BlockNDN:A Bitcoin Blockchain Decentralized System over Named Data Networking

Tong Jin, Xiang Zhang, Yirui Liu, Kai Lei*
Institute of Big Data Technologies
Shenzhen Key Lab for Cloud Computing Technology & Applications
School of Electronic and Computer Engineering (SECE)
Peking University, SHENZHEN 518055, PR. CHINA
Email: 1501213944@sz.pku.edu.cn, Corresponding Author*: lei@pkusz.edu.cn

***Abstract*—Blockchain provides a new approach for participants to maintain reliable databases in untrusted networks without centralized authorities. However, there are still many serious problems in real blockchain systems in IP network such as the lack of support for multicast and the hierarchies of status. In this paper, we design a bitcoin-like blockchain system named BlockNDN over Named Data Networking and we implement and deploy it on our cluster as well. The resulting design solves those problems in IP network. It provides completely decentralized systems and simplifies system architecture. It also improves the weak-connectivity phenomenon and decreases the broadcast overhead.**

Keywords—blockchain; NDN; peer-to-peer; bitcoin; next generation networks

I. Introduction

Blockchain is a shared, unchangeable ledger for recording the history of transactions [1]. It was first used in bitcoin, functioning as a data structure. Now it is abstracted from bitcoin system and has developed into a promising technology solution which provides secure and valid achievement of distributed consensus in areas such as supply chain & logistics, finance and healthcare. A blockchain is comprised of digitally recorded data in packages called blocks. Each block contains the hash value of the previous-block which comes out first and eventually forms a linear chain of blocks with the others. Different distributed storage solutions and consensus schemes should be designed in a blockchain system depending on different requirements, which enables nodes confirm and record information without authorities. To take bitcoin [2] as an example, it uses proof-of-work (PoW) [3] as consensus algorithm and designs full node storage that the so-called full nodes in the system should maintain the whole blocks by themselves.

However, the growth of blockchain still hits bottlenecks and impediments at present. The weak-connectivity and improper protocols cause the propagation delay in IP network which causes blockchain forks in some systems [4]. Besides, although blockchain is a fully decentralized system by definition, it is hard to establish the connections between homogeneous nodes in practice. For example, with neighbor discovery in bitcoin, some IP addresses of capable volunteers are recommended as the connection options for users. These volunteers are called DNS seed nodes which function as infrastructure to help normal users to communicate with each other. They increase the possibility of the existence of supernodes and bring some insecurity factors such as frauds and single point of failure. Also, the TCP/IP protocol does not provide native support for multicasting, resulting in an increase in data transmission overhead.

Named Data Networking (NDN) project [5] is an important candidate for next-generation internet architectures. NDN network puts contents into first place. Contents are named following hierarchical naming rules and these names replace the role of IP addresses. Interest packets and data packets are the two fundamental packet types in NDN. Each interest packet corresponds to only one data packet. Nodes in NDN pull data from networks using interest packets. The one side requests data, the consumer, sends out an interest packet carrying the data name and this interest will be forwarded to the other sides, the producers. Producers then send back a data packet to the requester along the same route form where its interest has come. The intermediate routers with caches selectively store data packets forwarded before so that data can be sent back immediately once new consumers request for the same content. It's worth pointing out that the architecture of NDN provides a good fit to the large amounts of static content via various P2P overlay networks [6].

Does blockchain fit better on NDN than on IP? What are the advantages? To investigate on these questions, we implement a bitcoin-like blockchain system that runs on NDN network. In our system, the hash values function as the unique identification of blocks. A node sends out an interest with the hash of the latest block it stores locally. If the other node receives this interest, finding that it has the same block in its blockchain and that block is not the latest one, it will collect all blocks behind into a data packet to send to the consumer. In this way, the new joiner or disconnected nodes can update their state quickly, that is, they can get the whole blocks in the system in a short time. Nodes can also request specific blocks in the system using broadcast mechanism. Our analysis shows that the bitcoin blockchain system over NDN circumvents those problems which may occur in IP network. The result shows that we have designed a more decentralized and simpler system. Nodes do not need to get information about other nodes like the identities and locations. They just search for specific blocks data in the networks. Also, NDN supports multicast and broadcast by nature, which reduced the cost of message passing.Note that a NDN router with content

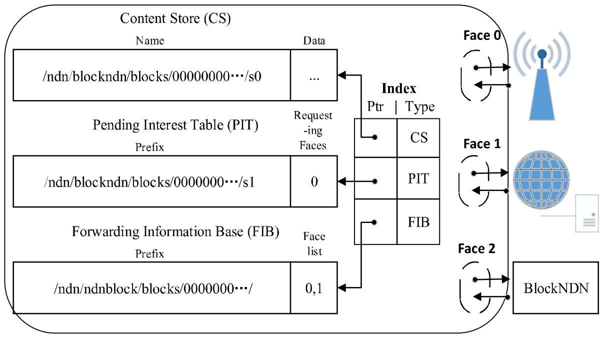

Fig. 1. The core NDN packet dorwarding engine.

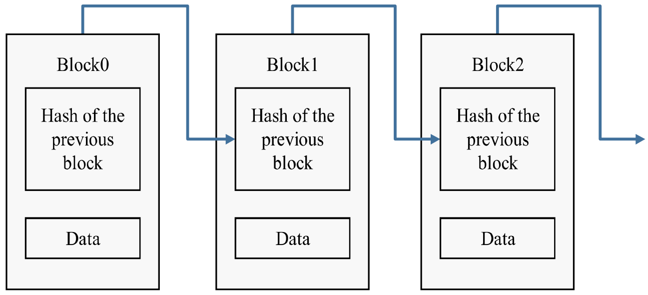

Fig. 2. A blockchain with a linked list of block.

store can work as a cache to optimize the system performance.

The main contribution of this paper is summarized as below:

- We creatively implement a blockchain system which is a groundbreaking experiment in NDN and make our own contribution to the promotion of NDN project.
- We summarize the shortcomings of blockchain systems over IP networks which include the possibility of the existence of supernodes and the corresponding fraud and the low transmission efficiency.
- We find out a more suitable way to combine NDN network and blockchain. Different from the traditional implementation, we don't simply mimic the architecture of bitcoin systems in IP network but propose a new design. Our design takes advantage of the superiority of content distribution in NDN. By using our naming rules, nodes fetch data from individual participants and send out self-state information at the same time.
- We analyze the advantages of our blockchain system over NDN comparing to those over IP. BlockNDN resolves the problem of systems over IP that we have summarized. Our design provides completely decentralized systems and simplifies system architecture. It can decrease the broadcast overhead. Our system also provides more anonymity to prevent both eavesdropping and traffic analysis.

The rest of our paper is organized as follows. We give a brief review of NDN and blockchain in Section II, and describe the design of our system, BlockNDN, in Section III. The Section IV shows its advantages and the Section V introduces how we implement our system. In the end, we discuss the future work and conclude the paper in Section VI.

II. Overview of NDN and Blockchain

A. Named Data Networking Background

NDN is a completely new networking model compared with TCP/IP. Data is the most important entity in NDN. They are identified and retrieved with data names following a hierarchical naming mechanism which is similar to the URI naming structure in current IP network. NDN architecture has two basic communication units: interest packets and data packets, both of which carry data names. Fig. 1 presents the three key data structure: Forwarding Interest Base (FIB), Content Store (CS) and Pending Interest Table (PIT). To request data, a data consumer sends out an interest packet which carries the unique name of the desired data. The intermediate NDN routers add an entry to the PIT of its own for recording the incoming interface where the interest packet comes in. Then they forward this packet by looking up the name in its FIB which has been populated by name-based routing protocols rather than ip-address-based [7]. Once the interest packet reaches a node with the requested data, the data's owner will send back a data packet. This data packet can trace in reverse the path back to the data consumer. When the intermediate routers receive the data packet, they will check whether the packet's data name has been recorded in the PIT. If there is a corresponding entry, they will forward the data packet back to the specified interface and store it in their CS so they can send back the specific content immediately next time. Each data packet is cryptographically signed, enabling the receiver to check the integrity and provenance of the data. There might be several producers to one consumer at the same time. Each producer has the same data packet that the consumer request for, so what is noteworthy is that NDN data delivery naturally supports multicast delivery and content distribution function [8].

B. Blockchain Background

On account of its particular subversiveness, blockchain even draws more attention than bitcoin-a peer to peer electronic cash system, from which it comes. The exclusive innovation worked as a data structure in bitcoin at the very beginning, being like an engine of the credit and security system of the bitcoin. Now it works as an infrastructure of distributed peers, providing services to different industries like banking, healthcare, supply chain and Internet of Things.

In brief, the blockchain data structure is a linked list of a series of blocks that is built with hash pointers (Fig. 2). Each block contains a list of data units called transactions and a message digest of the previous block to ensure that the previous transactions will not be tampered [9]. Every blockchain system should resolve consensus problem which means all the participator should assent to the order of blocks and have the same capacity of knowledge about the blockchain of the system. There are many consensus schemes such as PoW, Proof-of-Stake (PoS) [10], PBFT [11] and RPCA [12]. Different schemes will work in different scenarios. Distributed storage solution should also be designed.

We introduce a bitcoin blockchain here as a concrete example to show how blockchain works. Bitcoin blockchain

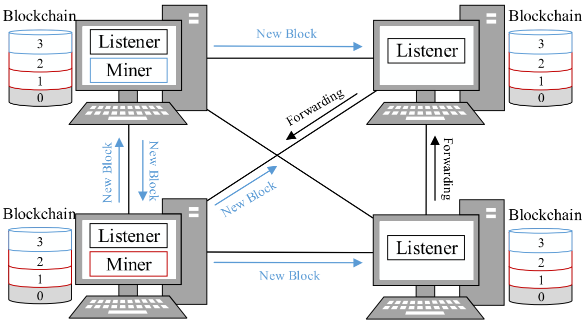

Fig. 3. Overview.

system uses PoW as consensus algorithm and nodes maintain the whole blockchain in their memories. Following demonstrates how a bitcoin blockchain system works.

- Nodes generate transactions and broadcast them to all users. Other nodes called miners collect these transactions into a block and try to make it valid. Miners should complete their PoW by changing the nonce values of the blocks repeatedly until the hash results satisfy system requirements. A Node creates a new block once it satisfies the PoW, and this block cannot be falsified without redoing the whole work which has been done in the system.
- When a new valid block is generated, the creator broadcasts it to the network. Every node checks the PoW and accepts this block only if all transactions in it are valid. After the acceptance, the block gets stored and another new block created later will chain after it.
- Eventually, every receiver maintains a chain of the blocks recording all the transaction chronologically. The longest chain in the network is the legitimate bitcoin blockchain (the best blockchain).

If the whole system behaves correctly, the longest bitcoin blockchains of users are same, which indicates that nodes arrive at a consensus.

III. BLOCKNDN DESIGN

A. Overview

According to different requirements, different blockchain systems are designed. Given the popularity of bitcoin, we decide to implement a bitcoin-like blockchain system named BlockNDN. Our system is totally decentralized. Initialization is complete after nodes download all blocks from the first to the latest one. Meanwhile, miners (Fig. 3) encapsulate data into blocks and broadcast them once they implement the proof-of-work. Listeners will add a new block to the local blockchain if they have checked the correctness. As a result, the height of the best blockchain [2] in the system continues to increase.

B. Naming Convention

Similar to the URI naming structure, data names in NDN follow a hierarchical naming mechanism, which is also one of the most significant aspects of application design. Because NDN doesn't have the concept of IP addresses, the ability to map

$\underbrace{\text{/ndn/blockndn}}_{\text{Application}}\underbrace{\text{/blocks}}_{\text{Function}}\underbrace{\text{/00000000000000000000000000024b89b4f...}}_{\text{Message digest}}$

a) An example of getblock data name

$\underbrace{\text{/ndn/blockndn}}_{\text{Application}}\underbrace{\text{/sync}}_{\text{Function}}\underbrace{\text{/0000000000000000000000000096cc85b3...}}_{\text{Message digest}}$

b) An example of sync data name

Fig. 4. The naming convention.

Bitcoin addresses directly to IP data [13] disappears, which can prevent both eavesdropping and traffic analysis. There are two different naming rules for information exchange in different scenarios. Every name consists of three parts: the application fixed prefix, the function prefix and the digest prefix.

The naming rules of BlockNDN are divided into two classes: one for application data names and one for sync data names. We name every block in the system blockchain following getblock naming rule. Each name carries the message digest of the previous block which the name's owner (the block) links to. As the blockchain structure is linear, a name prefix carrying the hash value of the previous block shows the place of the block in the chain. We follow the same principle in IP that uses digests to mark block height in blockchains rather than numerical symbol. We define the first block in blockchain, the block at height 0 or depth 0 as well. All users know the hash value of the first block which is shown in Fig. 4(a). Part one of this getblock data name is the application name so that the interests can be forwarded towards the producers directly. The purpose of the part two indicates that it is used for naming blocks and that the digest of the previous block is shown in the third part. Interests with the name in Fig. 4(a) suggest that the consumers have requested for the block at height 1.

Similarly, the name for sync data also consists of three parts (Fig. 4(b)). However, the third part serves the purpose of carrying the digests of the highest blocks in users' own memories. This kind of interests following this naming rule can be sent as synchronous requests when the requesters are new ones or have been disconnected from the network for quite a while, in which case it becomes time-consuming to ask for blocks one by one. The responders then extract the current state digests from those interests and return corresponding data packets when they compare to their own blockchain.

It is noteworthy that several new blocks can be generated simultaneously, which can cause blockchain forks in the system. This problem can be solved with the exclude filter [14] like what is done in the ChronoSync. Exclude filter is a kind of selector that can be sent along with the interest to exclude data that the requester no longer needs.

C. Updating Status

When consumers are new ones or have been disconnected from the network for quite a while, it will take a long time to ask for data by sending getblock interests one by one. We have described the update of status with sync data exchange before and let us go into more details. A node sends an interest carrying an individual state digest and the hash of its highest block to the network to inform the latest state of its dataset.A Node providing

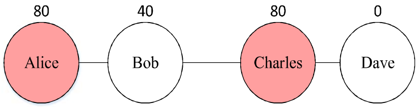

Fig. 5. Updating status between users.

Hash of the previous block	Time stamp	Nonce	Message1	Message2	...

Fig. 6. Block structure.

sync service gets the state of this consumer through the getblock interest name prefix and checks if the name matches with any block in its own blockchain. If the match is successful, it will return all the blocks the requester doesn't have. As a result, the requester gets what it wants in very short time.

Dave is a newcomer and Bob has been disconnected for a long time, both of whom have missed lots of blocks at the current state (Fig. 5). Alice and Charles have the whole current blocks the system has generated. Their blockchains are the same. Dave and Bob send sync interests to the system with their state at present. Due to geographic distance, Charles returns blocks from height 1 to height 80 to Dave while Alice sends back data packet to Bob with blocks from height 41 to 80.

D. Creating a New Block

In order to simplify the implementation, business information will not be carried in transactions like common blockchain systems. A piece of message in a block just contains mathematical expression so that the result which can be checked quickly for other users. Messages are also laid out in linear structure from top to bottom, which is different from Merkle trees in other systems.

A Miner puts some messages, the hash of the highest block in its blockchain and the time into a block according to a common structure [15] shown in Fig. 6. Then it generates a random integer, nonce, and change this value to modify the digest of the whole block in order to produce a hash below the threshold level. The threshold in our system is a fix value with 27 leading zeroes in bits, while the threshold in the real bitcoin system should be constantly updated. A Miner can update time item and continue its PoW work if all the nonce value fail to get a good result. When users receive blocks, they check validity of the PoW and data in blocks sequentially and decide whether to store the blocks.

E. Broadcasting a New Block

Miners should broadcast blocks immediately once they produce PoWs, or else their works and investment will be meaningless. We use the broadcast method of the Chronosync protocol [16] for reference. As illustrated in Fig. 7, every node keeps an outstanding getblock interest to request the next block automatically when it stores a new block. When all parties have the same knowledge about the blocks, the system is in a stable state. In a stable state, one getblock interest can be transmitted over a link at most in one direction. As soon as some party generates a new block, the state changes, and the outstanding interests get satisfied. In Fig. 7, the state change has been caused by Alice's new data which is multicasted to other two users following the PIT entries set up in routers by sync interests.

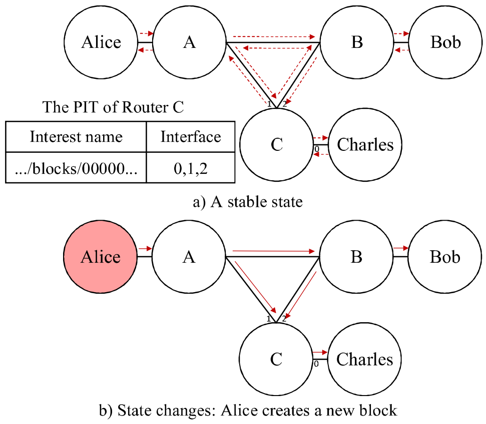

Fig. 7. Broadcasting a new block.

Because of the naming rules, users can get the name of the next block even if it hasn't been created. Requests become substantially easier. A participant only needs to generate the hash of the latest block and sends an interest with name carrying this value. It's also worth pointing out that, different from the Chronosync, users in our system do not construct or maintain digest trees because blockchain is naturally a digest chain and reflects the current state of the owner's dataset.

IV. Discussion and Analysis

In this section we discuss the advantages of BlockNDN compared to the traditional blockchain system in IP network.

A. Difference Between Logical and Physical Topology

In a peer-to-peer system over IP, nodes establish connections with other nodes and the volunteer nodes at the application layer, which form logical topological map. However, messages are delivered along the physical paths of physical network which is different from those links in the logical map. In our system over NDN, information is propagated along the physical paths directly, which is more efficient than the system with supernodes in IP network.

The broadcasting of blockchain system in IP network is complicated. Nodes send messages to every neighbor that they have established the connection with. Their neighbors also follow the same way. In theory, nodes in blockchain are completely homogeneous without any level division, whereas the implementation is contrived. It is difficult for nodes to find others participants online without any prompts. To resolve this problem, bitcoin in IP network introduces the concept of volunteer nodes. The IP addresses of these nodes are recommended to users as default connection options to help to discover neighbors and to send neighboring nodes lists. These volunteers bring a certain amount of risk to the system because we cannot ensure that there exists no fraud among them. Meanwhile, the logical and the physical topology are different between each other, which contributes to poor connectivity and leads to low efficiency of network propagation.

Fig. 8 is an example. The Black lines and the red lines represent the physical paths and the logical paths between nodes, respectively. Nodes with label of *c*, *d* and *f* are supernodes. Take node *e* as an example, it has built a connection with *f* at the very

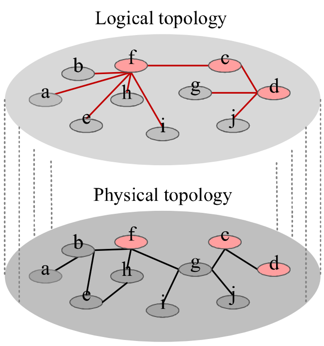

Fig. 8. The logical topology and the physical topology.

beginning and chooses to establish a link with node *a* after node *f* has sent it a list of IP addresses of other users. When node *e* has produced a new block, it will broadcast messages to *a* and *f* firstly and inform node *j* finally when the propagation is done.

Since NDN has no conception of IP address, nodes do not need to exchange information over and over again to establish relationship with others. They just request data by sending interests with name prefixes that carry no identities of participants. Interest packets are forwarded to producers along physical paths.

B. *Complexity of Data Exchange*

Unlike NDN, the TCP/IP protocol does not provide native support for multicasting. As a result, extra work should be done for improving system performance.

To avoid sending block messages to those nodes which have already received them from others, data in IP network are not forwarded directly [17]. Instead their existence is announced to the neighbors by inv messages. The inv message contains a set of block hashes that have been stored or created by sender and then be forwarded to the neighbors. A node, receiving an inv message that it has not maintained locally yet, will issue a getdata message which contains the hashes of the block it needs. Finally, the transfer of the block will finish via individual block messages which are sent by the owner. In sum, two nodes will complete information exchange when they complete sending an inv message, a getdata message and a block message.

In NDN, an interest itself indicates the sender does not have the data at present. The absence of inv messages simplifies the propagation method of the system.

C. *Broadcast Overhead*

In a blockchain system, peers broadcast their blocks and transactions to the others. It is important to add that NDN routers with Content Store are equipped with cache capabilities and are able to increase the network performance.

To evaluate the broadcast overhead in different network architectures, we set up a corresponding scale of evaluation. In the whole system, the cost of the average node of each information propagation is defined as the broadcast overhead in this network architecture, which is shown in (1).

$$\mathrm{c} = \frac{1}{R}\sum_{i=1}^{N} a_i b_i \tag{1}$$

Among the above, R is the total number of nodes which are informed of the new message. N is the number of nodes in the whole network. a_i is the number of forwarding of node i along logical paths. b_i is the average-physical distance for node i, which represents the hops node i takes for forwarding the message to those neighboring nodes according to the logical topology.

Owing to the consistency of logical topology and physical topology in NDN network architecture, the average-physical distance that it takes is 1. A message will be forwarded to all the other neighboring nodes when a node receives this message for the first time. Within (2), the broadcast overhead under NDN network is as following:

$$c_{NDN} = \frac{1}{N}\left(1 + \sum_{i=1}^{N}(k_i - 1)\cdot 1\right)\cdot 3 \approx 3(k-1) \approx \frac{3l}{N} \tag{2}$$

For every new message M, k_i is the degree of node i in the graph G of physical links and it represents the number of adjacent neighbors of node i in the physical map. N is the number of nodes in the whole network. k is the average number of the neighbors of G and l is the number of the edges of G. Let us suppose that all nodes in the system will receive message M eventually.

Similar to gossip spreading, the propagation of information over IP networks can be divided into two phases [17]. Some research has shown that the message overhead of the specific flooding broadcast strategy is lower than gossip spreading strategy [18], so we suppose the broadcast strategy of the system in IP networks is as simple as possible which sends fewer messages than the gossip one. When receiving a new message from a neighbor for the first time, a node will send a new message to the other neighbors in logical topology. Since logical topology and physical topology are different, the broadcast overhead in IP network is as following:

$$c_{IP} = \frac{1}{N}\left(1 + \sum_{i=1}^{N} b_i(k'_i - 1)\right)\cdot 3 \tag{3}$$

For every new message M, k'_i is the degree of node i in the graph G' of logical links and b_i is the average-physical distance of physical topology that node i takes to forward the message to each neighbor in the logical map.

We conclude that the broadcast overhead of the system in NDN or IP network depends on the physical topology and the logical topology. c_{IP} is only lower than c_{NDN} in cases that the connectivity of the physical topology is relatively high. In the real bitcoin system over IP, each user has an average of 32 open connections [17] which is higher than those in physical links. In fact, the message broadcast over IP is more like gossip spreading which costs several times more than flooding broadcast strategy. In general, the broadcast overhead in NDN network is relatively lower than in IP network.

V. IMPLEMENTATION

We implemented the BlockNDN system using Named Data Networking Forwarding Daemon (NFD) [19], a core component of the NDN platform for deploying a system on a computer rather than a simulator, and tested our system on Linux. Every

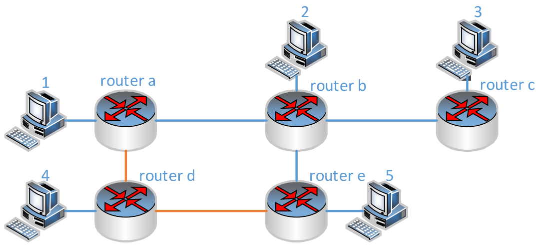

Fig. 9. Each node also act as a miner.

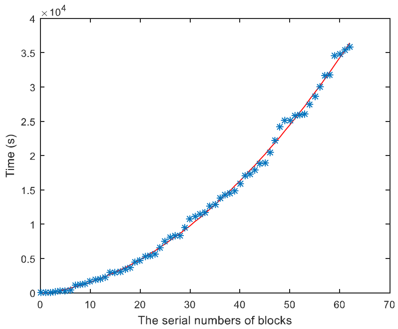

Fig. 10. Time record for the generation of blocks.

user can run the program in proper working order. In our system, each node act as a miner. Nodes kept collecting data and calculating ceaselessly from the very beginning. They also requested new blocks at the same time. After a period of time, we added in node 4 to simulate the join of new members (Fig. 9). The node can update their blockchain in short order.

The difficulty of proof-of-work is configured to a fairly low degree. Miners only need to produce blocks with 27 leading zeros in bits. 27 leading zeros is an empirical value, in which case nodes can dig blocks in a reasonably-tolerable period of time. Within 597 minutes, 62 blocks were created which formed the best blockchain of the system (Fig. 10).

VI. Conclusion and Future Work

In this paper, we present BlockNDN, a bitcoin blockchain decentralized ledger system running on NDN network. Rather than blindly or mechanically copy the design and completion of bitcoin in IP network, we suggested a design method similar to the ChronoSync to achieve decentralization and leverage the advantages of NDN architecture. In BlockNDN, the naming rules imply more specific semantics. A sync Interest packet with senders' state is forwarded to the producer, which gives a more simplified method to update dataset. Getblock data names with digests of the blocks also make retrieval and request more effortless. The resulting design provides more decentralized systems than those over TCP/IP and makes system architecture simpler.

Our future work will focus on performance of the system. We will port the application code to another NDN platform, NDNSim, and conduct comparisons between systems over NDN and IP. We will also try to answer the question that whether our system architecture can reduce the rate of blockchain forks which can lead to substantial divergence within the users. At the same time, improvements need to be made to resist attacks and strengthen system security. It's worth pointing out that users can develop chat applications, audit systems and others based on our system. We also believe that blockchain can offer a new perspective for the progress of NDN in aspects like routing forwarding, namespacing and key management.

Acknowledgment

This work has been financially supported byShenzhen Key Fundamental Research Projects (No.JCYJ20160330095313861, JCYJ20151030154330711 and JCYJ20151014093505032).

References

[1] The IBM blockchain: blockchain overview. [Online]. Available: http://www.ibm.com/blockchain/what-is-blockchain.html

[2] The bitcoin standard client. [Online]. Available: http://github.com/bitcoin/bitcoin/

[3] Satoshi Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Consulted, 2009.

[4] J.A.D. Donet, C. Pérez-Solà, and J. Herrera-Joancomartí, "The Bitcoin P2P Network," The Workshop on Bitcoin Research, vol. 8438, pp. 87-102, March 2014.

[5] Zhang Lixia, et al., "Named data networking (ndn) project," Relatorio Tecnico NDN-0001, Xerox Palo Alto Research Center-PARC, 2010.

[6] Van Jacobson, et al., "VoCCN:voice-over content-centric networks," Proceedings of the 2009 ACM Conference on Emerging Networking Experiments and Technologies, Rome, Italy, December 2009, 2009.

[7] Zhang Lixia, et al., "Named data networking," Computer Communication Review, vol 44, pp. 66-73, July 2014.

[8] Zhenkai Zhu, et al., "ACT: audio conference tool over named data networking," Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking (ICN), Toronto, Canada, August 2011.

[9] A. Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, Steven Goldfeder, Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press, 2016.

[10] PPCoin: peer-to-peer crypto-currency with proof-of-stake. [Online]. Available: http://archive.org/details/PPCoinPaper

[11] M. Castro, B. Liskov, "Practical Byzantine fault tolerance," Symposium on Operating Systems Design and Implementation USENIX Association, vol. 20 ,1999 ,pp. 173-176.

[12] Executive Summary for Financial Institutions: Ripple Solutions Guide. [Online]. Available: http://ripple.com/files/ripple_solutions_guide.pdf

[13] Koshy, Philip, D. Koshy, and P. Mcdaniel, "An Analysis of Anonymity in Bitcoin Using P2P Network Traffic," Financial Cryptography and Data Security, Springer Berlin Heidelberg, 2014:469-485.

[14] Ccnx techincal documentation: Ccnx interest message. [Online]. Available:http://www.ccnx.org/releases/latest/doc/technical/InterestMessage.html.

[15] The Bitcoin Wiki: Block. [Online]. Available: http://en.bitcoin.it/wiki/Block

[16] Zhenkai Zhu, A. Alexander, "Let's ChronoSync: Decentralized dataset state synchronization in Named Data Networking," Proceedings of the IEEE International Conference on Network Protocols(ICNP), Gottingen, Germany. October 2013, 2013.

[17] C. Decker, R. Wattenhofer, "Information propagation in the Bitcoin network," Proceedings of the IEEE Thirteenth International Conference on Peer-To-Peer Computing (P2P), Trento, Italy, September 2013.

[18] Meng Lin, K. Marzullo, and S. Masini, "Gossip versus Deterministic Flooding: Low Message Overhead and High Reliability for Broadcasting on Small Networks," Proceedings of 14th International Symposium on Distributed Computing (DISC), 1970.

[19] NFD: Named Data Networking Forwarding Daemon. [Online]. Available: http://named-data.net/doc/NFD/current