JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 25, No. 11: 1676~1686, Nov. 2021

블록체인 상호호환성 메커니즘

쩌우칭[1] · 이영석[2*]

Blockchain Interoperability Mechanism

Zhou Qing[1] · Young-seok Lee[2*]

[1]Ph.D Student, Department of Information & Communication Engineering, Kunsan Natioal University, Kunsan, 54150 Korea
[2*]Professor, Department of Information & Communication Engineering, Kunsan Natioal University, Kunsan, 54150 Korea

요 약

본 논문에서는 모듈화, 추상화, 계층화 개념을 기반으로 블록체인 상호호환성을 제공하기 위한 크로스-체인 개념의 해결방안을 제안한다. 컨센서스 알고리즘과 특정한 응용 로직으로부터 크로스-체인 기능을 분리하며, 크로스-체인 운영의 타당성과 합법성을 보장하기 위해 머클 증명을 활용한다. 또한, 동종 블록 체인과 이종 블록 체인의 기본 구현이 다르기 때문에 크로스-체인에서는 이를 분리하여 다루기로 한다. 동종 블록 체인의 경우 TCP와 유사한 크로스-체인 전송 프로토콜(CCTV)을 제안한다. 이종 블록 체인의 경우 크로스-체인 기능을 실현하기 위해 릴레이 체인을 구성하는 방법을 제시한다. 제안된 방식은 크로스-체인 데이터의 정확하고 효과적이며 신뢰할 수 있고 질서 있고 시기적절한 전송을 가능하게 할 수 있다.

ABSTRACT

In this paper, we propose a general cross-chain solution based on the idea of modularity, abstraction, and layering, which decoupling the cross-chain function from the consensus algorithm and specific application logic, and utilize a Merkle proof to ensure the validity and legality of cross-chain operations. Since the underlying implementations of homogeneous and heterogeneous blockchains are different, we treat them separately. For homogeneous blockchains, we suggest a TCP-like cross-chain transport protocol (CCTP). While for heterogeneous blockchains, we present a method to construct the relay chain to realize the cross-chain function. The proposed scheme can enable the correct, effective, reliable, orderly, and timely transmission of cross-chain data. However, the essential difference between the operations within a single blockchain and the interoperability between different blockchains is that the trust domain is different. Cross-chain interoperation itself breaks the completeness of the blockchain, therefore, some efficiency and safety must sacrifice to trade-off.

키워드 : 크로스-체인, 머클 증명, 경량 클라이언트, 유효성 증명

Keywords : Cross-chain, Merkle proof, Light client, Validity proofs

Received 7 October 2021, **Revised** 11 October 2021, **Accepted** 16 October 2021

*** Corresponding Author** Young-seok Lee(E-mail : leeys@kunsan.ac.kr, Tel : +82-63-469-4695)
Professor, Department of Information & Communication Engineering, Kunsan Natioal University, Kunsan, 54150 Korea

Open Access http://doi.org/10.6109/jkiice.2021.25.11.1676 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. Introduction

Blockchain technology as a trusted machine participating in the credible society construction has been universally accepted after more than ten years of evolution and development. With the increasing maturity of blockchain technology, the application scenario is increasing exponentially. However, the reality of society consists of many different industries and economic fields. It is not realistic to build all the business in diverse areas on one single blockchain, which is more practical for each industry or field to establish its economic circulation systems separately.

While blockchain has brought us great convenience, it has also encountered some obstacles. Of all the barriers to the blockchain, cross-chain challenge constrains the development space to the greatest extent. So far, there is no universal solution for the exchange of assets and arbitrary data transfer between different blockchains, which is because cross-chain operations are complex to implement due to the closed nature of blockchains.

Based on the current mainstream blockchain platform and existing cross-chain technology research, in this paper, the requirement details to achieve cross-chain interoperability are further analyzed and summarized, and then puts forward a common cross-chain scheme for isomorphic blockchain and heterogeneous blockchain respectively. Based on this scheme, developers can implement the cross-chain logic, and users can build their cross-chain modules and define a series of interoperation protocols. We can also perform cross-chain interoperation as simple as a local function call. Although this scheme can realize the accurate, orderly, fast, and verifiable transmission of cross-chain data from the source chain to target chain, the implementation process is not easy.

Ⅱ. Related Work

In the early stages of cross-chain technology development, asset transfer represented by Interledger Protocol[1] and BTC Relay[2] received the most attention. Recently, cross-chain infrastructures have occupied most of the stage, of which PolkaDot and Cosmos are the most influential two projects.

In May 2013, Herlihy proposed a concept of atomic swap on the BitcoinTalk forum[3], whereby sub-transactions occur either simultaneously or not, without a third state. This technology has evolved into a popular cross-chain technology called hash locking.

In October 2014, Blockstream put forward the concept of a sidechain for the first time, which uses a two-way Peg sidechain mechanism to realize the transfer of crypto assets between the main chain and side chain following an exchange rate [4]. Sidechains allow bitcoins to effectively move from the Bitcoin blockchain to the sidechain and back and allow new features to test on the side chain. The sidechain is the first cross-chain technology that has a giant impact.

In 2015, Bitcoin proposed a Lightning Network [5] that uses hash time locking technology to increase its transaction rate through off-chain micropayment channels. The Hash Time Locking mechanism implements a fast transaction channel under Bitcoin.

In 2016, the BTC Relay scheme [2] was released, and one-way cross-chain communication from Bitcoin to Ethereum was implemented based on a relay chain scheme. In the same year, Vitalik Buterin published an article titled Chain Interoperability, which made a comprehensive and in-depth analysis of blockchain interoperability issue [6].

In 2017, PolkaDot [7] and Cosmos [8] first proposed to build a blockchain cross-chain infrastructure platform that supports compatibility with all blockchain applications. Fig.1 lists the historical events of the development of blockchain interoperability technology.

In 2021, Cosmos finally activated the Inter- blockchain communication protocol that fosters interoperability between Cosmos and other blockchains, while solving scalability issues via shard mechanism.

Today, the Cross-chain Messaging Passing Protocol

of PolkaDot is still under construction, which will truly enable arbitrary value transfer between different blockchains.

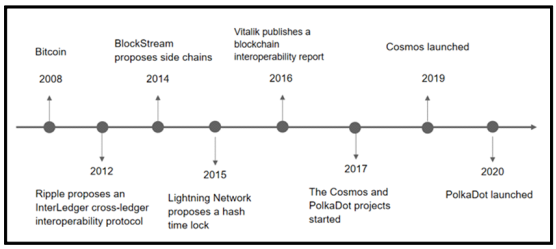

Fig. 1 A timeline of the development of blockchain cross-chain technology

Ⅲ. Theoretical background

Blockchain uses two important hash data structures, the hash list and the Merkle tree, to ensure that data cannot be tampered. In this section, we will briefly describe the two data structures and their common usage. These two data structures are also critical when designing a cross-chain interoperability solution.

3.1. Light Client

In the context of blockchain, a client is a piece of software that connects to other clients in a point-to-point manner, which will communicate in a network, and each client is a node, so in some contexts, the client also uses nodes instead. There are two types of nodes: full node and light node, of which full node is responsible for verifying and relaying transactions and blocks on the network. However, downloading and verifying the blocks of the entire chain consumes time and resources. On some devices such as mobile phones with limited resources, a light client adopts to indirectly interact with the blockchain network by connecting full nodes. The light client does not need to run all the time, nor does it need to read and write a lot of information to the blockchain. The light node interacts with the full node with minimal trust and is primarily used to validate payments. The light node's block header contains information about the Merkle Tree Root, which is like a digital fingerprint of the account balance in the blockchain and all this information stored in the smart contract. The fingerprint changes when any part of the information changes. Assuming that the majority of miners are honest, the block headers and the "fingerprints" they contain can be considered valid. The light client requests some information from a full node, such as the balance of a specific account. Since the light clients know the "fingerprints" of all blocks, they can verify that the answers given by the full nodes match the "fingerprints" they have. It is a powerful tool that can be used to demonstrate the validity of information without prior knowledge.

3.2. Merkle Proof

The hash of all the transactions in each block was organized in a Merkle Tree-like structure whose Merkle root stored in the block header. The Merkle tree is used to summarize all transactions in each block. Each leaf node is the hash of transaction information, continuously merging the two adjacent hashes into a string and then hashing until only one node at the top, the Merkle root, is deposited in the block header. If there is only an odd number of transactions that needs to be summarized, the final transaction will be copied to form an even number of leaf nodes, thus, a balanced binary tree is constructed. With the Merkle tree, any branch can validate some data effectively.

A Merkle proof is a subsection of a Merkle tree used to prove a given piece of data is a part of a Merkle tree. The process of Merkle proof is constantly hashing together the corresponding hash of the hash and climbing the tree until obtain the root hash, and then compared to the known root hash. Merkle proofs are usually used to decide upon the following three factors:

(1) Whether an input value exists in a Merkle tree;
(2) Prove that a data is part of a dataset without storing all the data in the dataset;
(3) Prove the validity of the data sets contained in large data sets without disclosing the complete data set or

its subset.

Similarly, we can concisely prove the membership of a particular block of data in a tree with a hash root. For example, if you want to prove that an MHT tree with a hash h contains data, they only need to provide a validator for the block and a series of internal nodes to recalculate the root and compare it to the root h provided.

3.3. Simple Payment Verification

The nodes in the blockchain category consist of full nodes and light nodes. The light node only stores the block header or part of the transaction data, while the full node stores all the block header and transaction data. Some devices like mobile phones cannot store all the data due to the limited storage capacity, and light clients are required to verify the payment on a light node.

During the process of verifying payments, the light node needs to request the hash sequence from the hash of the transaction itself along the Merkle tree to the Merkle tree hash root stored in the block header, through an RPC message to the adjacent full node to confirm the existence and correctness of this transaction. Confirming a transaction in a block of N transactions requires only a hash value of log2(N) bytes [9].

Simple Payment Verification refers to payment verification rather than transaction verification, where the two terms are different. Payment verification only needs to verify whether the transaction is confirmed, where transaction verification needs to check whether the transaction meets certain conditions, such as whether the balance is sufficient, whether there is double -spending, and so on. Only if these conditions were met can the transaction pass verification. The goal of SPV is to verify that payment is authentic and how many confirmations it can obtain.

The detailed verification process is as follows:

(1) Calculate the transaction hash value that needs to be verified for payment.
(2) The node acquires and stores all blocks of the longest chain from the blockchain network to local.
(3) The node obtains the Merkle tree hash verification path for the payment to be verified from the blockchain.
(4) Based on the verification path, the root hash value of the Merkle tree be calculated and the result is compared to the root hash value of the Merkle tree in the block in the region.
(5) If consistent, the payment is authentic and valid.
(6) Check the confirmation number received for the transaction according to the location of the block header.
(7) In the blockchain, each block has a Merkle tree, and the leaf node stores the hash value of transactions in the blocks, and each layer recursive can get the Merkle root values. The Merkle root value is stored in the block header to summarize and quickly verify all transaction data in the block.

3.4. The blockchain trilemma.

The trilemma of blockchain refers to a public blockchain system that is impossible to satisfy decentralization, safety, and high performance of the three requirements at the same time Fig. 2. Blockchain must make an optimized tradeoff between the three dimensions.

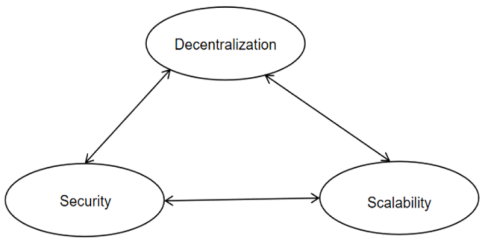

Fig. 2 The blockchain trilemma

Decentralization refers to the decentralized configuration of a blockchain network, and it is often associated with geographic location, connection patterns, and synchronization of network nodes.

Security refers to the consensus security and anticensorship capability of the blockchain in the presence of malicious nodes. The two concepts correspond to safety and liveness in the conventional

distribution consensus.

Scalability refers to multiple aspects: blockchain systems need to maintain security and efficiency as transaction throughput increases and network size grows.

Ⅳ. Blockchain Interoperability Solution

Cross-chain interoperability refers to the transmission of information, data, and assets between two relatively independent blockchains. As a matter of fact, we have no approach to transfer assets from one chain to another, assuming that the cross-chain transaction initiation chain is A, the destination chain is B. In fact, these assets on blockchain A will not be sent directly to chain B. The essence of cross-chain is to lock a certain number of assets on chain A and then issue an equivalent asset replacement on chain B, and when redemption call smart contract was deployed to destroy this alternative asset, the source blockchain A release previously locked assets, then the difficulty of realizing cross-chain lies on the interoperability of cross-chain messages.

4.1. Basic requirements to achieve blockchain interoperability

From the concept and requirements to understand interoperability, the essence of cross-chain is to transfer the message M on chain A safely and reliably to chain B, meanwhile reach the expected effect on chain B. A successful cross-chain scheme should solve the following issues:

(1) Authenticity of message M, that is, the message M indeed exists on blockchain A, and it does come from chain A.

(2) Routing of message M, ensuring that cross-chain messages reach the target chain quickly, accurately, safely, and in an orderly manner.

(3) Proof of the validity of message M, where validity refers to whether message M is still valid when it reaches the B chain, which covers mainly three aspects: whether the transferred assets where locked on blockchain A, no double-spending exist, and the state does not change during the data transmission period, etc.

(4) Execution receipt of message M. The source blockchain needs to confirm whether the cross-chain interoperability is successful after executing the cross-chain transaction. A confirmation message should return from the source blockchain to the sending blockchain.

(5) Cross-chain transactions did occur on the target chain, and they occured only once.

To address these critical issues, we aim to establish a standard cross-chain solution, just like the TCP/IP protocol for the Internet. The initiative blockchain and response blockchain of cross-chain transactions should at least support these functions as follows:

(1) Both the source chain and destination chain have a queue pair, whose output message queue will be used to handle outgoing messages, and the input queue records incoming messages.

(2) The source chain needs to provide proof of authenticity for cross-chain messages. That is to say when sending cross-chain messages, evidence of authenticity is also required. This proof requires building a Merkle tree for cross-chain messages and including Merkle root in cross-chain messages or cross-chain blocks. The validation process is similar to simple payment verification. The receive chain rebuilds the Merkle tree after receiving this message, then compares the two Merkle roots to check whether they are the same. It indicates that this message is authentic. Otherwise, it is a fraud message.

(3) A valid route is necessary for the cross-chain message. The routing process needs to build a unified cross-chain information format, specifying the source and destination blockchains of transactions and the contents of messages. The routing information records on the third-party relay chain, and when the receiving chain validates the cross-chain message, they can look it up on the relay chain. The relay chain achieves indirect synchronization of the source chain

and the target chain.

(4) Proving the validity of cross-chain information is critical. Cross-chain is not only the transmission of information, and the more meaningful part is the settlement system. In addition to cross-chain data, evidence of ledger status should also send to the receiving chain to prevent double-spending, and both data and evidence should process in strict order. Value transfers occur between specific accounts, not entire ledgers. At present, the KV database-based account storage methods cannot be used to provide proof of validity. Therefore, blockchain needs to design a new verifiable storage structure to make SPV-like validation more convenient. We can consider the UTXO model and the relay chain Merkle tree.

(5) Proof of cross-chain execution results, that is, receipt or acknowledgment, similar to validity, requires the support of a new data structure and operating algorithms.

Cross-chain can be classified as homogeneous and heterogeneous cross-chain according to the underlying technology. Homogeneous blockchains have the same block structure and consensus mechanism, so the destination chain can understand the transaction on the source chain, unlike transactions within the single blockchain, cross-chain transactions across different trust domains. To verify the validity of a cross-chain transaction, we need both a light client and an additional binding Merkle path. In addition, the block location of the initiation chain that is fixed by the genesis block and block height together needs to be recorded for possible rollback operations. Cross-chain between the heterogeneous blockchains is different due to the block structure, and the consensus algorithm of the source chain and target chain is different. The light client of the destination blockchain cannot verify the cross-chain transaction. Hence, we consider adding a state transfer parameter. Although the block structure of the source blockchain is not uniform, we can standardize a state transfer data structure and record the state change results on the relay chain.

4.2. Cross-chain between homogeneous blockchains

Because the security mechanisms, consensus algorithms, network topology, block generation logic, and verification methods in homogenous blockchain are consistent, cross-chain interoperation is relatively simple compared with heterogeneous blockchain. We can achieve this through a TCP-like handshake protocol by changing the status of cross-chain messages and transaction proofs to indirectly synchronize the state of the source chain and destination chain.

The ledger format in two homogeneous blockchains is the same, and both the initiation blockchain and response blockchain can understand the respective counterparty's transactions. So similar to a single blockchain, we can also use a light client to verify the cross-chain transaction. Although different blockchains have different trust domains, they all support the underlying P2P network, so we can also use light clients to verify cross-chain transactions. We only need to bind it to a connection between the two blockchains.

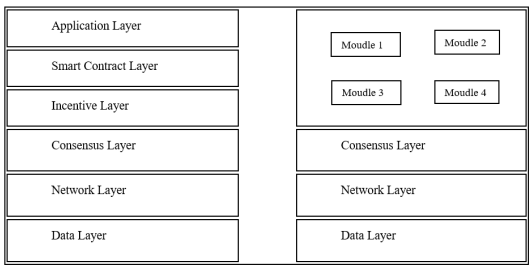

Fig. 3 The hierarchical structure of the blockchain

According to the conventional six hierarchical structure of the blockchain, we can simply abstract blockchain into the application layer and core layer as shown in Fig. 3, and then establish a cross-chain protocol between these two layers.

The core layer of homogeneous blockchains is the same. If we can build a protocol between the application layer and the core layer as the cross-chain infrastructure, and in this way, the users can construct their business logic on the application layer, there is no need to worry about the underlying consensus algorithm and the

implementation of the cross-chain protocol. The upper incentive layer, the smart contract layer, and all the distributed applications can be regarded as an application module, which is similar to the application layer protocol of the Internet. We can allocate each application module a port number, and bind it to the transmission channel, and establish a cross-chain transmission protocol (CCTP) similar to TCP to achieve cross-chain interoperability.

The cross-chain functionality can be modularized to several parts, and each item is compared with TCP as shown in Table 1.

Table. 1 The comparison between CCTP and TCP

CCTP	Purpose	TCP
Port	Each application module or process is assigned a unique port through which the application can call the underlying transport channel to transmit data.	Port
Light Client	Light clients used to track the consensus states of the counterparty blockchain, according to which together with related Merkle proof can verify the validity of coming cross-chain transactions.	None
Connection	Initialize, establish, update and release a connection between two full nodes of the source blockchain and the target blockchain, each connection associated with the client of the counterparty blockchain to verify the validity of the incoming transaction. Here, the connection was between blockchains rather than between websites, which requires the support of TCP protocol.	None
Channel	Initialize, establish, update and release a channel between two connection end nodes, initialize a channel to request memory space on the two full nodes, one connection can establish more than one channel, packets are transmitted orderly over each channel by using a sequence number.	Connection

Light client:

The two participants of the cross-chain transaction first initialize a light client to verify the data transferred from the source chain before sending data, indicating that the chain can verify the validity and legitimacy of cross-chain transactions from the source chain.

Connection:

Before sending data, two blockchains need to establish a logical connection to connect two chains. For simplicity, we can randomly choose two full nodes to connect. Together with the light client, this connection is responsible for guaranteeing the legitimacy of a cross-chain transaction, that is, confirming that a cross-chain transaction did occur on the target chain and that the cross-chain transaction commits only once.

Channel:

Cross-chain transactions transmit over the channel, on which we can use a sequence number to ensure the transactions delivers in order. Each of the channels maintains four queues to manage the two party's outgoing and incoming messages respectively. When initializing, the protocol requests memory space on two connected full nodes, and more than one channel can be established on a single connection.

The connection can establish using three times hand handshake method similar to the TCP protocol, and the release process utilizes four waves. The channel initialization and release process are the same as the connection.

Suppose an account on Blockchain A intends to send a cross-chain transaction to another account on Blockchain B, the data need to be transferred, and the process are as shown in Fig. 4.

(1) The user sends a cross-chain transaction from the source account to Blockchain A, on which the transaction is executed, and locks the appropriate amount of assets.

(2) Write the cross-chain transaction to A's outgoing message queue, whose functions are like a mailbox where all the cross-chain transactions place.

(3) To notify the destination blockchain of the events that occurred on blockchain A, a relay-chain is required to forward the cross-chain transaction CTX in A's outgoing message queue to B's incoming message queue. The relay-chain discovers the cross-chain transaction in A's outgoing message queue and the corresponding Merkle Proof through polling or listening mechanism, and then packages the transaction and proof information together into a CrosschainTxPacket and forwards to blockchain B. Meanwhile, it queries the block information where CTX locates, packages the blockhead information into a CrosschainBlockheaderPacket, and sends it to the blockchain B.

(4) After receiving CrosschainTxPacket, blockchain B will verify and execute these transactions. First, it verifies whether the block header on the A chain passes through the consensus process by validators. Then, it verifies whether the Merkle proof of the

cross-chain transaction in the CrosschainTxPacket is equal to the block header hash in the CrosschainBlockheaderPacket. When all verifications passed, the B chain starts to perform corresponding operations, such as generating assets on blockchain B, returning transaction receipts, etc.

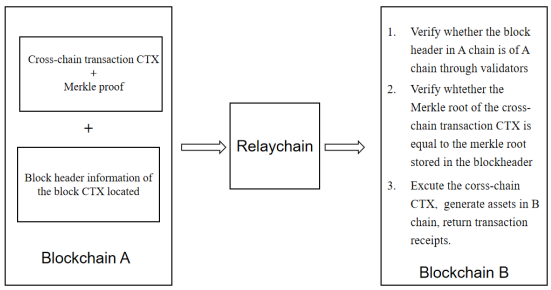

Fig. 4 Cross-chain data for homogeneous blockchain interoperation

4.3. Cross-chain between heterogeneous blockchains

Blockchains based on probability consensus algorithms such as PoW and PoS are significantly different from those based on traditional determinism consensus algorithm BFT in terms that block generation and final confirmation mechanisms. Therefore, it is not easy to design cross-chain protocols between heterogeneous blocks. Generally, a third-party relay chain is needed to assist.

Since there is no blockchain to rely on another blockchain to determine the validity of transactions, interoperability is an inherent contradiction in blockchain, and we have to make some additional assumptions to achieve cross-chain interoperability.

A decentralized blockchain A interoperable with a blockchain B is equivalent to a decentralized blockchain C containing both A and B's ledgers [10]. That is to say, to realize cross-chain interoperation on heterogeneous blockchain equivalence to construct a blockchain covering the source chain and destination chain, this chain is commonly known as a relay chain.

If chain A sends a message M to chain B, a channel needs to initialized between block A and block B to transmit the cross-chain transaction and the corresponding proof information. In addition to the cross-chain message itself, corresponding proof of validity and state transfer suggestions should also be included. Fig.5 identifies all the data that needs to be transmitted across the source and destination chains.

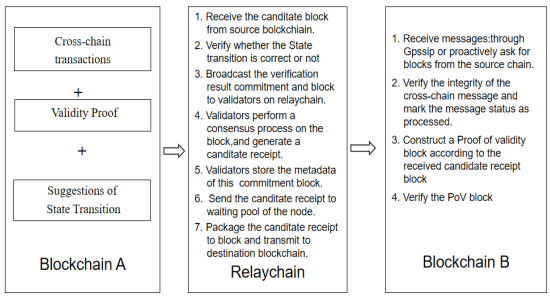

Fig. 5 Cross-chain data for heterogeneous blockchain interoperation

Cross-chain messages are collected by the sending chain and then verified, packaged into blocks, and reach consensus among validators.

When a relay chain validator receives an uncommitted block and proof of validity from the source blockchain, it checks that this block follows the state transition rules of the source chain. It only needs to: verify the list of state transitions carried in the candidate block, the values in the source chain database.

The validator does not need to check each value in the parallel chain state, only the modified value to ensure that the modification is valid. If passed, it signs and broadcasts proof of validity to other validators assigned to the source chain. Once the validator's consensus verification pass indicates it is a valid delivery, they will construct a "candidate receipt."

To synchronize the status of the initiator chain to the relay chain reliably, each block generated by the initiator chain generates a corresponding proof of validity block. The verifier node on the relay chain verifies this validity block through the algorithm provided by the initiator chain. If the verification passed, the block header and related metadata of the initiator chain was written into the relay chain, which is equivalent to the state of the initiator chain synchronized to the relay chain. The proof

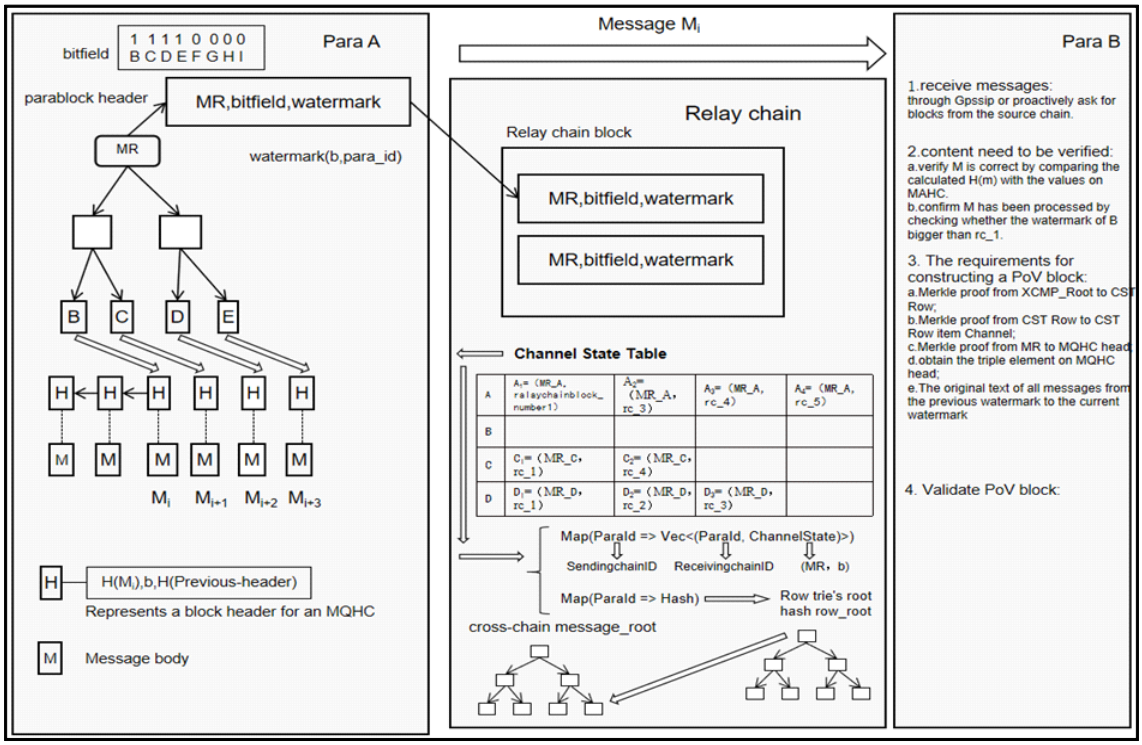

Fig. 6 An overall diagram of cross-chain message access flow

of validity block consists of all transactions, data read when processing each transaction (such as transfer sender balance) and the validity proof, and data written when processing each transaction (such as updating transfer sender B balance) and its proof. In this way, the verifier does not need to store any initiator chain state and only needs to re-execute the transaction.

Considering the implementation of validity proof and state transition verification need Merkle proof, we can organize the hash value of cross-chain transactions into a Merkle tree structure, utilizing a message hash chain that tracks the cross-chain messages. All the message hash chains constitute a Merkle tree, and the path from the Merkle root to the leaf node is a channel. We use the message hash list rather than the message chain because the hash chains have a property, which is when we delete a leaf node, the integrity of the data will not change.

The status of the cross-chain messaging channel is maintained by the relay chain using a two-dimensional table whose row represents the source chain, and the column represents the destination chain. Each item of the channel tracks the latest sender message queue root and the relay chain block number. Fig.6 is an overall diagram of cross-chain message access flow.

Under the method to build a blockchain, relay blockchain also needs to construct its block generation algorithm and consensus algorithm.

On the source blockchain end, all the hash values of the cross-chain message construct a Merkle tree. Fig.6 shows the details. The leaf node of the Merkle tree is a triple (H(Mi), b, H(Previous-header)), where H(Mi) represents the hash value of the cross-chain message Mi, b is the block number of the relay chain at which the parent message emit, which can use as the global time, H(Previous-header) is the hash value of Mi's parent, and the leaf node H represents a block header for the message queue hash chain (MQHC). Each MQHC is a channel for the source chain to the destination chain. All

the message queue hash chains construct a Merkle tree, and the root hash MR is recorded into a block and then send to the relay chain. Watermark is used to record the location of the nearest relay chain block processed on the destination chain incoming queue.

On the relay chain, there are two important data structures: channel state table (CST) and two maps, each CST item is a tuple (MR, relay chain block number) stored in a row, and each row represents the message that comes from the same source chain. These row items were hashed together to construct a row-root, and whenever a single data item changes in a row, its row-root changes, then further construction of the Merkle tree with these row-roots as leaf nodes can fully reflect the state roots of the relay chain.

On the destination chain, the validator node will reconstruct a block according to the original message in the outgoing message queue, block header of hash message chain, the Merkle proofs on relay chain, and the Merkle proofs on source chain. The validators will also verify this block and commit it, and the block header will update to the relay chain. Then, the block header goes up to the relay chain, and the relay chain updates its status. The sending chain also knows that the message it sent has been processed by querying the relay chain. The relay chain updates the state, and the sending chain queries the relay chain to know that the message it sends has finally been processed.

Ⅴ. Conclusion

In this article, we propose a solution to the blockchain interoperability problem, with the goal of allowing data and assets to move between two independent blockchains on demand. Since the underlying implementations of homogeneous and heterogeneous blockchains are different, we propose a general solution framework. These two approaches are totally different. The solution for homogeneous blockchain is a cross-chain transmission protocol similar to TCP protocol, responsible for forwarding cross-chain messages only. The approach for heterogeneous blockchain is to use a secondary blockchain to facilitate the delivery and validation of cross-chain messages. The cross-chain metadata will be stored on the blockchain after reaching the consensus, and the cross-chain messages themselves will directly transfer from the outgoing message queue of the source chain to the incoming message queue. At present, the development of cross-chain technology is still in the initial stage, and there are no cases of large-scale commercial applications. Among the solutions already launched, none of them is perfect, and each of them has defects. The research in this paper is just a beginning, and more details and implementation methods are worth further exploration.

REFERENCES

[1] S.Thomas and E.Schwartz. (2016). A Protocol for Interledger Payments [Online]. Available: https://interledger.org/interledger.pdf.

[2] Consensys. (2016, September). BTC Relay Documentation Release 1.0 [Online]. Available: https://buildmedia.readthedocs.org/media/pdf/btc-relay/latest/btc-relay.pdf.

[3] M. Herlihy. (2018, July). Atomic Cross-Chain Swaps [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3212734.3212736.

[4] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille. (2014, October). Enabling Blockchain Innovations with Pegged Sidechains [Online]. Available: https://www.blockstream.com/sidechains.pdf.

[5] J. Poon and T. Dryja. (2016, Jan). The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments [Online]. Available https://lightning.network/lightning-network-paper.pdf.

[6] V. Buterin. (2016). Chain interoperability. *R3 Research Paper* [Online]. Available: https://www.r3.com/wp-content/uploads/2017/06/chain_ interoperability_r3.pdf.

[7] G. wood. Polkadot. : Vision for a heterogeneous multi-chain framework draft 1 [Online]. Available: https://polkadot.network/PolkaDotPaper.pdf.

[8] C. Goes and I. GmbH. The Inter-blockchain Communication

Protocol: An Overview [Internet]. Available: https://ibcprotocol.org/documentation.

[9] R. Merkle, “A Digital Signature Based on a Conventional Encryption Function,” *LNCS*, vol. 293. pp. 369-378, Aug. 1987.

[10] P. Lafourcade and M. LombardPlatet, “About blockchain interoperability,” *Information Processing Letters*, vol. 161, pp. 105976, Sep. 2020.

쩌우칭(Zhou Qing)

2004년 7월 Central China University 수학과 학사
2010년 7월 Beijing University of Technology 컴퓨터과학과 석사
2018년 3월 ~ 현재 국립군산대학교 전자정보공학부 박사과정
※관심분야 : 블록체인, 정보보안

이영석(Young-seok Lee)

1992년 2월 충남대학교 컴퓨터공학과 학사
1994년 2월 충남대학교 컴퓨터공학과 석사
2002년 2월 충남대학교 컴퓨터공학과 박사
1994년 2월 ~ 1997년 2월 LG전자 연구원
2002년 3월 ~ 2004년 8월 한국전자통신연구원 선임연구원
2004년 9월 ~ 현재 국립군산대학교 컴퓨터정보통신공학부 교수
※관심분야 : 정보보안, 모바일컴퓨팅