A Multi-Chain Model for CBDC

Wei-Tek Tsai [1, 2,3,4,5], Zihao Zhao[1], Chi Zhang[1], Lian Yu[2, 3, 5, 6], Enyan Deng[2]

[1] Digital Society & Blockchain Laboratory, Beihang University, Beijing, P. R. China
2 Beijing Tiande Technologies, Beijing, China.
3 IOB Laboratory, National Big Data Comprehensive Experimental Area, Guizhou, China
4 Arizona State University, Tempe, AZ 85287, USA
5 School of Software and Microelectronics,Peking University, Beijing, China
6 Andrew International Sandbox Institute, Qingdao, China

Abstract- Recently digital currency has received significant attention, and among many topics within digital currency, CBDC (Central Bank issued Digital Currency) has been a hot topic as it will affect national currency systems. RSCoin is the first CBDC model sponsored by Bank of England, and it uses the UTXO model from Bitcoin. This paper proposes a new CBDC model Panda model that can store account balance like current banking systems, but use efficient consensus protocols to ensure that relevant parties have consistent views of transactions and accounts. The Panda model is scalable as it can include as many financial institutions or individuals as possible in the system with increasing or decreasing workloads. The model has been simulated in the TaiShan Sandbox in Qingdao**.**

Keywords—Blockchain; CBDC; Internet-Blockchain;

I. INTRODUCTION

Blockchain (BC) is a type of immutable, traceable, distributed ledger technology (DLT). BC has garnered a significant attention recently [9]. Data in the BC is stored in blocks, and blocks form a chain using a cryptographic hash of the previous block. Every block contains a timestamp and transaction data. BC uses a replication based distributed system, and all nodes participate in transaction voting, verification, and storage. BC ensures the immutability, traceability, and allows the application level to be transparent. It is suitable for constructing large scale, complication data storage systems that must keep a long history of records, and is resistant against both internal and external security threats [10].

An important application of BCs is CBDC (Central Bank issued Digital Currency). Bank of England first proposed the CBDC model in 2015 [1] [3] [4], and expressed strong interest to deploy CBDC in UK. According to the Bank, the following are primary reasons for the creation of CBDC:

1. It can make trades efficiently, reduces the costs and increases transparency;
2. As it carries the reputation of central banks (such as Bank of England), It instills confidence in the market, and will be an effective way for central banks to regulate the financial markets;
3. As third-party payment becomes increasingly important, the financial markets are moving toward new payment systems. These new platforms may not be under the direct regulation of the central bank, and CBDC can be used as a means to regulate them;
4. Traditionally, payment between banks are guaranteed by Bank of England, Using CBDC, the central bank could extend the same type of assurance for these new forms of financial platforms

RScoin is the first CBDC model proposed and sponsored by Bank of England [1]. As the first CBDC model, RScoin has received significant attention in the CBDC community. Unfortunately, Bank of England recently announced that it is no longer pursuing CBDC. Yet at the same time, other countries have indicated their interests in CBDC and may deploy these systems in the future.

CBDC is often based on an existing BC model, for example, RScoin is heavily influenced by Bitcoin because it adopts the UTXO(Unspent Transaction Output) model from Bitcoin. However, RScoin has distinct architecture and operation rules, and these are far different from the Bitcoin model.

Bitcoin is first-generation BC technology, it only records the history of transactions and does not record each individual's account balance on BCs. For Bitcoin, every account can only be used once, and this creates significant inconvenience to query and use. Furthermore, the decentralized consensus protocol of Bitcoin is POW (proof-of-work) or Bitcoin mining that costs significant and needless energy to maintain the ledger. POW also leads to high latency and small throughput. As RSCoin uses UTXO model and thus has the similar characteristics in account management.

Ethereum addresses the shortcoming of Bitcoin's lack of account information by using a status tree to keep track of account information. Ethereum also allows "smart contracts" (some called chaincode) to be run on top of BCs to perform computation in an automatic manner. Another advantage of Ethereum is that it provides a BC platform for programmers to develop applications on top of Ethereum without the need of owing their own BC platform.

ECB (European Central Bank), Bank of Japan [11, 12], and Bank of Canada [13] have used several BC models to experiment payment processing and security settlement. Their experiments indicated that current BC or DLT technologies are not mature enough to be applied in

978-1-5386-9266-0/18/$31.00 ©2018 IEEE
DOI 10.1109/DSA.2018.00016

financial markets such as stock trading, clearing, payment, especially in the context of central bank operations. As CBDC will be involved in all these activities and more, thus one can also conclude some BC technologies are not mature for CBDC. These central banks have used relatively mature BC models in their experiments. Yet Bank of Canada has found flaws [13] such as fault-tolerance issue in spite of the fact BCs have built-in redundancy management.

This paper proposes a new CBDC model where each node can have account balance, rather than using the UTXO model. But instead of having one BC system, the new CBDC model in fact a collection of BCs with two different types of BCs: ABC (Account BC) and TBC (Trading BC) [6]. These two BC models allow BCs to be scalable as each type of BCs focuses on one functionality only. In software engineering, one software module or system should perform one function only, by separating different functions to different BCs, the system becomes scalable and manageable.

The Panda model has been simulated with various configurations to evaluate its feasibility and scalability. Simulation models were run on top of TaiShan Sandbox located at Qingdao, China. The sandbox system is a T&E environment where people can try their solutions for feasibility and demonstration. The experimental results are presented in Section 4.

This paper is organized as follows. Section 2 discussed the background of CBDC; Section 3 describes the panda model with theoretical analysis; Section 4 presents the experiment results; and Section 5 concludes this paper.

II. CBDC BACKGROUND

According to Dyson and Hodgson [5], CBDC has two operation models:

Direct Access via Accounts at central banks: Any individual or commercial organization can open an account in the central bank and using these accounts to save or transfer money as shown in Figure 1. The disadvantage of such model is that the central bank will have a tremendous workload, because the central bank would not only need to render services for all users, but also need to implement fraud prevention and anti-money laundering regulations on all accounts.

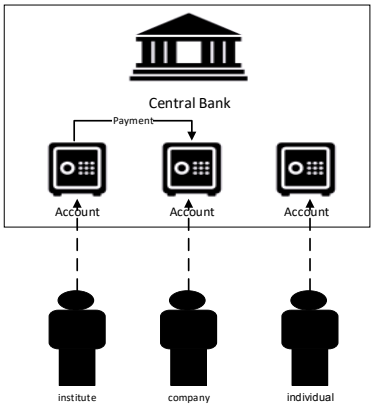

Figure 1 Direct Access

Indirect Access via Digital Cash Accounts: In this model, the central bank would still create and hold the digital cash, but all payment and customer services will be provided by intermediary institutions such as commercial banks. This is similar to the current central bank – commercial banks model. The main difference is that, under CBDC, the money is stored at the central bank instead of at commercial banks. Commercial banks are only in charge of managing accounts, while all payments and transfers are done by the central bank. According to Dyson and Hodgson, the Intermediary Model has four primary advantages: a) It minimizes the burden upon the central bank; b) It is a much more market-driven approach and would encourage firms to improve and expand the services they provide; c) The regulatory framework already exists and can be adopt easily by CBDC; d) It can increase competition in current and payment account services.

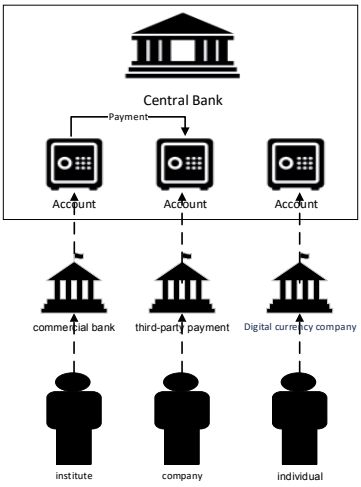

Figure 2 Indirect Access

CBDC will bring significant changes to the financial markets:

- CBDC will preserve the existing central bank-commercial bank model, where the central bank

controls CBDC, and commercial banks provide different services, ensure the liquidity of CBDC together.

- The central bank needs to open up its core system to third party platforms and expand the services of the central bank. As third party payment platforms becomes popular, the regulation of these financial institutes poses an increasing challenge to central banks. Thus, being able to provide them this service allows the central bank effectively regulate the market.
- The central bank can use CBDC to quickly implement monetary and economic policies, such as the interest rate. CBDC can have a different interest rate from traditional currency, and thus allowing two interest rates to co-exist concurrently.
- CBDC can form a new kind of regulatory framework as CBDC is digital currency and is completely stored within the central bank. Therefore, all transactions can be recorded on the BC, creating a massive amount of data that can be monitored and analyzed to help improve market stability.

III. PANDA MODEL

3.1. ABC and TBC

One can divide BCs into two types, ABCs and TBCs [6], where an ABC is responsible for account information and a TBC is responsible for transaction information and transaction history.

An ABC is responsible for maintaining account information. For instance, a small commercial bank could maintain one ABC, while a larger bank can maintain two or more ABCs to handle the load. Each ABC adopts multi-node BC design, thus making the records secure from unauthorized modifications.

An ABC has the following main operations:

Account creation: Creating a new account involves generating the account information, creating a new key pair, etc. The key information of the account needs to be stored on the BC to prevent illegal tampering.

Account Uploading: Transferring the account information to a TBC. When uploading the account, the system needs to use Byzantine algorithms at the ABC nodes to ensure consistency and prevent double-spending problems. To allow the account continue to be available to the client, the system creates a sub-account with required amount for each account when uploading and only locks up the sub-account.

Account Updating: When a TBC completes a transaction, it will return information to the ABC. If the transaction is successful, the ABC needs to update the account balance. These changes must be kept on the BC while using Byzantine algorithms to ensure consistency and prevent tampering. If the execution of the transaction fails, then the ABC will not update the account balance.

A TBC is responsible for executing and keeping transactions with a multi-node structure. TBC does not keep any account information, it only requests account information from ABC when needed. When the transaction is completed, TBC will send account information to ABC and record the transaction on its BC at the same time.

TBCs can be divided into two types: internal TBCs and cross-ABC TBCs. Internal TBCs works only with one ABC, and can efficiently execute transactions that only involve accounts from that ABC. Cross-ABC TBCs handle transactions that involve accounts from different ABCs, these TBCs takes more time to process transactions.

To further protect the privacy of users and banks, the data recorded on TBCs can be encrypted so that only participating banks and central banks can view this data. This design requires a permission management system of members participating in the BC. This design means that only central bank can access all the data on CDBC systems, but others will have limited but consistent views.

The ABC-TBC architecture has been used to design large trade clearing systems with bigData [14].

3.2. Architecture

Based on ABC and TBC architecture, the central bank architecture can be divided into three parts:

The first part is maintained by ABCs. Account information includes information of the account owner and balance, etc. All edits to account information will be recorded by ABCs to prevent unauthorized tampering.

The second part is TBCs, and they record all the transactions made and data are stored in TBCs. When a user needs to conduct a transaction, he will send request to a TBC, then the TBC sends an account upload request to the involved ABCs according to the input and output sides of the transaction, and process the trade, then return the updated account information to the ABCs to clear.

The third part is the central bank. The central bank has two primary functions:

1. Issuing CBDC. All CBDC comes from the central bank, and the central bank regulates the economy of the entire country and maintains the stability of the currency by controlling the issuance of currency.
2. Monitor the behavior of all BCs including all ABCs and TBCs. The central bank has to audit all ABC and TBC activities, and these chains must send their activity diaries to the central bank. This allows the central bank to track and monitor how CBDC is traded, and evaluate the state of the market and prevent any illegal activities. This can be done by having the central bank participate in these ABCs and TBCs as a node in these BCs.

The central bank is made up of BCs, the regulatory technology and bigData analysis. All transactions and accounts will be recorded on the central bank's BCs to maintain data consistency. The nodes also provide data to the regulatory agency, and the bigData analysis department. This data comes out from BC, and thus has high reliability and accuracy.

Regulatory technology can detect and prevent the execution of illegal transactions in a timely manner, simplify the complexity of KYC(know-your-customer) and AML(Anti Money Laundering) and reduce the costs.

Aside from being used in regulation , bigData analysis can also be used to analyze the data on economic operations, help the central bank understand the economic development trend, and thus be able to scientifically adjust economic policies and promote economic development.

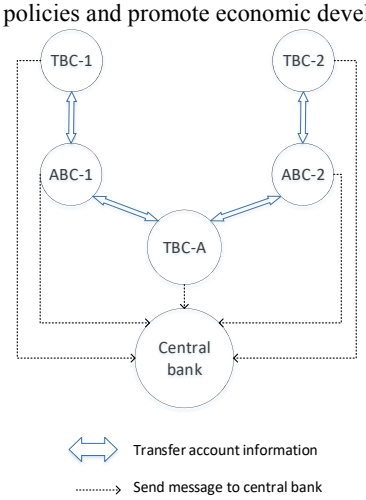

Figure 3 Architecture

As shown in the above diagram, ABC-1 is responsible for maintain Bank-1's account information, and TBC-1 is responsible for processing transactions within ABC-1. ABC-1 and TBC-1 combine to form the internal architecture of Bank 1. Similarly for ABC-2, TBC-2 and Bank 2.

When cross-ABC transaction occurs, such as when an account on ABC-1 initiates a transactions with an account on ABC-2, then it must be processed by TBC-A, which draws account information from both ABC-1 and ABC-2, then updates them both when the transaction is done. Then the related ABCs will update their own information and completes the transaction.

The central bank will record and supervise all actions of ABCs and TBCs.

3.3. Transaction Flow

Given transaction that account-A transfers *n* dollars to B, where A is managed by ABC-1 and B is managed by ABC-2, the flow of the transaction is as follows:

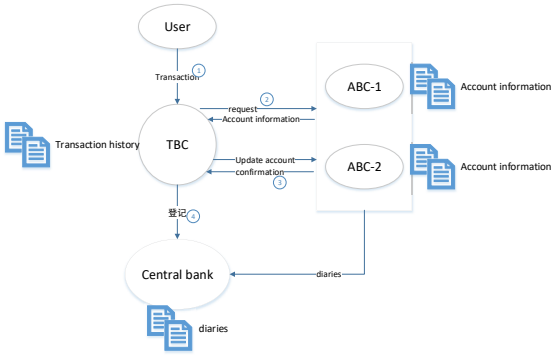

Figure 4 Transaction Flow

1. Initiation of the transaction: user-A sign the information (A->B, N) with his own digital signature, then sends the data to TBC.
2. Uploading of account information: After the TBC receives the information, The BC will send the request to the related ABC. Then ABC-1 will verify the request which include the authentication and the account balance, etc. If the verification is successful, ABC-1 will lock some money of account-A, and send account-A's information to TBC. ABC-2 will perform the same steps for account-B.
3. Transaction execution: Nodes in the TBC will vote for transaction and build blocks. When block building is completed, the TBC will return the successful information to ABC-1 and ABC-2, and ABCs will update their account information.
4. Audit: when ABCs and TBCs complete this transaction, they will send their diaries to the central bank and the central bank will audit it. If the central bank has a node participating in these ABCs and TBCs, it will not be necessary to send information.

At this point, the transaction is completed. For transactions within the same ABC, the process is similar to the above except that the related ABC will be just one ABC.

3.4. BC Algorithms

3.4.1. TBC Algorithms

A TBC uses the following algorithm to conduct transactions assuming two ABCs are involved, A and B:
1. Bank A freezes the assets needed to conduct the trade, and sends the account information to the TBC. When the TBC receives the transaction, it will verify the information received from A. It is the same for bank B.
2. The transaction is executed on the TBC, the TBC will build and vote the block. If most nodes agree, this block will be approved. If not, this block will be dropped.
3. If transaction is successful, the TBC will calculate the balance, sends the transaction and balance to all bank A's nodes, as well as bank B's node. If transaction fails, TBC will send the original balance to these nodes.

4. Account information in the TBC will be marked as "expired" indicating that the data is no longer available for trading.

Table 1 TBC Algorithms

Algorithm TBC	
Input： a transaction Tx1（A->B,N）	
1	CheckTx(Tx1)
2	Forall Account in Tx1
3	Request(Account.owner, Account)
4	ACC←Receive from Account.owner
5	CheckBalance(ACC)
6	Byzantine(Tx1)
7	Execute(Tx1)
8	Forall Account in ACC
9	Send(Account.owner, Account)
10	Receive(Account.owner ,success)
11	Delete (Account)
12	Send(CentralBank,Tx1)

3.4.2. ABC Algorithms

ABCs primary algorithm involves uploading and updating account information.

Uploading account:

1. Verify if the request is valid (if the digital signature is correct; if the account belongs to the particular ABC; if the account is locked; if there is enough funding), if the request is not valid then return a disagreement;
2. Make a vote on the account to ensure that the account status is consistent
3. Lock the account, and at the same time, encrypt the account information and upload it to TBC.

The account-uploading algorithm is as follows:

Table 2 Account-uploading Algorithm

Algorithm ABC. Upload	
Input： Tx1（A->B,N）, Request(Account)	
1	if CheckTx(Tx1) = 0 and CheckReq(Request(Account)) = 0
2	Byzantine(Account)
3	Lock(Account.sub_account)
4	return (Account.sub_account)
5	else return(false)

Updating Account:

1. Verify if the request is valid, and if the account is currently locked;
2. Pre-updating account information;
3. Make a vote on the account to ensure that the account status is consistent after that transaction;
4. Update the account balance and unlock the account;
5. Sends updated information to the central bank.

Table 3 Algorithm of Updating Account Information

Algorithm ABC.Update	
Input： Tx1（A->B,N）, Request(NewAcc)	
1	CheckTx(Tx1) and CheckReq(Request(NewAcc))
2	if Lock(Account)
3	PreUpdate(NewAcc))
4	Byzantine(NewAcc)
5	Update(NewAcc)
6	UnLock(Account)
7	return (success)
8	send(CentralBank,Tx1, NewAcc)
9	Else return(false)

3.5. Some Features

This section evaluates the Panda model based on the following criteria:

1. **High throughput and low latency**: Transactions are handled locally if it is related to one ABC. This kind of transactions will be processed with low latency. Cross ABC transactions require TBCs to use high-speed network to reduce latency.
2. **Low energy consumption**: Panda model uses permissioned BCs without mining. The maintenance of the BCs does not require massive energy consumption.
3. **Security**: All data in BC uses encryption and digital signature must be verified when initiating transactions. Furthermore, Byzantine protocols with reputation systems [10] to ensure that any compromised nodes can be identified quickly, and those compromised nodes will be removed from the system immediately after detection.
4. **Privacy**: Every ABC can protect the privacy of their customers, as it will release only the information necessary for transactions. Otherwise other BCs cannot access the information.
5. **Monitoring**: The central bank can directly monitor the activities on ABCs and TBCs by incorporating its own node in every ABC and TBC. Every new block built will be available for the central bank for monitoring.
6. **Reliability**: The Panda has significant built-in redundancy, specifically the overall network is an IOB model with as many BCs as needed. Each BC has multiple nodes whether it is ABC or TBC, ABCs and TBCs also share significant information during transaction processing. Thus significant redundancy has been incorporated into the model to ensure that data will be lost easily. The detail of reliability will be discussed in section 3.8.
7. **Timeliness**: after a transaction is processed, the system will give timely feedback to the client whether the transaction was successful or not.

3.6. Regulation

As for regulation of the commercial banks, the central bank has the following two options:

1. Whenever a transaction is completed, the information is send to the central bank, and they maintains its own BC to keep record and analyze this information.
2. The central bank can set up its own node on ABCs and TBCs, and thus gaining access to the original data.

A simplified Panda-model model is as following:

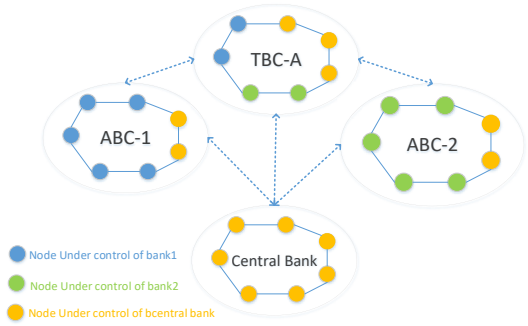

Figure 5 Node Distribution

The nodes on TBC should be made up of the central bank and participating commercial banks.

The central bank keeps records of all commercial bank transactions, and can analyze the activities of every account.

To implement KYC and AML, the central bank and commercial banks can both verify the identity of parties involved in a transaction, to ensure the transaction is legal.

Illegal or faulty transaction can be discovered and stopped as soon as detection. Potentially, CBDC transactions will be real time as each CBDC dollar is considered as cash, and settlement can be instant. When a user logs in, the system will verify its identity, and if there is any cause for suspicion, the system will lock down the associated accounts and prevents any related transactions from being made.

3.7. Expandability

ABC expandability: It is possible to extend an ABC if it can no longer handle the load by splitting the ABC into multiple ABCs, each handling one segment of accounts. A load balancing strategy is used between these ABCs to maintain the ledger together. This is sharding, commonly used in databases as well as in digital currencies. However, in databases each shard will not interfere with each other, but in most digital currencies, these shards do interfere with each other. In the Panda model, as ABC handles account management only, thus if it is sharded, each shard will not interfere with each other like shards in databases.

TBC expandability: As the number of transactions between banks increases, one can add additional TBCs to deal with the additional load. All TBCs can be run concurrently, and thus the system's processing speed will increase as the number of TBCs increases.

System expandability: For the entire CBDC system, if a new bank is opened, the bank needs to establish an ABC and internal TBC, and can choose to join existing cross-bank TBCs. In addition, if there are a large number of transactions between certain banks, a dedicated TBC can be set up to handle these transactions to increase the efficiency of the entire system.

For example, in the following diagram, Bank-3 opened its own ABC-3 and internal bank TBC-3, while Bank-3 joined the original TBC-A to be able to conduct transactions with Bank-1 and Bank-2. At the same time, there is a large amount of business between Bank-2 and Bank-3, so TBC-B can be set up to specifically handle the business between Bank-2 and Bank-3.

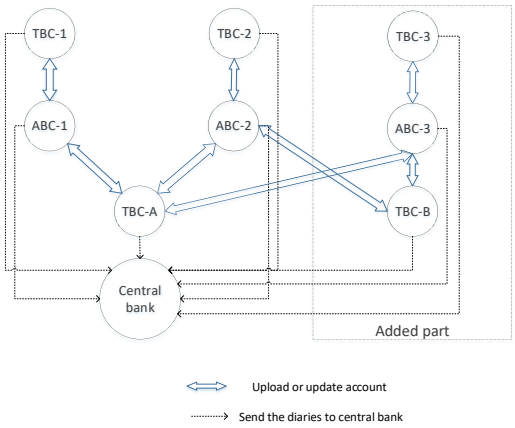

Figure 6 System Expandability

3.8. Reliability

This section evaluates the Reliability of Panda model. This section assumes:

- One transaction involves two ABCs only;
- Each ABC and TBC is made up of n nodes; and
- The probability of each node making a mistake in a transaction is P and every node is independent.

According to the Byzantine agreement, the system will fail only when more than 1/3 of the nodes in the system have failed, and thus for a single transaction, the probability of failure is:

$$\sum_{i=\lfloor \frac{n}{3} \rfloor+1}^{n} C_n^i P^i (1-P)^{n-i}$$

C is for "combination", which means the number of selections of i items from n nodes.

For a transaction, if any one ABC or TBC fails, then the system will fail. Thus the probability for overall system failure is:

$$1-(1-\sum_{i=\lfloor \frac{n}{3} \rfloor+1}^{n} C_n^i P^i (1-P)^{n-i})^3$$

The probability of continuously processing t transactions without any failure is:

$$1 - (1 - \sum_{i=\lfloor \frac{n}{3} \rfloor + 1}^{n} C_n^i P^i (1-P)^{n-i})^{3t}$$

Let $P = 0.01$, then the probability of executing 1,000,000 transactions without failure is as follows:

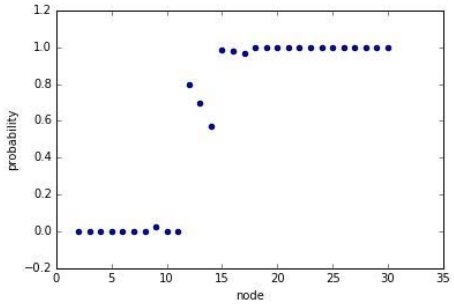

Figure 7 Probability of Executing Successfully

As the diagram shows, when n increase, the probability of success increases. But when $n > 18$, the probability of success stays relatively constant. s

3.9. Network Traffic

For each block, ABC needs to run BFT two times and TBC need once, the traffic of a Byzantine agreement is about $2n^2$, thus, the total network traffic of BFT in a block is about $6n^2$.

The client needs to send a message to all TBC nodes, thus the number of traffic is n. During the unloading account stage, every node in a TBC needs to send a message to corresponding node in the related ABC and ABC need to reply, the network traffic is $4n$. It is the same for updating stage. The total network traffic of them in a block is about $9n$.

Thus the total network traffic for the entire process is about $6n^2 + 9n$.

3.10. Comparisons

RSCoin is the CBDC model designed by University College London, and its design principles are as follows:

1. Divide nodes into groups, and make each group handle different transactions, and thus as the number of group increases, the speed will also increase;
2. Groups use clients to pass messages, this design avoids the shortcoming of mass communication in BCs.
3. Uses a two-phase commit protocol instead of Bitcoin's POW.
4. Every node is permitted by the central bank.

The following table is a comparison between RSCoin and Panda models.

Table 4 Comparisons between RSCoin and Panda-model

	RSCoin	Panda-model
Distributed ledger	Every mintette keeps a part of the ledger the does not use a BC, the central bank keeps all record using chain structure.	ABCs are responsible for account information and TBCs for transaction data.
Throughput	Throughput increases fastly with the number of mintettes. The original speed is 2000 transactions/s.	Throughput increases fastly with the number of ABCs and TBCs.
Low Latency	Relatively high latency as each transaction requires four stages.	Local transactions have relatively low latency, while cross region transactions have higher latency.
Low Energy	Does not use mining, relatively low energy cost.	Does not use mining, relatively low energy cost.
Security	Security is related to the number of mintettes. Uses two-phase commit protocol.	Security is related to the number of nodes in ABCs and TBCs, use BFT protocol.
Privacy	Mintettes are assigned to transactions dynamically, and do not have all transaction information.	Account information is only uploaded while processing the transaction at TBCs, complete account balance and history at ABCs like traditional banking.
Regulation	The central bank audits all transactions that come from mintettes.	The central bank audits all transactions and has nodes on ABCs and TBCs.
Timely Feedback	Relatively fast, each transaction will receive a confirmation and results.	Relatively fast, each transaction will receive a confirmation and results.
Expandability	Uses permissioned mintettes, and speeds increases with the number of mintettes.	Uses distributed ledger and load balancing. Separate the accounts and transactions into distinct BCs, adding TBCs and splitting ABCs allow the system to handle as many accounts and transactions as needed.

In terms of business models, Panda-model is suitable for a central bank-commercial bank model. A transaction in RSCoin requires permission from different mintette groups, while a transaction in the Panda model requires verification through ABCs only.

Secondly, RSCoin uses a dynamic assignment of node groups to handle transactions, and thus account information is stored in different mintette groups. This has two implications:

1. It is difficult to protect the privacy of the customer, because customer cannot choose which mintette group to store information;
2. Banks cannot provide personalized service to its customers because the customer information is fragmented by various mintette groups. While in the Panda model, the client can choose a specific ABC to handle its account, and store its account formation. Thus, the commercial bank can provide personalized services, and also be able to analyze the relevant data.

IV. PERFORMANCE

To verify these performance and latency for the system to confirm transactions, we implemented the Panda model presented in Section III using LaoShan BC and measured its performance on TaiShan sandbox provided by Andrew International Sandbox Institute in Qingdao. Taishan Sandbox is the world's first blockchain-based industrial sandbox and the first blockchain sandbox platform with big data support. Each ABC or TBC consists of 4 nodes with a 2.4 GHz processor and 2GB RAM. We Use RSA1024 for key distribution and AES256 for data encryption. Note that TaiShan Sandbox system is a T&E(Testing environment) environment with 20 servers supporting 250 BCs in a cloud environment. To obtain better performance, physical servers need to be used.

Each transactions consisting of one input and one output which is similar to bank transfer, we use (Alice, Bob, 10) to represent that Alice transfers 10 to Bob.

This section focuses on the scalability of the Panda model, that is, how throughput changes when ABCs and TBCs increase. The index is the number of transactions processed by the system per second (tps, Transaction per Second).

4.1. LaoShan BC

We first tested the speed of our BCs on TaiShan Sandbox. We send simulated transactions to our 4 BC's nodes and count the tps and the average delay of transactions:

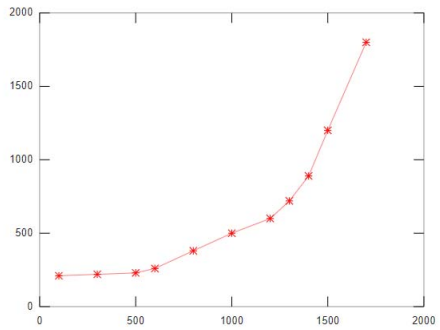

Figure 8 Delay of Transaction

From figure 8, one can see that when we send a few transactions, the system still needs 250ms to process, which point to an intrinsic delay due to networking overheads of less than 250ms.

When we increase the number of transactions, the size of each block will increase, but the production speed of each block will slow down. Generally speaking, when the number of transaction increases, both throughput and average delay will increase. But, the average delay increases faster.

Considering the user experience, the throughput mentioned in the following tests refers to the throughput of the system when the delay is less than 1 second.

4.2. Increase TBCs

This section evaluates the impact of increasing the number of TBCs on the system while keeping the number of ABC unchanged. The initial system consists of an ABC and a TBC. Based on the initial system, we continue to increase TBCs and measure the throughput.

Table 5 Throughput While Increasing TBCs

The number of TBCs	Throughput
1	912 tps
2	1428 tps
3	1476 tps
4	1390 tps
5	1258 tps

From the above experiments, one can find that by adding a TBCs can bring about a certain increase in throughput. However, when the TBC continues to increase, one can see diminishing returns.

4.3. Increase ABCs

This section evaluates the impact of increasing ABCs on the entire while keeping the number of TBC unchanged. The difference from adding TBCs alone is that we need to consider the ownership of accounts. In the case of one ABC, all the account are managed by the same ABC.

When we increase the number of ABCs, we need to consider which ABC the account belongs to. In this section, we simplify the problem: For multiple ABCs, we consider ABCs inner transfer and the transaction format is (ABC-1, ABC-1, n). We will analyze cross-ABC transaction in the next section.

Table 6 Throughput While Increasing ABC

The number of TBC	Throughput
1	912 tps
2	1195 tps
3	1184 tps
4	1175 tps
5	1171 tps

From the above results, one can find out that similar to adding TBCs, the throughput increases while add one ABC. However, continuing to increase TBC does not bring any benefit. And different from adding TBCs, The negative impact of adding ABC is smaller because different ABCs will not interfere with each other when internal transactions are made.

4.4. Cross-ABC Transactions

When we increase the number of ABC, we need to consider the ownership of different accounts. For example, for two ABC systems, there are four transactions: (ABC-1, ABC1), (ABC-1, ABC-2), (ACB- 2, ABC-1), (ABC-2, ABC-2). In this section, we use a system with two ABCs and two TBCs to evaluate the impact of different types of transactions.

Table 7 Throughput of Different Transaction

Transactions in TBC-1	Transactions in TBC-2	Throughput
(ABC-1,ABC1)	(ABC-2,ABC-2)	1818 tps
(ABC-1,ABC1) (ABC-2,ABC-2)	(ABC-1,ABC1) (ABC-2,ABC-2)	1722 tps
(ABC-1,ABC-2)	(ACB-2,ABC-1)	1706 tps
(ABC-1,ABC-2)(ACB-2,ABC-1)	(ABC-1,ABC-2)(ACB-2,ABC-1)	1670 tps
mixed	mix	1590 tps

One can see that different transactions do have an impact on system throughput. If a TBC deals only with one type of transactions, it can increase the throughput of the system as TBC needs to obtain two ABCs authorization and transfer account information to these ABCs. But if TBC deals with mixed transactions, the throughput decrease because TBC needs to communicate with all related ABCs.

4.5. Increase both ABCs and TBCs

This section evaluates the impact of increasing both ABCs and TBCs. Image that ABC-1 need to make transaction with ABC-2 and ABC-3, the whole system will be slow as ABC-1 is overload. To solve this problem, we try to divide ABC-1 into two part and add TBC among them.

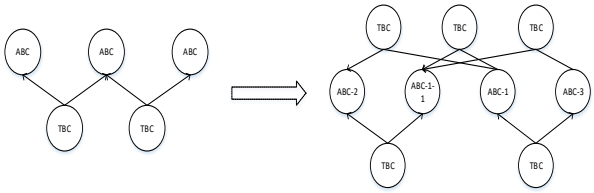

Figure 9 Increase Both ABC and TBC

For architecture 1, the overall system throughput is 1683. For architecture 2, we divide ABC-1 into ABC-1-1 and ABC-1-2, and set TBC between different ABCs. The throughput of this system reached 3316, and the improvement effect was obvious. In real life, TBCs can be set according to dynamic requirements. This demonstrates that by adding both ABCs and TBCs at the same time, performance can be significantly improved.

V. CONCLUSION

As CBDC receives significant attention, more central banks have announced that they will issue digital currency in the future. This paper proposes the Panda model with ABCs and TBCs, by splitting and adding ABCs, and adding TBCs, the CBDC system can be scaled up with as many BCs needed in the system. The model also fits well with the current central bank-commercial bank model where a commercial bank can host one or more ABCs, and they do transactions with other banks via TBCs. The Panda model is in fact an IOB model and can be used in applications other than CBDC.

ACKNOWLEDGMENT

This work is supported by National Key Laboratory of Software Environment at Beihang University, National 973 Program (Grant No. 2013CB329601) and National Natural Science Foundation of China (Grant No. 61472032) , (Grant No. 61672075) and (Grant No. 61690202).

REFERENCES

[1]G Danezis and S Meiklejohn. “Centrally Banked Cryptocurrencies”[C] NDSS Symposium. 2016.

[2]O'Dwyer K J, Malone D. “Bitcoin Mining and its Energy Footprint”[C] Irish Signals & Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies. IET, 2014:280-285.

[3] Ali R, Barrdear J, Clews R, et al. “The Economics of Digital Currencies” [J]. Bank of England Quarterly Bulletin, 2014, 54:276-286.

[4] Robleh A, Barrdear J, Clews R, et al. “Innovations in Payment Technologies and the Emergence of Digital Currencies.”[J]. Bank of England Quarterly Bulletin, 2014, 54:262-275.

[5] Ben Dyson and Graham Hodgson, “Digital Cash: Why Central Bank Should Start Issuing Electronic Money,” Positive Money 2016, www.positivemoney.org

[6] W.T. Tsai, Blower R, Zhu Y, et al. “A System View of Financial BCs”[C] Service-Oriented System Engineering. IEEE, 2016:450-457.

[7] Nakamoto S. “Bitcoin: A Peer-to-peer Electronic Cash System” [J]. Consulted, 2009.

[8] V Buterin, “Ethereum: A Next-Generation Generalized Smart Contract and Decentralized Application Platform" http://ethereum.org/ethereum.html

[9] W.T. Tsai, et al. "Intellectual-Property BC-Based Protection Model for Microfilms." Service-Oriented System Engineering (SOSE), 2017 IEEE Symposium on. IEEE, 2017.

[10] W.T. Tsai, Xiaoying Bai, and Lian Yu. "Design Issues in Permissioned BCs for Trusted Computing." Service-Oriented System Engineering (SOSE), 2017 IEEE Symposium on. IEEE, 2017.

[11] European Central Bank and Ban of Japan, “Payment Systems: Liquidity Saving Mechanisms in a Distributed Ledger Environment,” Stella, a joint research project of the European Central Bank and the Bank of Japan, Sept. 2017.
https://www.ecb.europa.eu/pub/pdf/other/ecb.stella_project_report_september_2017.pdf

[12] European Central Bank and Ban of Japan, “Securities Settlement Systems: Delivery-versus-Payment in a Distributed Ledger Environment,” Stella, a joint research project of the European Central Bank and the Bank of Japan, March 2018. http://www.boj.or.jp/en/announcements/release_2018/data/rel180327a1.pdf

[13] Bank of Canada, “Project Jasper: Are Distributed Wholesale Payment Systems Feasible Yet?” June 2017. https://www.bankofcanada.ca/wp-content/uploads/2017/05/fsr-june-2017-chapman.pdf

[14] W.T. Tsai, E.Y. Deng, et al. “Application of Blockchain to Trade Clearing”, To appear in the proceedings of The 3rd International Workshop on Blockchain, Smart Contracts, and Digital Society, 2018.