang, HUANG Da: Decentralized model for distributed storage syste m. Computer Engineering and Applications 53(24), 1-7(2017).
6. Bruce Schneier: Applied cryptography. 2nd edn. China Machine Press, China(2000).
7. Eli Ben-Sasson, Alessandro Chiesay, Christina Garmanz, Matthew Greenz, Ian Miersz, Er an Tromerx, Madars Virza Zerocash: Decentralized Anonymous Payments from Bitcoin. I n: 35th IEEE Symposium on Security and Privacy, pp. 459-474.Computer Society, THE F AIRMONT, SAN JOSE, CA.
8. ChenChen: Methods of Structure-Preserving Signature Based on Non-interactive Zero-kno w ledge Proof. Northeastern University(2012).