A Personal Information Management Using BlockChain

Yongseon Ji
Department of Computing
Soongsil University
Seoul, Korea
93waves@soongsil.ac.kr

Suhwan Bae
Department of Computing
Soongsil University
Seoul, Korea
shbae0213@soongsil.ac.kr

Yongtae Shin
Department of Computing
Soongsil University
Seoul, Korea
shin@ssu.ac.kr

***Abstract*— Recently, the blockchain was introduced in many existing business models. In this paper, personal information management technology using blockchain is proposed, and only trusted institutions form blockchain networks and manage the flow of personal information through their heads. Agreements are made through rack algorithms, and the adequacy of the technology is analysed by evaluating the stability of the network.**

Keywords—Block Chain, Consensus, Personal Information, Network

I. Introduction

Recently, the blockchain model has been developed into a consortium or private block chain, replacing many existing business models. For example, IBM's Hyperleger was developed as a business consortium model. Eiderium also provides tools for developing consortium blockchain. Managing personal information on the Internet is one of the most important aspects. This paper presents a model in which a personal information management model can be constructed into a private block chain to efficiently manage the history of personal information. Only trusted agencies form a network of blockchains. In addition, to divide access between the ledger and the database, the network is organized into super-node and general node, and consensus is carried out through the Raft Agreement algorithm.[1]
Chapter 2 describes the relevant research in this paper. Chapter 3 describes the structure and behavior of the personal information management model presented in this thesis through the private block chain. Chapter 4 evaluates the network segmentation stability of the block chain using the Raft algorithm. Chapter 5 describes the conclusions of this paper.

II. Ease of Use

A. *Blcok Chain[2][3]*

The block-chain appeared in the Bitcoin proposed by Satoshi Nakamoto and became common. The characteristics of the block chain provide a consistent distributed system with the originator storing the transaction details. The block-chain provides the integrity of the data using the mesh tree of the structure shown in Figure 1 below. When a new block is created, the block contains a hash of the previous transaction history in the header. These structures allow the block chain to verify the gastro-modulation of data and provide the integrity of the data.

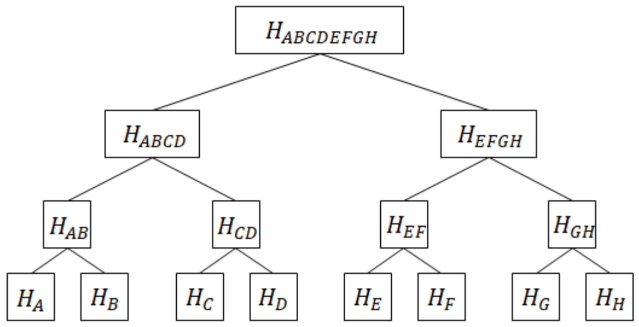

Figure 1. Merkel Hash Tree

B. *Consensus Algorithm*

This section describes the consensus algorithms used in the block chain. A consensus algorithm is an algorithm for each node to achieve consensus on a single result when there is a difference in information between nodes in a P2P-based system such as block-chain.

1) PBFT[4]

PBFT is one of the solutions to solve the Byzantine issue. Simply put, if a majority of errors occur during the settlement process, it is considered to have been successful if a majority of them are agreed. The operating procedure of the PBFT is shown in Figure 2.

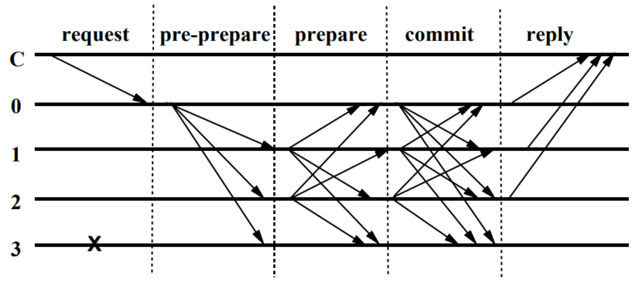

Figure 2. Operational procedure of PBFT

- Request : Client sends block containing transactions to start node.
- Pre-prepare : Send blocks from start node to remaining n-1 nodes.
- Prepare : N-1 nodes receive blocks and then send back to the other n-1 nodes that they received blocks.

- Commit : Each node reviews the results from the other nodes to determine the validity of the block and then notifies n-1 nodes of their final state.
- Reply : Finally, if two-thirds or mode of the nodes have been agreed upon as barbed, then the block will be created and transactions within the block will take place.

Through the above procedure, the PBFT consensus algorithm should be agreed by two-thirds or more nodes. PBFT can prevent attacks if there are (n-1)/3 or less malicious nodes. Therefore, the number of nodes that make up the PBFT is 3,7,11... It is most efficient to have dogs (3n+1). In fact, the block-chains of NEO [reference NEO whitepaper] and COSMOS [reference COSMOS whitepaper] are also composed of four and 100 nodes, respectively. However, PBFT consensus algorithms are inefficient in systems with more nodes, as more nodes often communicate.[Note PBFT whitepaper]

2) *Raft*

This is an agreement algorithm that made Paxos, which is currently the most well-known agreement algorithm, more easily. Consensus algorithms that provide secure CFT (Crash Fault Tolerance) can keep all nodes in a consistent state. The node state transition diagram of Raft is as follows Figure 3.

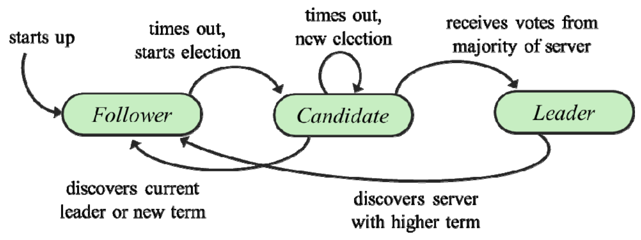
starts up
times out, starts election
times out, new election
receives votes from majority of server
Follower
Candidate
Leader
discovers current leader or new term
discovers server with higher term

Figure 3. State Transition Model For the Raft Algorithm

This is an agreement algorithm that made Paxos, which is currently the most well-known agreement algorithm, more easily. Consensus algorithms that provide secure CFT (Crash Fault Tolerance) can keep all nodes in a consistent state. The node state transition diagram of Raft is as follows Figure 3.

In the Raft consensus algorithm, each node can be Follower, Candide, and Leader. The basic state of the node is the Follower and it shifts from the process of selecting the leader to the candidate state. Raft elects leaders through voting and randomized timeouts. Leaders regularly broadcast their selection as leaders. When a leader is elected, the leader plays a role in collecting and executing transactions. Upon request, the leader creates a duplicate of the transaction requested by the Follows, and each Follower sends the response of generating the replication of the transaction to the Leader and receiving the transaction. When the Leader receives a response from more than half of the Followers, he actually executes the transaction and sends the results to the client requesting the transaction. The leader then notifies the other nodes of the Commit and other nodes also conduct transactions. As such, Raft provides Crash Fault Tolerance through three states of node.[Note Raft whitepaper]

3) *Key-Value Stores*

The Key-Value Store is constructed as shown in Figure 4 below. The Key-Value Store is a NoSQL-type database that cannot manipulate various data than the SQL database, but has the advantage of accessing data at fast navigation speeds. The key value is hashed using the hash table to store the key-Value pair in the table. The hashed key values act as the location of the database and can access the value. In addition, the Key-Value Store uses an auxiliary hash to prevent a hash function from colliding.

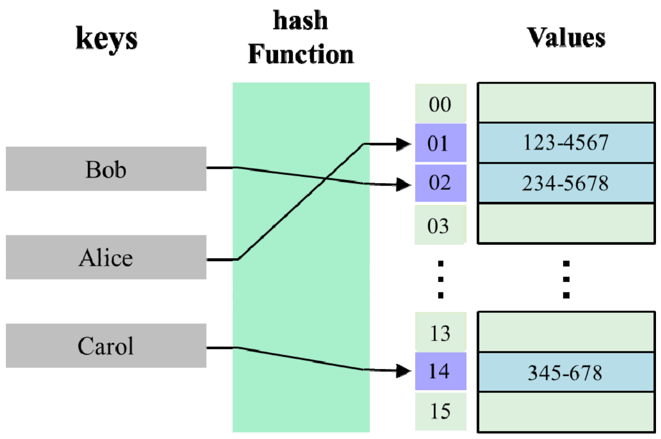

Figure 4. Key-Value Database Using Hash Function

III. A PERSONAL INFORMATION MANAGEMENT USING BLOCK CHAIN'S ARCHITECTURE

This chapter describes the components of the Personal Information Management System using the Block Chain. The system describes the functions of the Director, Node, Agreement Algorithm, and Smart Contact.

A. *Block Format*

Blocks in the block chain of proposed systems consist of headers, metadata, and payloads of blocks. The headers of the block include the number of blocks, the hash of the previous blockhead, the hash of the previous transactions, and the hash of the previous database. The payload includes a hash list of the transactions performed. Metadata includes basic block information and time to create blocks. Figure 5 shows the proposed block format.

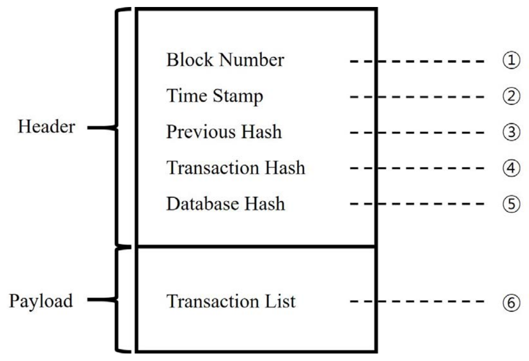

Figure 5. Block Format

- The number given in the order of the creation of the block.
- Indicates the time the block was created.
- The hash value of the previous block head. This value can be used to check whether the block is gastric modulation. It can prove integrity.
- Indicates the hash tree root value of the transactions in the current payload. It serves to validate current transactions.
- Indicates the root value of the current database hash and the state of the database at the time of the block creation.
- Indicates the hash values of transactions executed after the previous block generation.

B. *Structure of database*

The database consists of a database's Key-Value store and a database hash tree. The Key-Value store becomes the key value of the user's hashed identifier and the key value provides access to the value of the user's personal information.

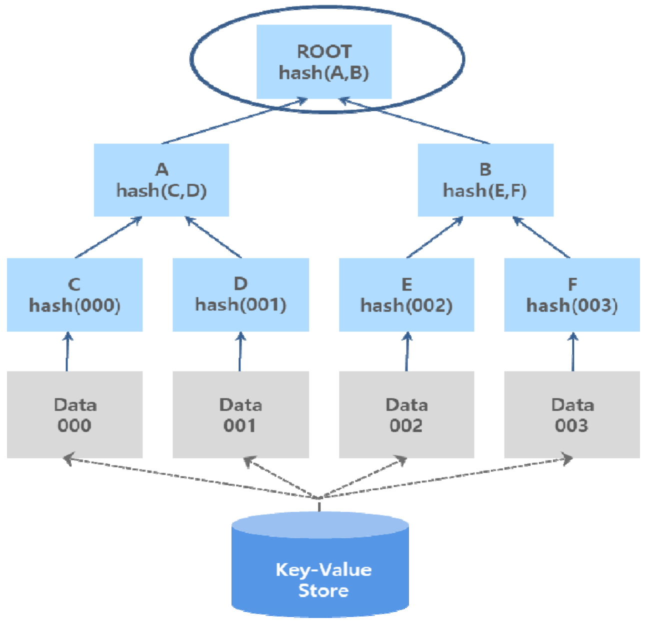

Figure 6. Hash Tree Using Key-Value Store

The database includes a hatchet on the database. Each node in a hash is a value that harms each key-value pair. When data in a database is created, deleted, and changed, a hash of the corresponding data is created and a new one is created. Even if the database changes frequently, it can quickly create a new one by using the characteristics of the hatchet. The root value of this has the status of the current database and is included in the header of the block when the block is created. Figure 7 shows this in pictures.

C. *Configuration of Node*

The proposed privacy management system consists of a private block chain. Therefore, it consists of a group of trusted participants. The proposed configuration of a personal information management system consists of a super-node and a general-node with all information

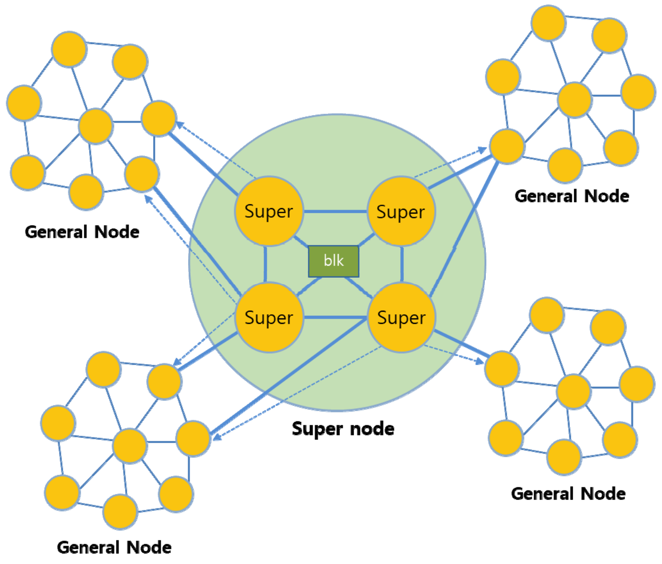

Figure 7. Super Node and General Node of Network

① Super Node: Super node is a node that has a list of original and contracted smart contacts. Super node actually performs transactions and updates the source.

② General Node: The general node has only a signed list of smart contacts. The normal node uses the contracted smart contact to request transactions from the client.

D. *Consensus Algorithm*

Blockchain systems that do not require mining use consensus algorithms to address the Byzantine issue or Crash Fault problem. The problem with General Byzantine causes a malicious node to malfunction and cause network confusion. Crash Fault problems cause normal nodes to fail correctly, causing network confusion. The block-chain system proposed in this paper consist of a private block-chain.

The biggest feature of a private block-chain is that only trusted participants constitute it. Therefore, it is more important to consider the Crash Fault problem than to consider the Byzantine issue. Therefore, a personal information management system consisting of the private block chain proposed in this paper is used as an agreement algorithm based on the Raft algorithm. The Raft algorithm is a leader-based algorithm, and all changes occur through the leader. Only super nodes construct the system through Raft algorithm, select the leader using randomized election, and maintain consistent status of all super nodes through consistent applications.

E. Smart contract Fuction and Operation

In order to manage personal information in the database, basic functions of generation, modification, deletion, and reading are required. The system in this paper provides these functions using the smart contract. The figure below shows the basic operation of smart contract according to the client’s request.

The figure below shows the basic operation of smart contact at client request.

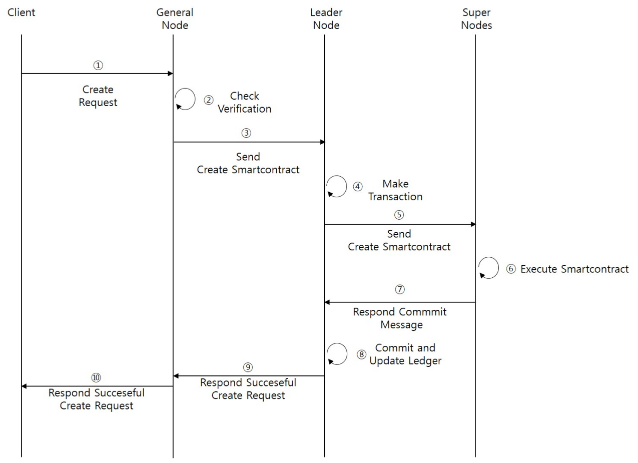

Figure 8. Operation Sequence of Smart-contract

Client's service use request comes in.

① The general node confirms the identity of the client.

② The general node sends the smart contract that matches the client's request to the existing leader node.

③ The leader node verifies the received smart contract and generates a transaction.

④ The leader node sends the transaction to all super nodes.

⑤ The super nodes receiving the transaction execute the transaction.

⑥ The super nodes send a commit completion message including the execution result of the transaction. At this time, all entities of the super node are not updated.

⑦ When a leader node receives a commit completion message including the same transaction result to a majority of super nodes, the leader node also commits, updates the ledger, and creates a block.

⑧ The super node propagates the new block and sends a message to the requesting general node that it has successfully performed.

⑨ The general node receives the message to the super node and sends a message that it has completed the client's request.

The following describes the entity changes by function via smart contact when requested by the client

First, Generation function. The first time a client's personal information is registered, new data must be generated in the database. When a general node receives a request for generation, Smart Contract is executed and a generated transaction is added to the leader node's transaction list. When transactions are executed in the reader node, new data is generated in the database. It also creates new hash tree for transactions and for databases.

Second, Reading Function. Used when clients need to access personal information. When a normal node receives a request for modification, Smart Contract is executed and an open transaction is added to the leader node's transaction list. The database is read when transactions are executed in the leader node. It also creates hash tree of transactions. The leader node transmits the corresponding personal information to the requesting general node. The general node provides the personal information received to the client.

Third, Modification function. Used when modification of client's personal information is required. When a normal node receives a request for modification, Smart Contract is executed and a modified transaction is added to the list of transactions of the leader node. The personal information requested in the database is modified when transactions are executed in the leader node. It also creates new hash tree for transactions and for databases.

Last, Delete function. Used when the client's personal information needs to be deleted. When a normal node receives a request for modification, Smart Contract is executed and a modified transaction is added to the list of transactions of the leader node. When transactions are executed in the leader node, the personal information requested in the database is deleted. It also creates new hash tree for transactions and for databases.

IV. Evaluation

In this section, evaluates the performance of the proposed system. We propose the stability of the system through the network partition probability through the Raft algorithm.

A. Reliability performance evaluation for network segmentation [5]

Crash Fault problems are more important than the Byzantine Fault problem in a private block chain consist of trusted participants. Therefore, it suggests performance for network segmentation probabilities. In the above paper, we evaluate the network partition probability using

The number of timeout counters, K which is the time until the follower node loses the leader and becomes a candidate, the number of total nodes N and the packet loss probability p. The cumulative distribution function and probability density function shown in the paper above demonstrate that network segmentation possibilities are highly dependent on p and K. The probability density function based on the size of the network also showed that larger networks were more reliable for network segmentation than smaller networks. [Paper] simulated network segmentation time according to network size and probability of packet loss.

Figure 9 shows the results of a network segmentation time simulation according to the size of the network. The results above show the stability of the larger network for higher network segmentation time.

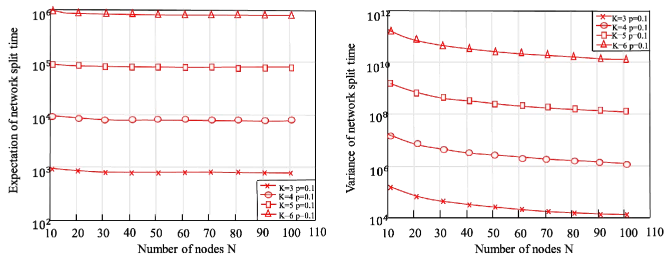

Figure 9. Network Split Time Given Different Network Size

Figure 10 shows the results of a network segmentation time simulation according to the probability of packet loss. The above results showed that adding one leader's election would extend the network stability time by 10 times.

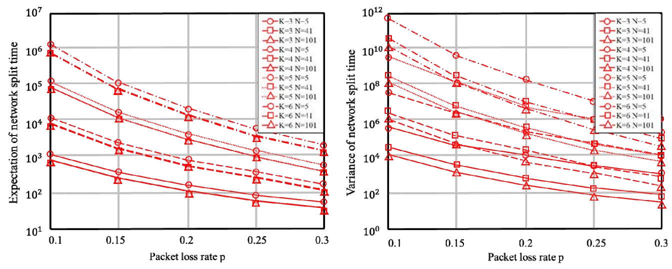

Figure 10. Network Split Time Given Different Packet Loss Rates

[5] tested the stability of network segmentation according to parameters such as packet loss probability, election timeout, and network size. Through this simulation, it can be seen that the probability of network segmentation caused by packet loss is lowered by increasing the election timeout. Therefore, if we increase the Election timeout of the super nodes in the block chain proposed in this paper, we can reduce the possibility of network segmentation and show stability.

V. CONCLUSION

Block chains can replace many of the existing systems. Existing personal information systems are often deployed by sending logs to servers and storing them in distributed databases. The systems presented in this paper presented a personal information management model using a private block chain. The characteristic of a private block chain is that only trusted agencies can participate to form a secure block chain. Super-nodes, which are trusted institutions, create personal information, generate requests for reading and deleting personal information in transactions. It then validates the transaction with consensus algorithm. The history of the logs is also verified by all super-nodes through the muckle hash and created blocks. These procedures verify the gastric modulation of the logs. In this thesis, we evaluated the network segmentation stability of the model's consensus algorithms through Chapter 4. Therefore, through the models presented in this paper, personal information management can be managed efficiently and safely.

ACKNOWLEDGMENT *(Heading 5)*

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP)(201817221336).

REFERENCES

[1] Diego Ongaro, John Ousterhout, “In Search of an Understandable Consensus Algorithm(Extended Version)”,2014

[2] Satoshi Nakamoto, “Bitcoin:A Peer-to-Peer Electronic Cash System”,2009

[3] Vitalik Buterin. “Ehtereum White Paper A Nect Gerneration Smart Contract & Decentralized Application Platform”. 2014.

[4] Muguel Castro, Barbara Liskov, “Practical Byzanine Fault Tolerance”, 1999

[5] Dongyan Huang, Xiaoli Ma, Fellow, “Performance Analysis of the Raft Consencsus Algorithm for Private Blockchains”, 2018