USECHAIN

Global Mirror Identity

Blockchain Ecosystem

Technical Whitepaper

Summary

Since 2009, the decentralized blockchain industry and technology have had a growth spurt. Bitcoin and Ethereum[1]have respectively brought brand new concepts such as "decentralized currency" and "smart contract" to the blockchain world. The blockchain technology has enabled the possibility for society to manage itself. However, in the current state of affairs, all the widely used public chains are based on anonymity, which makes digital crypto-currencies tools for money laundering, smuggling, illegal fundraising, and other illegal and criminal activities. In the meantime, because of low performance and efficiency, and bad scalability, the largest usage still only remains in digital crypto-currency trading. In anonymous blockchains, the main body is addresses, and in society, the main body is natural persons. Therefore, for the implementation of scalable applications, there must be a mirror identity protocol between addresses and human beings.

Usechain is dedicated to developing a public chain based on mirror identity, establishing an ecosystem based on identity public chain, promoting the implementation of all sorts of blockchain applications that links blockchain addresses to verified real users, and realizing close integration between social credit, infrastructure, and commercial use. Through the Mirror Identity Protocol(MIP), based on zero-knowledge proof, we can realize separation of identity certification from identity

information, enabling the identity chain to have the same privacy and security level as anonymous public chains, while realizing the mirror identity between the address on the chain and the identity of persons off the chain. By fully utilizing the mirror identity information, on the one hand, we can significantly improve currently existing blockchain algorithms such as network sharding algorithms, consensus algorithm, virtual machines and so on, and on the other hand, we can provide functionalities that existing anonymous blockchains do not have, such as one-person-one-vote, address penalties, etc., elevate the procedure performance of the blockchain itself, enable the blockchain itself to handle commercial usage on a large scale, and wield enormous commercial value.

This document will focus on Usechain technical architecture, key technological principles, and technological protocols.

Contents

1. Overview

The Mirror Identity Blockchain (which we shall call "Identity Chain") Project is dedicated to providing a public chain open towards the verification of real users, while fully protecting personal privacy, keeping the information on the chain fully transparent, and accord every address on the chain to a natural person in the society. In the identity verification module, we balance the professionalism of third-party certification agencies and the decentralization of the network structure. In the meantime, we use the main and sub addresses, ring signature, and the identity data encryption technology to protect users' privacy, which means only under the condition that the user authorizes could all the address information of this natural person be obtained.

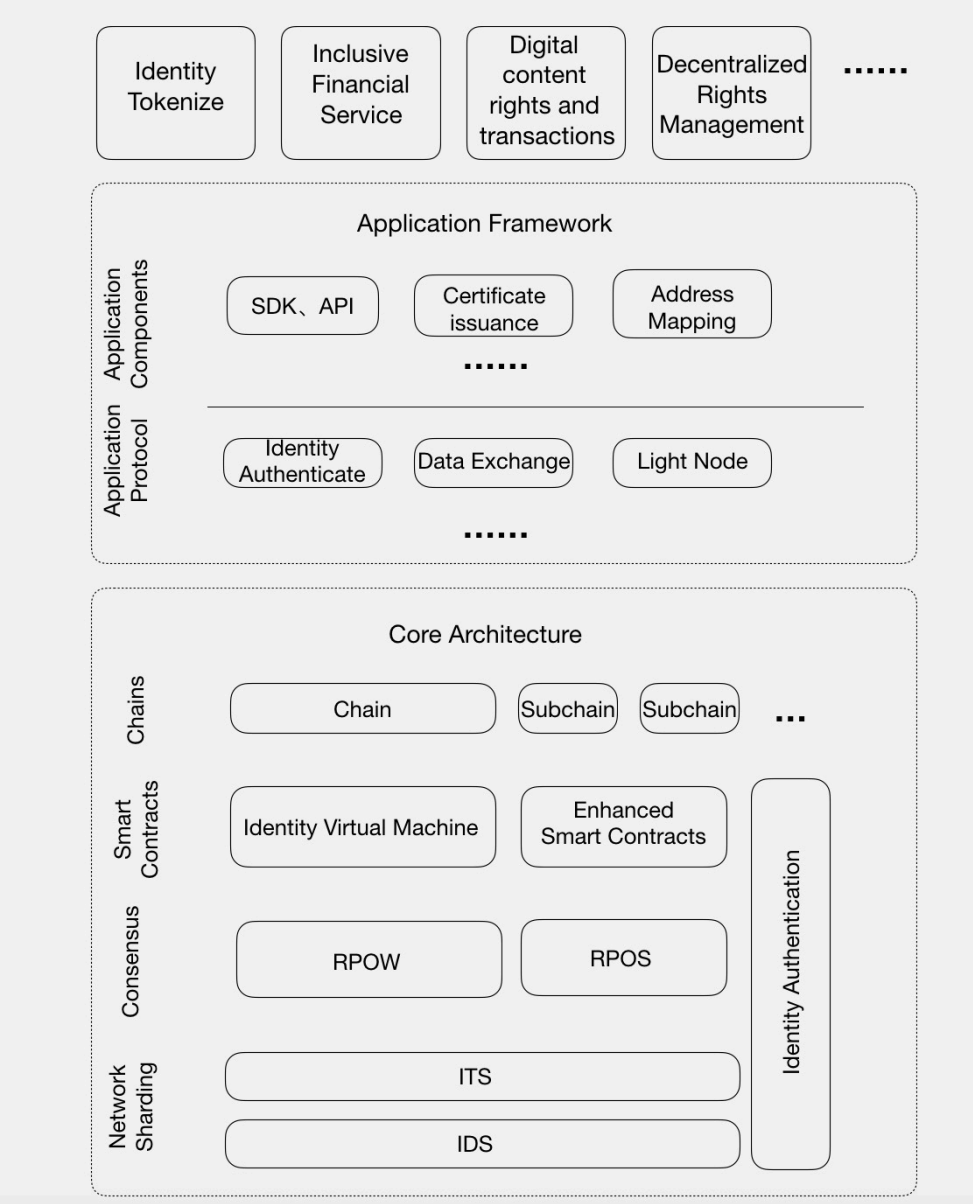


Fig 1-1 Usechain Structure Model

On top of the advantage of privacy protection by fully utilizing the real identity ledger, we re-construct the base plan of the blockchain, thereby significantly improving the performance of the system. This project will develop the application side combined with the specific commercial scenarios, while realizing commercial data and resource sharing, in order to construct a complete identity chain ecosystem.

In terms of structural design, Usechain can be simply divided into

application service, application framework, and the base public blockchain system. There’ re 3 major innovations with the identity chain based on identity information:

1) On the basis of current blockchain technology, we’ re the first to propose the Mirror Identity Protocol (MIP), and establish the multi-level accord system between the address on the chain and verified real users. For the ordinary user, every address is verified and has relatively high credibility, without any other information being divulged, and therefore fully protecting privacy. For all the nodes on the blockchain management committee, on the condition that every node votes to agree, they can obtain every address associated with the same verified real user (and thereby obtain all the transfer information on all the addresses). The verification process of real users is completed by multiple third-party certification agencies at the start, the accord between the real social ID information and the address on the chain requires the consent of both on-chain decision committee and multiple third-party agencies (providing government institutions a way to trace everything).

2) Using the Mirror Identity Protocol as the cornerstone, in the meantime, Usechain proposes technological and design innovations on multiple levels. On the one hand, we use a new consensus mechanism and sharding plan, to improve the scalability and speed of the network, thereby lowering transaction cost. The items the project will first consider are mainly Randomized Proof of Work (RPOW) based on software algorithm and network sharding/identity sharding, and we will eventually achieve a hardware-based Randomized Proof of Work (RPOW) consensus algorithm that has low cost and high efficiency. On the other hand, we’ ll introduce the Identity Virtual Machine (IVM). The IVM is the new standard for high performance smart contracts, easily defined and sandboxed through low adaptation. The identity chain will build the communications layer in between, enabling the IVM to interact and even directly code the blockchain base, outside APIs, and subchains. In this way, we make up for shortcomings in the Ethereum virtual machine, construct highly efficient smart contracts that can

interact with outside data, thereby expanding the span of application scenarios. The IVM will have built-in detection algorithms for smart contract loopholes, examining the contract while it is being compiled, realizing a secure smart contract system.

3) Usechain achieves a well-designed community governing system, in order to realize the consensus process for people' s subjective problems. First, Usechain has achieved a complete voting system, able to realize a fair one-person-one-vote management system. When we discover contracts accidentally malfunctioning, hacking, and other malicious acts, the community can vote to decide whether we track down the culprit. Furthermore, Usechain has specified a series of penalties, for when it is necessary to cooperate with governments' law enforcement agencies and investigate the culprit' s legal responsibilities, in order to curb and deal with malicious activities.

The Core of the Identity Chain is the identity, verified by a method that combined centralization and decentralization. With the identity, identity+data→credit. This is different from traditional credit platforms such as Equifax in that the data has no boundaries: all the data applied on the chain is shared across the public chain level, making it a big data base platform that actually has complete user portraits. On the public chain, the currency is one Token (USE), and the Token flows in different applications, which generates data, and data+identity generates credit. The public chain requires more optimized consensus mechanisms to support consumer-level transaction confirmations, the creating tools for smart contracts already operating, and on-chain to off-chain data interface.

And a more thorough DApp based on the identity chain, we call subchain. Every subchain has its own complete identity chain protocol, but is closely linked to the public chain and has interworking data. One of the DApps is the identity token based on education degrees. The user mints identity tokens through degree verification+public chain Token, and the identity tokens are limited but have collectible value. Merchant-side payment, merchant-side points monetization, and decentralized no-border P2P, etc., can all be subchain applications based on the identity chain. Meanwhile, we introduce online fault tolerance mechanisms: whenever there occurs that a large amount of funds has been stolen, we can vote to temporarily freeze the account, and then report the case to legal institutions in order for them to arbitrate. And based on the actions taken by judicial departments, we can publicize the user’ s information, and deal with the natural person linked to the account.

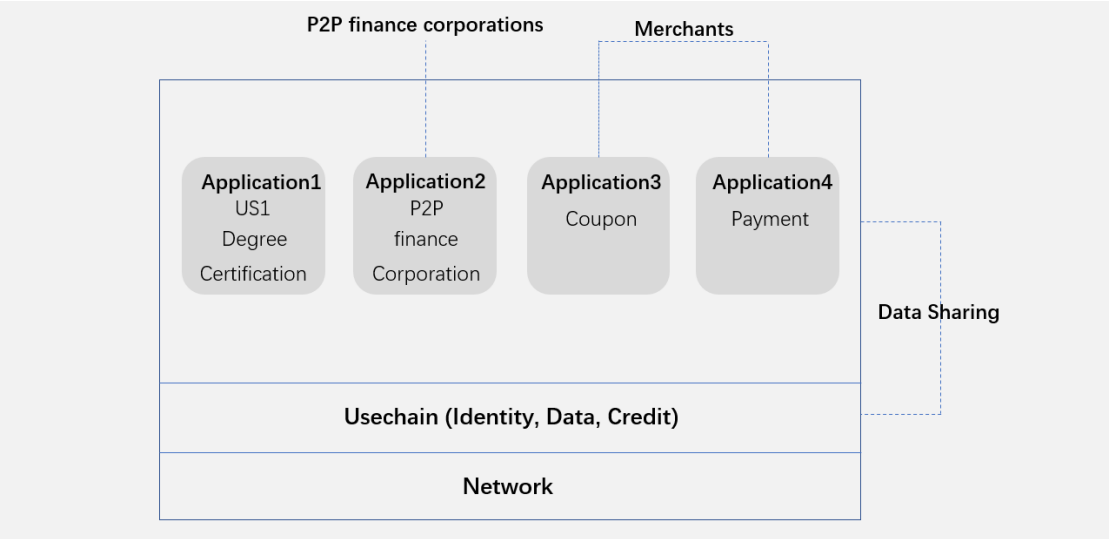


Fig 1-2 Identity Verified Application Development

The identity chain is a new-model public chain based on real users, can provide complete privacy protection and identity verification, provide specialized low-cost high-efficiency consensus algorithms based on identity, and expand the environment for smart contracts. On the basis of the identity chain, different identity-related applications can quickly

establish a decentralized, resource-sharing, and self-developing ecosystems.

2. Technological Framework

2.1 Ecological Architecture

In the ecological architecture of the whole identity chain, there’ re 2 major parts: the base level chain and the upper level ecological applications. Different from other blockchain applications which focus heavily on anonymity, the addresses on the chain will have a verified, one-to-one relationship with real people. And because of this real name account feature the identity chain will have a great impact on social media, personal credit, commercial promotions, and other various fields.

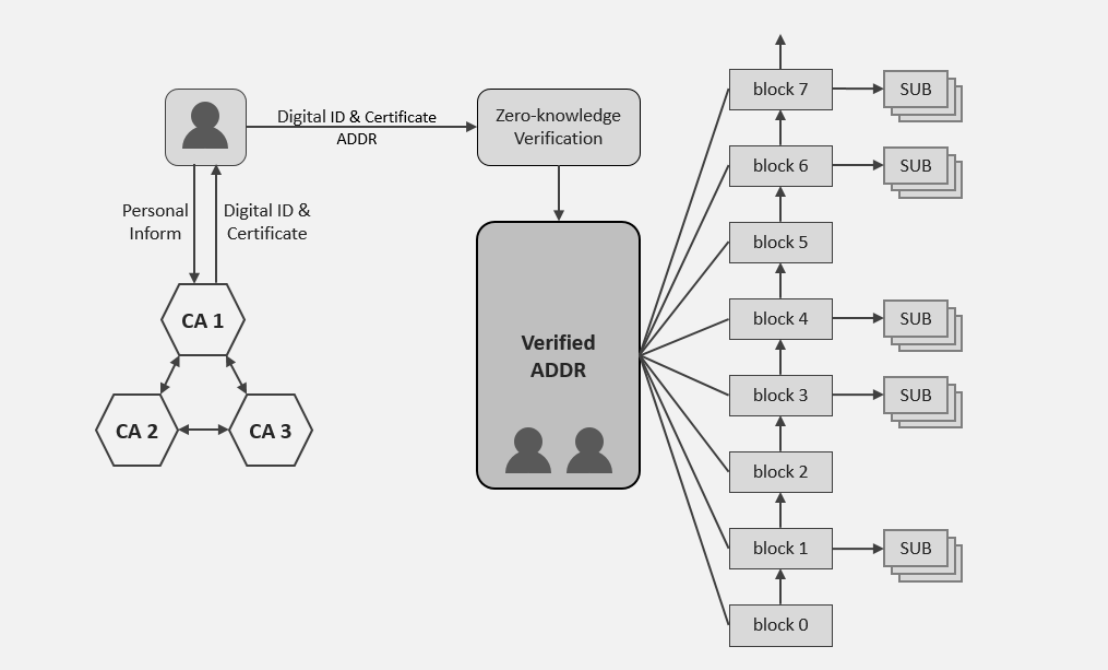


Fig 2-1 Identity Verification System Architecture

In designing the base chain, to deal with a series of problems that currently existing blockchain technologies have, such as the waste of resources in mining, the slow speed of transactions, the low capacity of

network transactions, the large quantity of synced data, etc., the identity chain will make these 4 major improvements based on the real identity specialties on the blockchain technology: Consensus Mechanism, Network Sharding, the Blockchain Virtual Machine, and Light Clients. First on the selection of consensus mechanisms, we make improvements on POW and POS, and propose RPOW (Randomized Proof of Work) and RPOS（Randomized Proof of Stake）, randomly producing the next block packager, which on top of avoiding computation power competition and ensuring decentralization, tremendously increases the block generation speed. Meanwhile, we use the hardware physics proof consensus algorithm to solve the problem of current consensus algorithms at its root.

In order to resolve network jams and support large scale transaction amounts, the Usechain chain-network system will introduce subchains to conduct different applications. The subchain is a blockchain generated from the main chain, on account of different needs of individual users, and can define its consensus method, block size, block time, and other functionality modules, according to specific application scenarios. In the meantime, based on the identity verification system on the Usechain main chain and the tokenization system of the USE, the applications on

the subchain can quickly be realized.

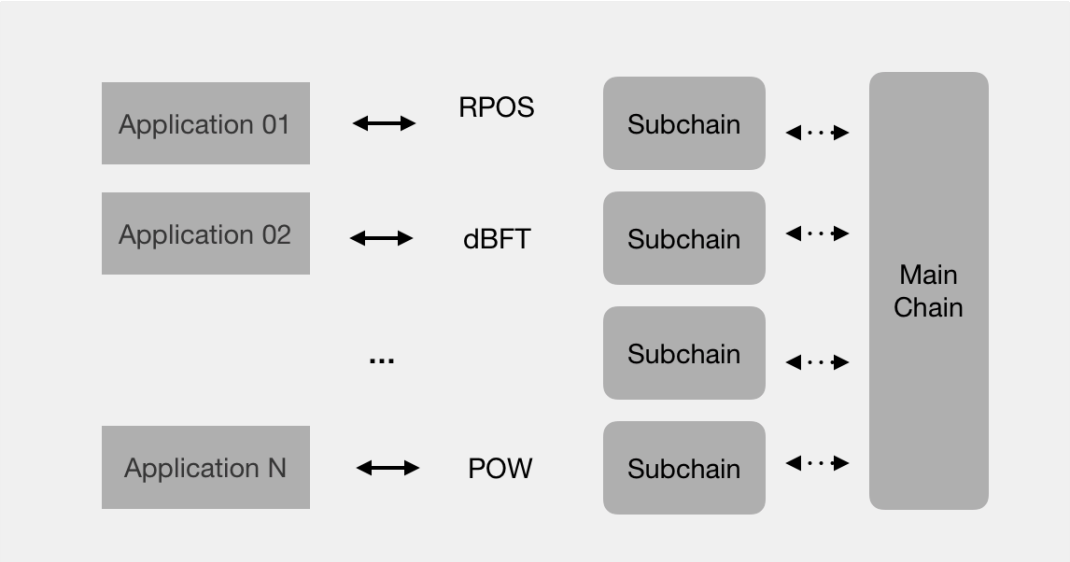


Figure 2-2 Subchain Applications

Based on the high credibility of the identity chain, we introduce the Sharding technology, conducting network sharding on the whole identity chain network according to user account addresses. Transactions on one address must be confirmed by all the nodes in the shard it belongs to and does not require the confirmation of all the nodes in the whole network, which on the basis of defending against double-spending attacks, increases transaction confirmation speed, and increase the capacity of network transactions.

On the basis of the new consensus algorithm and the sharding plan, we improve the blockchain virtual machine environment, dedicating ourselves to improving the computational performance of the virtual machine, lowering or avoiding calculation costs, optimizing the memory allocation module, adding more language support, detecting contract loopholes, supporting subchains and cross-chain activities, and external API interfaces.

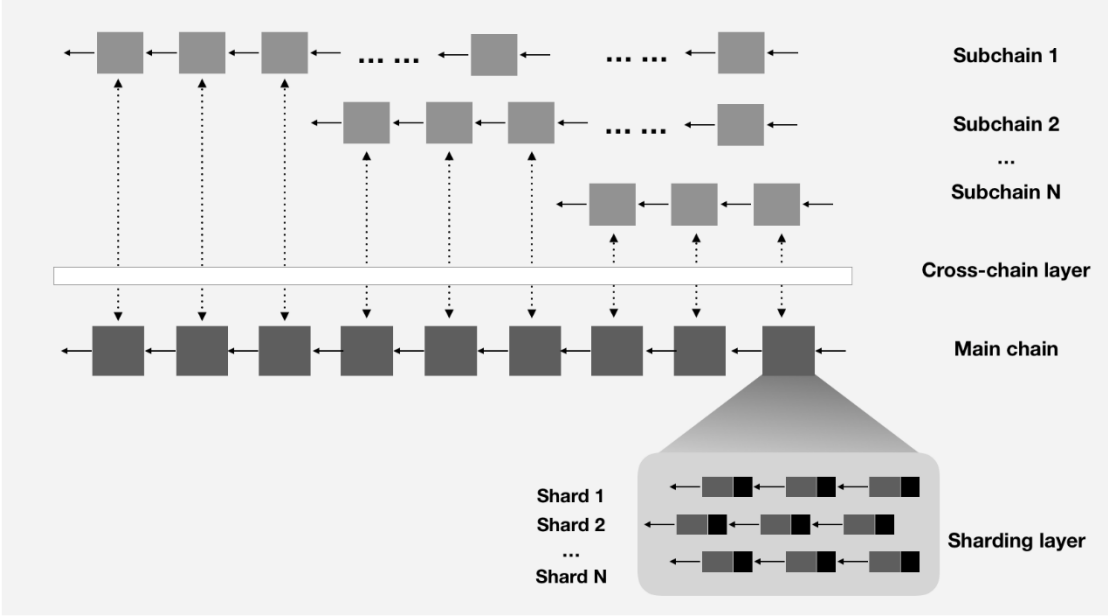


Figure 2-3 Subchain, Main Chain, and Sharding

In the meantime, in order to avoid the problem of excessively large synced data, we will focus on the Usechain light nodes, and on the theoretical basis of the Merkel proof, try to lower the data syncing amount that happens on light nodes, while ensuring the accuracy of verifying transactions and data. It is not necessary for every node in the identity chain network to be a full node, and those nodes that don’ t participate in consensus can be chosen as light nodes. Meanwhile, the design of the Usechain base level identity public chain is open, which provides a plethora of functionality modules and strong smart contract and virtual machines systems. Based on Usechain base level public chain, developers can quickly develop various sorts of DApps.

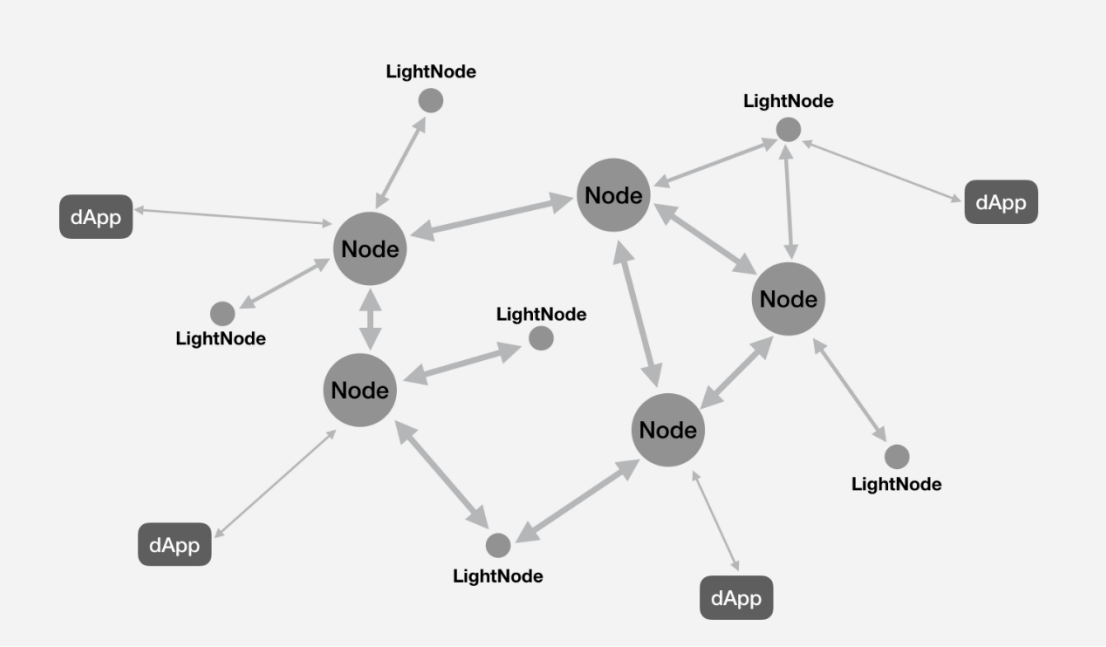


Figure 2-4 Usechain Network Nodes

The identity chain, through "decentralized application subchains" and "smart contracts", introduces the subchain, the public chain, and off-chain elements, forming blockchain smart contracts that are inline with real-world business logic. And support many industries and multiple channels. We provide mobile-end services through the technological structure, such as: mobile wallets, mobile DApps, mobile smart contract services. In the identity chain ecosystem, third-party developers (or third-party users) can quite easily establish their subchains and interwork with other subchains, providing the blockchain's mobile services, and pushing for the realization of the identity chain being applied in all sorts of difference scenarios together.

2.2 Software Stack

The design goal for the identity chain is to achieve high transaction capacity able to support the usage of tens of thousands of users simultaneously; to increase the block speed and the transaction confirmation efficiency; to optimize the virtual machines and simplify the

development and debugging process. We will achieve the construction of the entire identity chain, from the transactions, blocks, consensus value, light node verification, P2P network, and the blockchain virtual machine on the base level chain, to the upper level blockchain application.

On the design layers, we can divide them up into the network layer, the data layer, the consensus layer, and the smart contract layer. Every layer can serve certain applications, and satisfy the specific needs of different applications, providing guarantees for personal or business that they can quickly and securely realize all sorts of application scenarios and business models.

Network Layer: In order to strengthen network capacity and increase network transaction speed, we introduce the KaZaA P2P Protocol。To match the KaZaA Protocol, we also designed the INS (Identity Network Sharding)and the ITS(Identity Transaction Sharding). INS and ITS mainly includes: Generating Shards, Directory Service Committee, Resolving Conflicts, Transaction Assignment and Processing.

Data Layer: Upon the basis of the blocks' chain-style structure, determining whether a block is valid depends on whether the last block exists and is valid, whether the time stamp is valid, whether the block' s proof of work is valid, whether the internal transaction is valid, and other various aspects. The identity chain, on this basis, will ass valid address data in every block, using the Merkel Tree storage. All transactions require the validation of the addresses.

Traditional blockchain verification or transaction confirmation is through maintaining the blockchain wholesale data, but that means locally there needs to be an enormous amount of memory space to normally run the

blockchain nodes. The identity chain will introduce light nodes, optimize the data structural design using Merkel tree, thereby tremendously reducing the amount of data required for light nodes, and providing the possibility of mobile devices and Internet devices being worked into the blockchain.

Consensus Layer: On the design of consensus mechanisms, first we designed the brand new RPOW（Randomized Proof of Work） and RPOS（Randomized Proof of Stake）. RPOW is based on proof of work and the special user identity recognition mechanism only available in the identity chain, and has worked in a random allocation algorithm, giving every miner different mining difficulties, in order to adjust the network' s virtual computing power allocation. This algorithm, on the one hand, can tremendously increase the efficiency of traditional POW algorithms to reduce waste of resources, and on the other hand, can reduce the trend of centralization of computing power in the mining pool, in order to establish better network fairness and security. RPOS is mainly based on DPOS（Delegated Proof of Stake）, and optimizes the method of producing a committee every period, while relinquishing the sequential production of DPOS in terms of block producers, instead, designing a RAS (Random Appointment Strategy) to deal with the transition of production power within the committee.

In order to take the identity chain' s security and operating efficiency to the next level, we introduce the proof of hardware (POH) consensus algorithm. Using the security measures within the chips of computing hardware itself (CPU, etc.), we realize a consensus algorithm that is unmodifiable but credible, picking randomly among all nodes which one is the block producing one.

Incentive Layer: The identity chain' s public chain token, in design, is very closely related to personal identity, and as the identity chain technology ever advances, the applications that follow will ever be developed, and the value of the tokens will ever increase. For those mining nodes for the entire network nodes, there will be certain amounts of tokens as rewards for keeping the ledger. Compared to the Bitcoin periodic halving method and the Ethereum fixed incentives method, the identity chain will also take into consideration balancing the mining gains and network transaction fee level and design a unique incentives method. The identity chain also supports every node and every subchain to issue their own identity tokens, distributive trade between different tokens, realizing people' s own personal values.

Smart Contract Layer : By designing the dedicated Identity Virtual Machine (IVM) for identity chain using the design indexes such as calculating performance improvement, contract developing cost reduction, memory allocation model optimization, and development normalization etc. Meanwhile, there will be improvement on the implementation mechanism of smart contract, as wells as internal vulnerability detection tools to support enterprises using smart contracts.

Application Layer and online fault tolerance: The identity chain can support various applications based on identity and constructs the base for the future identity application industry. On this basis, the project team will develop a special application and online fault tolerance mechanisms. Whenever the identity chain has conflicts or bifurcations, it will automatically freeze the accounts and the block in question, while the

other nodes trade on. There will a special committee to arbitrate on the block in question and will sign to verify the decision.

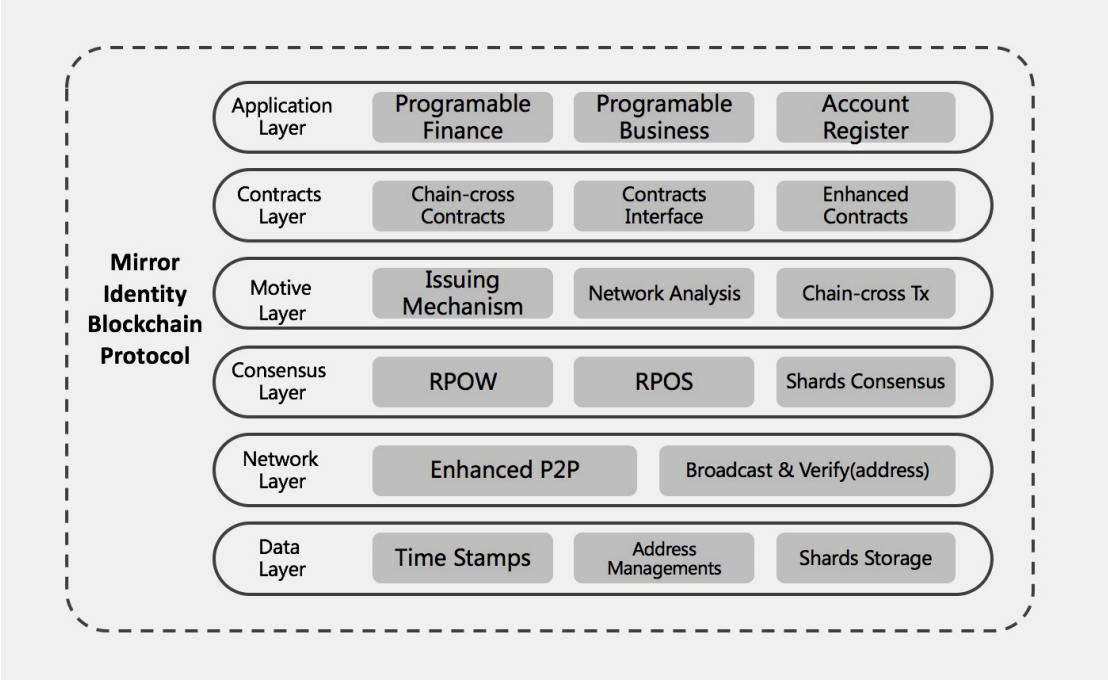


Figure 2-5 Mirror Identity Blockchain Protocol

3. Identity Verification

3.1 Main Address and Sub Address

The main address is the user’s main account, which is created when the user generates the wallet. Every user has only one main address that is linked one-to-one with identity. The main address uses the Elliptical Curve Secp256k1 to generate the public and private key, and then uses the public key to generate the address. Both main and sub addresses require verification to start transactions. There’s not limit on the number of sub addresses, which will be generated based on the user’s needs, using the user’s main account public key, and are kept separately from the main address. Therefore, the dependence relation between main addresses and sub addresses are not visible.

3.2 Sub Address Generation Algorithm

Sub addresses are generated according to the Elliptic-curve Diffie–Hellman (ECDH)[8, 17] algorithm. Suppose the pair of the user's public and private keys is (A, a), there will be a random Keypair(S, s) generated, using the committee's public key $(B_1, B_2, \ldots, B_n)$, to general its own pair of public and private keys:

Public key generation method:

$$A_1 = Hash([s]B_1 + A)G + Hash([s]B_2 + A)G + \cdots + Hash([s]B_n + A)G + S$$

Private key generation method:

$$a_1 = Hash([s]B_1 + A) + Hash([s]B_2 + A) + \cdots + Hash([s]B_n + A) + s$$

The sub account is then: (A_1, S), and the corresponding private key will be stored into the Keystore。

3.3 Identity Verification Process

In the upper level applications, first we need the real name verification of the identity chain address. The identity chain will achieve the verification and construction of every real entity's main address through multiple publicly trusted third-party agencies. The process is as follows:

1) The user applies for the CA certificate to the third-party agencies, and the third party verifies and returns to the user the certificate. This process will only occur once.

2) Then, the user can prove to the blockchain that they have the verification using zero-knowledge proof (but there's no need to publicize any information about the CA). The address generated by the user will be recorded by the blockchain smart contract and becomes the user's only main address. There is no identity information on the main account, so the identity information stays confidential.

3) The user uses their public and private keys and a randomly selected public key on the blockchain to apply to the committee for secondary

address authentication through the ring signature strategy.

4）When the user wants to send a transaction, the validity of the transaction must be verified. Usechain verifies the address through the CA contract, determines whether the transaction is legal, and then executes the transaction if it is legal.

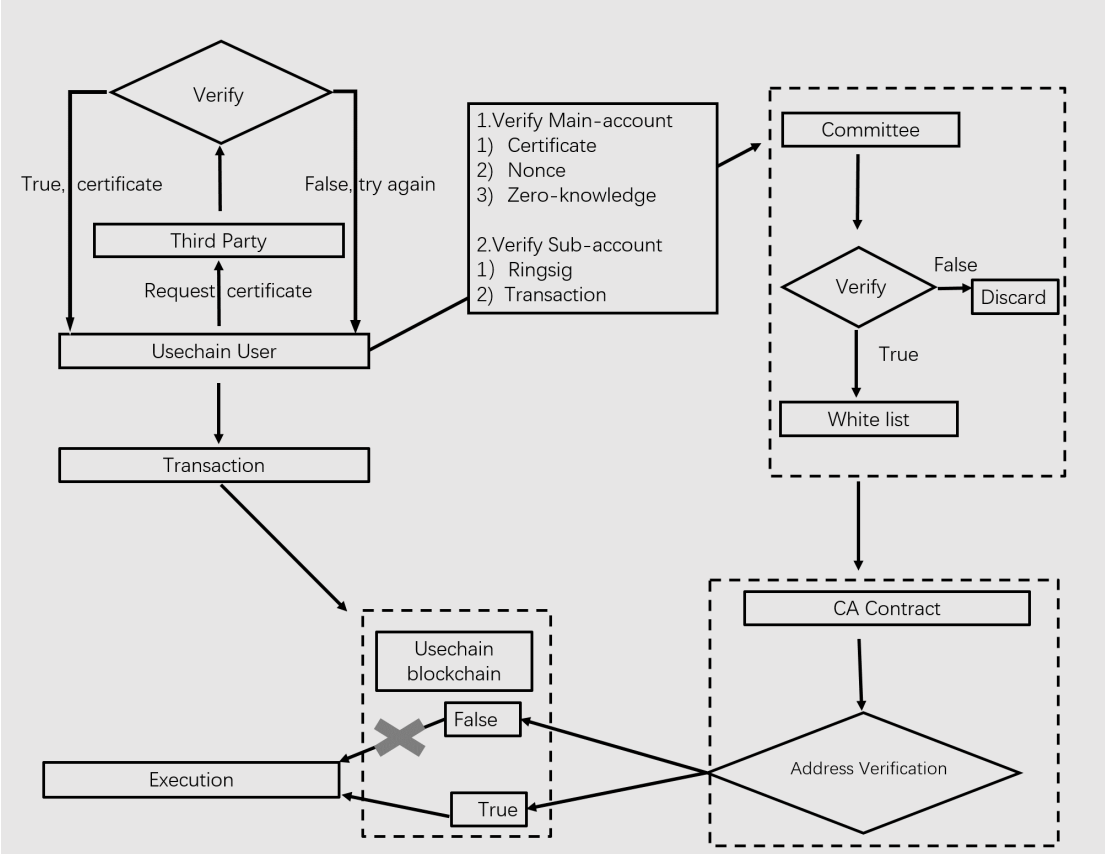


Figure 3-1 Identity Chain Verification Process and Transaction

5）In necessary cases such as if the account is suspected of illegal activity, the committee scans the network for account relevance. The committee's public and private key pairs are respectively(B_1, b_1)，(B_2, b_2)，…，(B_n, b_n)，so according to the elliptic curve Diffie-Hellman key exchange algorithm:

$$\begin{cases} B = [b]G; S = [s]G \\ [s]B = [s][b]G = [b]S \end{cases}$$

The committee calculates:

$$A_{11} = Hash([b_1]S + A)G$$
$$A_{12} = Hash([b_2]S + A)G$$
$$\cdots\cdots$$
$$A_{1n} = Hash([b_n]S + A)G$$
$$A'_1 = A_{11} + A_{12} + \cdots + A_{1n} + S$$

When $A_1 = A'_1$, you can find the account relevance.

4. Consensus Algorithm

In a distributed network, ensuring consistency (all nodes reach consensus on the same proposal or data) is the central and most important issue, and the consensus algorithm is the algorithm used to do this. Since there are many nodes in a distributed network, communication delays in the network are inevitable, and the nodes may suffer from complications such as downtime, fault, and inefficiency. For blockchains, it is necessary to consider these factors and defend against an attack from a certain number of malicious nodes, while also achieving maximum decentralization. Therefore, the consensus algorithm is the most critical part of the blockchain system and worthy of continued research and optimization. At the same time, it is imperative to realize that it is impossible to find a perfect consensus algorithm that can not only achieve a very high transaction load, but also ensure the rapid confirmation of transactions and maintain decentralization. Usechain's entire network does not rely on a single main chain to carry all the

applications. Through network sharding, side chains, and other technologies, the main chain and sub chains are divided to handle different application areas. In designing subchains, the most suitable consensus algorithm will be selected according to the requirements of specific application scenarios.

4.1 RPOW

4.1.1 RPOW

The Bitcoin network uses a proof-of-work mechanism[10]. Through the computational competition mechanism, nodes compete to solve the computation problem first, and the fastest node performs the accounting work. So far, the POW algorithm is the only consensus algorithm that has been tested by a large number of users for a long time.

Satoshi Nakamoto designed POW with the intention of allowing every Bitcoin node to participate in the decision-making mechanism of the entire system. However, with the progression from GPU mining, to FPGA, and then ASIC mining, the pool of centralized computing now completely deviates from the ideals of democracy and decentralization. Many miners completely do not understand the Bitcoin ecosystem but control the direction of its development. In addition, computational competition in

the POW system consumes a lot of electricity without producing any social by products, which is a waste.

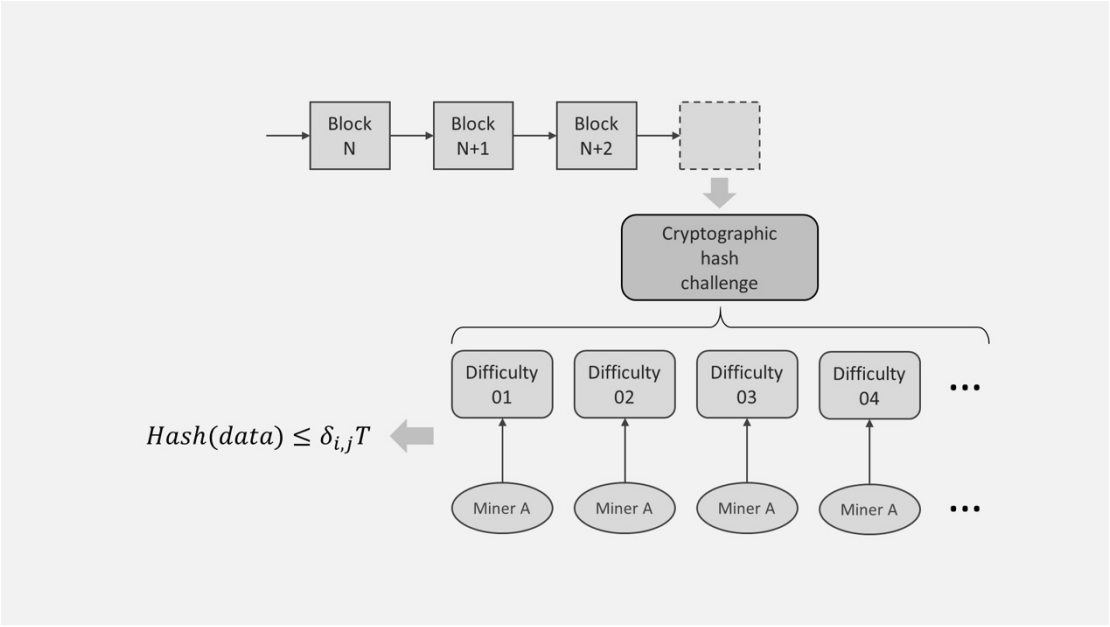


Figure 4-1 RPOW Consensus Algorithm

In order to fix the problems emerging with POW while maintaining its excellent characteristics, this project proposes the RPOW algorithm. In the RPOW algorithm, each miner i gets a random difficulty reduction factor $0 < \delta_{i,j} < 1$, when packing the jth block, so each miner needs to calculate a hash value that satisfies.

$$Hash(data) \leq \delta_{i,j}T$$

where T is the base difficulty value of the whole chain. This way, the computation difficulty varies for each node in the network, and each block will change randomly. In order to achieve $\delta_{i,j}$ randomness, identifiability, and unmodifiability, $\delta_{i,j}$ can be calculated based on each mining master address, current blockchain data, and pre-stored information. The specific miner mining process is as follows:

1) Miner A synchronizes the full amount of the ledger data and imports the

mining account information. The mining account address is T;

2) After receiving the latest block, check the validity of the block;

3) Package the transaction data in the transaction buffer pool and check the transaction validity of the packaged transaction at the same time;

4) A calculates its current mining difficulty. The basic difficulty will be adjusted based on the time of the last block’ s output and the number of blocks to be produced. At the same time, the miner will perform data signature on the block data and mining address of the previous block to obtain the signature string. S;

5) Calculate the matching degree of the signature S and the address T, and calculate the difficulty adjustment according to the matching degree. A small number of nodes may have a much reduced computation difficulty, increasing the probability of producing the block;

6) Adjust the block data (nonce, timestamp, transaction, etc.) and perform hash operation until the block meets the difficulty requirements;

7) Broadcast a new block. Other nodes receive the block and verify the validity of the block including the difficulty.

The RPOW algorithm can reduce the concentration of computing power in the mining pool, reduce the waste of power resources, and at the same time increase the block production speed of the entire system by more than 100 times.

4.1.2 Hardware-based RPOW

This project will use a software-based RPOW algorithm as a transitional consensus algorithm. The identity chain will eventually implement a hardware-based RPOW consensus algorithm that uses dedicated hardware devices to generate consensus mechanisms and identity authentication mechanisms. This will ensure the security and efficiency of the blockchain network without requiring any energy consumption at all.

4.2 RPOS

4.2.1 DPOS Mechanism Description

DPOS (Delegated Proof of Stake) is similar to the People's Congress system. In DPOS, everyone who owns the token can vote for block producers, and the 21 with the most votes are elected as witnesses (the number of EOS block producers is currently 21, while Bitshares has 101, but in theory a single digit number of delegates would work as well). The weight of a node’ s vote is positively correlated with the number of tokens they own. The elected witnesses produce blocks during the period of their service. The mining order is negotiated by the witnesses (usually a cycle). Each period, elections for witnesses are conducted again, and nodes can vote out malicious or incompetent block producers. DPOS gives each stakeholder the right to vote and reduces the cost of running the network.

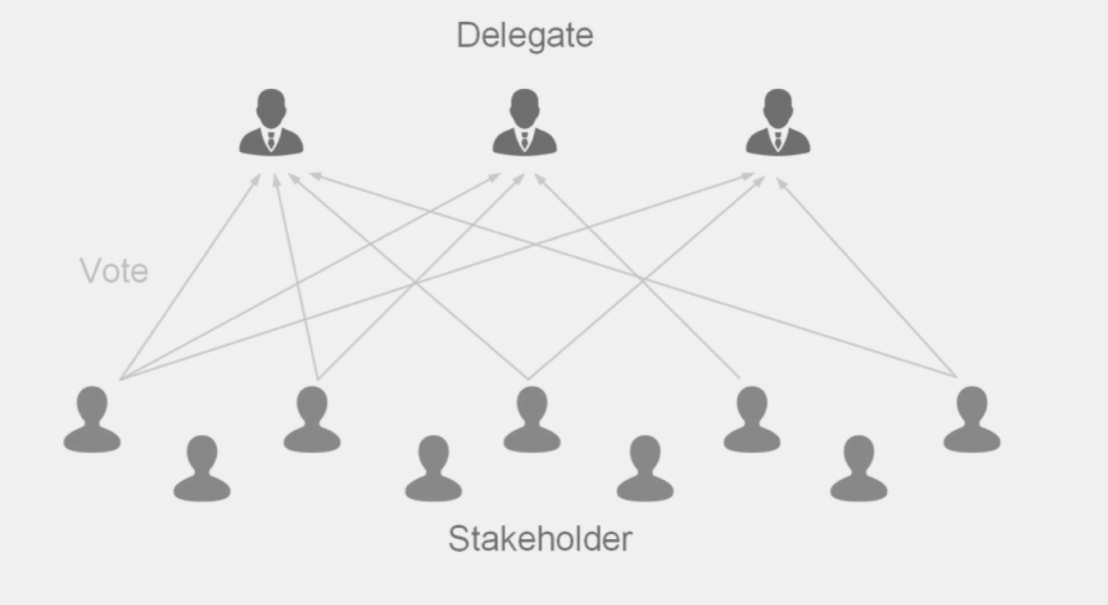


Figure 4-2 Witness Vote

4.2.2 DPOS Design

DPOS is used on the subchain of Usechain as an important member of its

blockchain network. In DPOS, the stakeholders and witnesses together maintain the operations of the blockchain.

Any node holding USE has voting rights and can only vote once per round of voting. Voting is achieved through sending transactions. The vote weight is positively correlated with the USE held by the node. There are two types of votes in the Usechain system (support votes and no votes). Usechain maintains a data structure to store statistics on all the voting information on the chain, and provides a query interface. Usechain will periodically clear votes from previous rounds to save space.

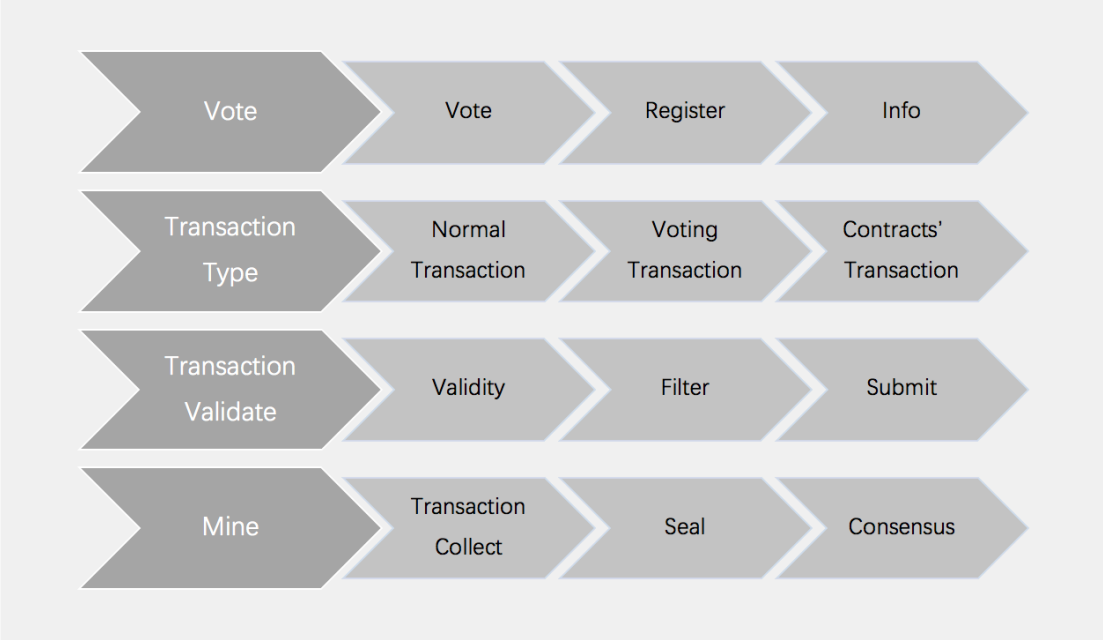


Figure 4-3 DPOS Transaction Process

Compared to POW, the DPOS consensus mechanism favors a cooperative relationship between block producers rather than a competitive relationship. Hence, blockchains using DPOS do not really need to take extra measures against forks. If a small number of nodes malfunction or orchestrate an attack, the other nodes will continue to function using the

longest chain. For witness nodes that miss too many blocks, issue invalid blocks, or are clearly attempting to exploit the system, it is highly likely that in the next epoch they will be voted out, and the normal nodes will select a new delegate through voting.

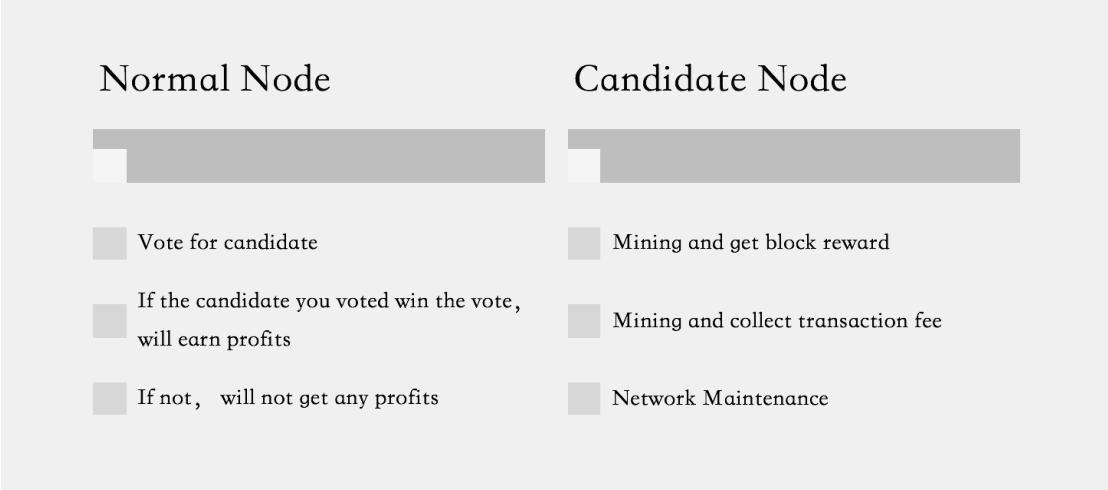


Figure 4-4 Normal Node & Candidate Node

DPOS greatly improves transaction processing speed, and because in this consensus mechanism only the witness nodes have the right to produce blocks, it does not generate the energy waste of brute computational force competitions like POW. But DPOS still has much room for improvement. In the DPOS mechanism, after a witness node packages a block, it must be verified by over 2/3 of the other witnesses in order to confirm that the block has a high probability of irreversibility. When the block is confirmed, we can use a BFT (Byzantine fault tolerance) algorithm to increase confirmation speed, as the BFT-DPOS(the DPOS based on Byzantine fault tolerance). BFT-DPOS consensus algorithm follows the following thought process: If there are 51 witness nodes, and each

witness node produces 5 consecutive blocks with a block interval of 1 second (the frequency of blocks is increased, but the rotation period of the delegate does not change, as in the block producer for each interval stays the same but they produce more blocks in that time. After a witness node produces blocks, it immediately sends the blocks to the other 50 witness nodes, and after the block receives more than 35 witness node’ s (more than 2/3) confirmation, the block is irreversible. This further increases the speed of the transaction, and at the same time solve the impact of network delay caused by producing the block. Before producing the blocks, the 51 clients will calculate the optimal block production sequence based on their mutual network delays. [15]

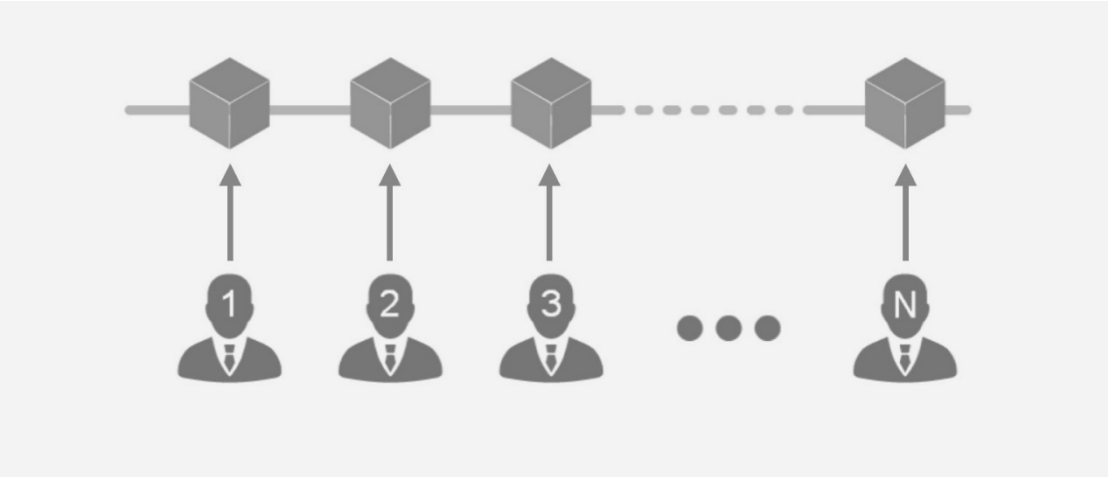


Figure 4-5 witness generate block

4.2.3 RPOS

The small number of witness nodes in DPOS could make the blockchain vulnerable to attack, as it is possible for an attacker to conduct a denial of service attack on each delegate node when it is their turn to produce.

Since the identity of each representative is in fact its public key, not the IP address, the threat of this particular attack can be mitigated but cannot be avoided. Production blocks represent a predictable sequence, and attackers can still attack by analyzing the producer's IP address.

In order to avoid this problem, the RPOS algorithm is proposed based on the DPOS algorithm. In RPOS, the random selection principle is used instead of the ordering principle. The producer B signs the previous block with their private key, attaches the signature to the current block, and the system calculates the next producer's ID from the signature value. With such a scheme, when A produces a block, the next producer B can calculate the ID of the next producer. Because block01 is signed, producer B cannot control the next producer ID. It is determined by the hash value of the previous block. The main flow of the RPOS algorithm is as follows:

1) In each voting period, main accounts that have passed authentication can vote to select witnesses. Each person can only cast one vote, and subaddresses cannot vote;

2) After the voting period, the system automatically counts the votes. The first N candidates with the most votes are chosen as witnesses and are responsible for producing blocks in the next epoch;

3) During the witnesses' block production period, when a witness packages a block, they sign the previous block and the witness' s main account address with the private key. Assuming the public and private key pair of the witness is (sk, pk),

select the random number $k \in [1, l-1]$, $Q = [k]G$, and calculate the equation:

$$r = H(A||pk||m)\ mod\ n$$

$$s = k - r \bullet sk\ mod\ n$$

Here, H is the hash function, m is the signature information, that is the hash value of the previous block plus the address information of the current witness, (r,s) is the signature value;

4) Write the obtained signature value (r,s) to the current block;

5) According to the signature value (r,s), calculate the producer of the next block, the calculation equation is:

$$ID = H(r, s)\ mod\ n$$

H is the hash function, mod is the remainder operation, n is the number of witnesses;

6) The producer of the next block has a qualified ID.

7) A witness that has not obtained the right to produce the next block cannot successfully produce the next block. Even if they generate the next block, it will not be verified by other nodes.

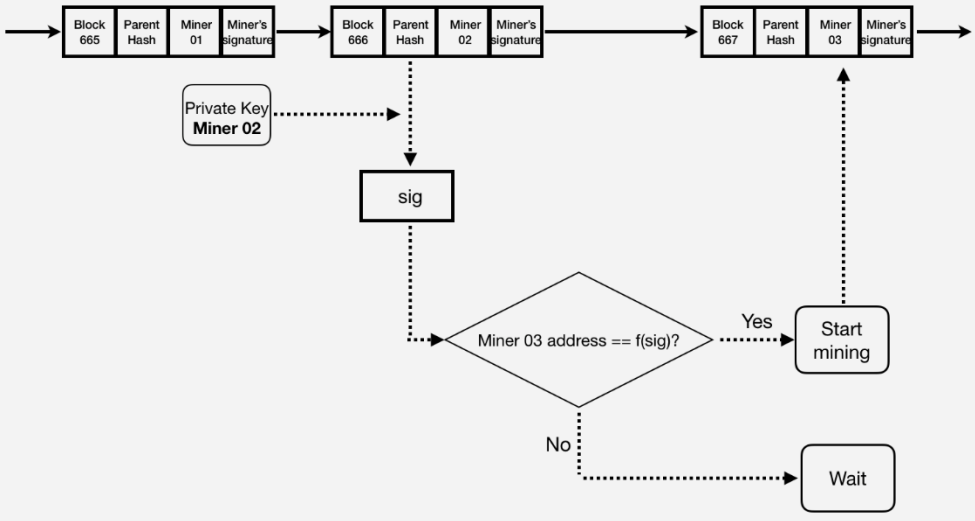


Figure 4-6 Basic RPOS workflow

If the producers A, B, and C are all attackers and collaborate with each other, A can constantly adjust the contents of the blocks while packing the blocks, so that the B-signed data can specify a specific producer, and even the attacker C can be specified. The power of production blocks only flows between attackers A, B, and C, and other producers cannot obtain production power. In order to solve such problems, if the average number of producers who obtain production power greatly deviates from the theoretical value, it will be resolved by reducing the probabilities and rewards of the representatives who have obtained the production rights many times, and even giving a certain punishment mechanism or kicking off the representative committee. In the identity chain, since each transaction and each miner has an identity, the cost of doing evil at each address is greatly increased, and it is easier for the network to track and identify malicious nodes. By using this feature of the identity chain.

In the normal network environment, the representatives selected by RPOS are all 100% online. Any information can be received by all representatives within 2s. This means that after a trade of 1.5 seconds on average, it will be written into the blockchain and the transaction will be known to all outgoing nodes.

Of course, if the entire network is congested or some nodes are down, the entire blockchain may be bifurcated. In order to ensure the validity of the transaction, it is necessary to wait for a certain number of block confirmations. If it is a POW algorithm, it completely depends on the barriers of computational power brought by the block length, and the irreversible waiting period for the transaction will be relatively long.

In RPOS+TaPos system, each transaction will include the hash value of the nearest block (backward within 5 blocks). This prevents a large number of transaction records on the chain of forked blocks and allows the system to sense whether the user is In the forked blockchain. After the fork is generated, the representative may find the generation of the fork and remind each user in a few seconds. If a node finds two consecutive lost packets, this node is 95% likely to be in a fork; if it finds that there are three consecutive lost packets, then 99% of the nodes are in the fork.

4.3 Tokenizing High-speed Mining Licenses

At present, mining has become a profitable industry. In 2017, Bitland’ s profit was US$2.5 billion. But the profits of the miners are detrimental to the entire ecosystem. On the one hand, a small number of high-speed miners have a large amount of Bitcoin, making Bitcoin's distribution very concentrated. On the other hand, the cost of mining causes resources to

be wasted. At the same time, the transaction costs make it difficult to use the actual application. We introduced the USMK, a high-speed mining permit license. The cost of USMK is 2 Kth power (USE+USK). Having a real-name account of USMK can reduce the difficulty of mining by K. 0. At any time, everyone can buy USM1, but with USMK, if K is greater than 1, USM (K-1) already has 20 pieces. Users who also have low-level USMK1 can also upgrade to USMK2. K2 is greater than K1. The premise is that there are already 20 USM (K2-1), and make up the difference. USMK can also be traded in the chain. However, accounts using USMK must have USMK for 24 hours before use.

A random miner can obtain the upper part of the block by dividing the hash value of the previous block by the miner of the NUSMK+1 to obtain a K number of zeros less than the normal difficulty. If the remainder is zero, then the company will replace the public account for mining. The rewards are returned to the public account. NUSMK corresponds to the number of USMKs. In addition, any account cannot dug two blocks in a row.

5. Sharding and Subchains

The current blockchain feature is that all nodes propagate transactions at the same time, and verify the execution results of smart contracts, ensuring that each node's stateDB is consistent. With the increasing number of calls for smart contracts and the increase in the number of transactions, the time for transaction confirmation also increases. With the ever-increasing volume of transactions, there has been significant network congestion. If the entire network processing transaction speed depends on the speed of some super nodes, this will make the network computing power controlled by a few super nodes. In this case, Usechain process transactions in different networks will greatly increase the speed of transaction processing and confirmation.

5.1 Sharding

Based on the identity authentication feature of each node in the identity chain, Usechain designed a fragmentation mechanism to expand the speed and scalability of transaction processing. Each shard has a corresponding node to process the transaction, and the slicing can process transactions and perform validation of the smart contract in parallel. Each shard has a different slicing ID, has a fixed selection of nodes to collect transactions, and packages the transactions in a period of time into a block, and at the same time makes a signature, and then collectively submits them to the upper layer for processing. The

consensus node packs different shard blocks into the blockchain. Different from other current public chains, the identity chain can use the user's authentication mechanism and fault tolerance mechanism to improve the security of the network. Considering the transaction processing speed of the fragments, the nodes in each shard cannot be too many. Therefore, the identity mirroring chain can divide the network into more fragments, which greatly improves the overall transaction processing performance of the network.

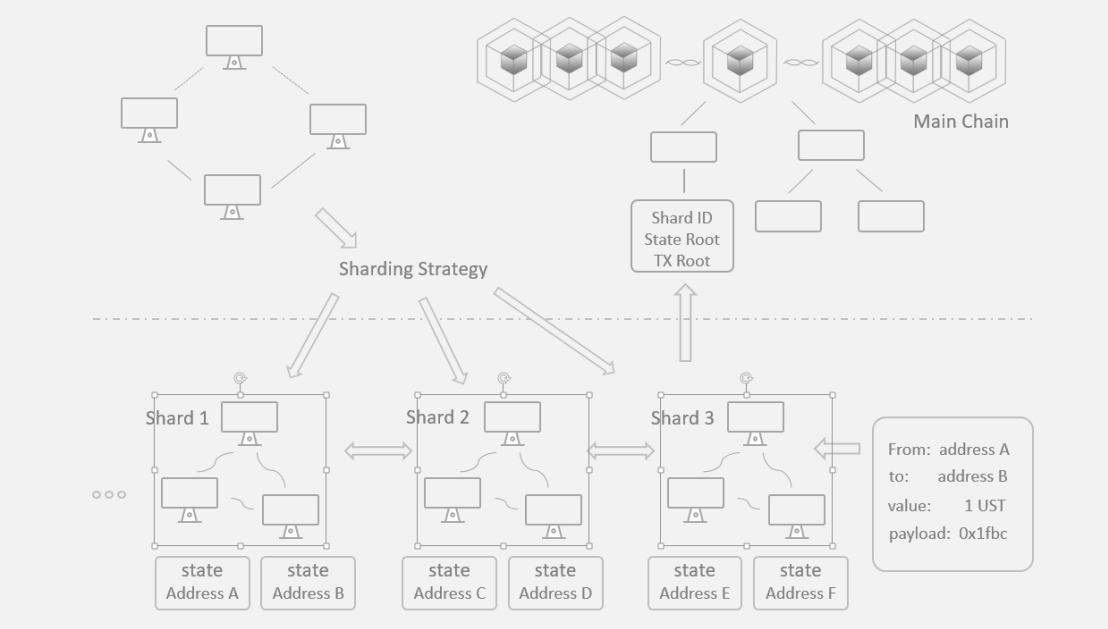


Figure 5-1 Sharding Model

The sharding is two layers. The upper consensus node runs a main chain. The main consensus node receives the sharding transactions collected from the sharding consensus nodes. The main chain also processes the transactions of the consensus nodes. The sharding consensus node

handles the transactions within the shard independently, and the number of shards increases while more transactions can be processed.

5.2.1 Main Chain Consensus Mechanism

The main chain uses the consensus algorithm of RPOW to randomly generate node packing blocks to avoid the mine pool control network. The upper-layer network nodes are elected at regular intervals. The upper-layer network nodes are selected by the RPOW consensus mechanism. The upper-level consensus nodes verify the transactions collected by the sharding nodes and join the blockchain after signing.

The characteristics of the upper network are:

1) In order to avoid super nodes or mine pools, the upper network nodes are randomly selected by the RPOW consensus;
2) The nodes of the upper network packet block are randomly generated each time ;
3) The RPOW itself is random. In order to reduce the probability of joint evils between nodes, the fragmentation nodes are updated regularly.

5.2.2 Sharding Consensus Mechanism

The sharding node has its own consensus mechanism. The intra-slice consensus node is elected based on the RPOS voting. The node needs to register in the contract of the main chain and pay a certain margin at the same time before voting. After the sharding node is selected, the

transaction within the shard is identified by the consensus node, packaged into a block and signed by the currently packaged node, and passed to the upper layer network node. The upper layer network node updates the blockchain after confirming that the block is legal.

Sharding features:

1) When the upper-level network node agrees with $T_{checkpoint}$ for a certain time, it will select a new network segment;

2) The nodes in each shard only deal with the transactions in their own shard, and the shards can communicate;

3) The addition of fragmentation can speed up the process of consensus and achieve consensus on transactions more quickly.

5.2.3 Sharding Transaction Processing

The main chain can support the transaction of the user's main address and subaddress, and the RPOW consensus package transaction. The intra-shard transactions are collected by separate authorized nodes and sent to the upper main chain for packaging, which can increase the number of shards to expand transaction processing speed. The user's address within each slice is generated by the user's home address public key, the committee's public key, the slice ID, etc., and can uniquely determine the user's address in each slice.

When trading between different shards, users do not need to manage their own address within the shard. Only the main address can participate

in cross-shard transactions and execute smart contracts. Trans-segment transaction processing means that the transaction is first processed within a segment, and after the transaction is confirmed on the main chain, the transaction addresses of other segments have been confirmed by the main chain before the transfer operation can be performed. Expanding the number of shards can increase the transaction throughput of Usechain, and nodes are connected to form shards through a sharding mechanism. The bottom layer of the slice is connected through the P2P network, and the transaction is spread in the same slice through certain rules. This avoids frequent acknowledgment across shards. The initiator of a transaction is allocated to different shards after a certain period of time to avoid malicious nodes always affecting a shard.

5.3 Subchain and Cross-chain Transactions

Cross-chain technology can be understood as a bridge connecting the blockchains. Its main application is to realize atomic transactions among blockchains, asset conversions, blockchain internal information exchange, or solving Oracle problems. Cross-chaining is a complex process. It requires not only a separate verification capability for the nodes in the chain, but also a decentralized input. It also requires the acquisition and verification of information outside the chain. Each chain in the blockchain world is an independent ledger and there is no correlation between them. In essence there is no way to transfer between

books, but the value of a particular user's storage in a blockchain can become value in another chain, which is a cross-chain transaction. Cross-chain is a conversion between different holders.

We also considered cross-chain transactions when we designed the main chain and shards. The sub-chain uses consensus algorithms such as RPOS, RPOW, etc. The sub-chain is run by our partners, and the sub-chain can run independently. The current feasible method of cross-chain trading is to use smart contracts to achieve cross-chain transfers by anchoring coins in the main chain and sub-chains. At the same time, the addition of splitting can accelerate the speed of cross-link transfer confirmation.

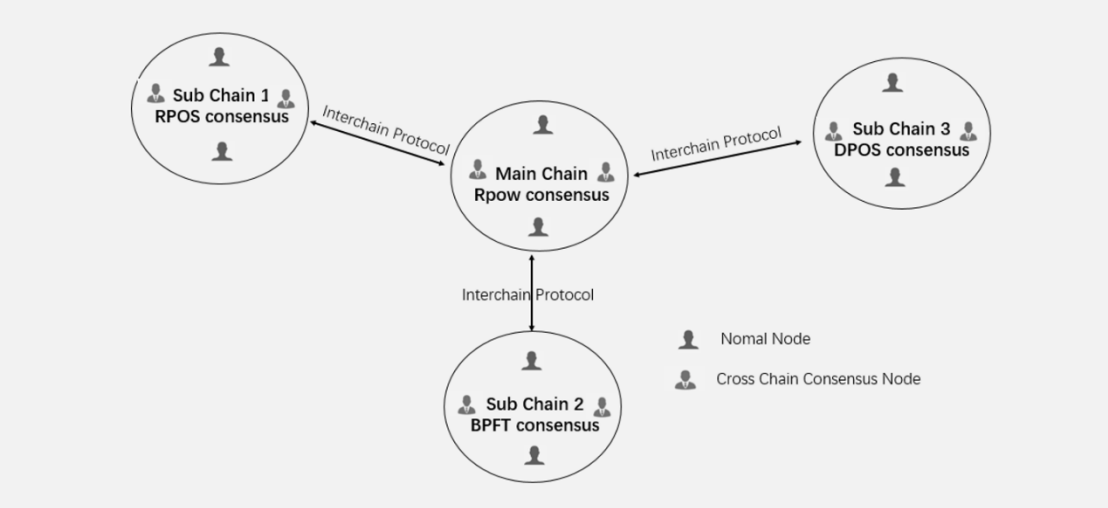


Figure 5-2 Cross-chain protocol

6. P2P Network

At present, the main problem facing the public chain is scalability. With the increase in the number of users, the transaction volume is getting larger and larger, but the total transaction speed of the network is fixed.

Therefore, the transaction and smart contract storage and call operations in the current blockchain network are more and more frequent, and the transaction is confirmed. There was no significant increase in speed. At present, the transaction confirmation rate of Bitcoin is 7 Tx/s, and the transaction confirmation rate of Ethereum is around 25 Tx/s. The unified consensus mechanism of the entire network limits the overall transaction speed, resulting in the slow rate of single transaction confirmation. The current solution is Bitcoin's expansion of the block size to include more transaction and sub-chain technology applications, but these methods do not fundamentally solve the problem of low performance of the main chain transaction.

6.1 Layered P2P Network

Based on the KaZaA protocol and the last network fragmentation requirement, a corresponding underlying P2P network structure is established, as shown in Figure 6-1. The network node is divided into a super node SN and an ordinary node ON. The SN usually has high bandwidth, high processing capacity, large storage capacity, no NAT restriction, and long-term online service capability. Each ON accesses the network and establishes and maintains a semi-permanent TCP connection with a parent SN; ordinary nodes hash the data on their shared links. Distributed P2P protocols are used between SNs to maintain

long-term TCP connectivity, robustness, and fast data distribution capabilities, forming a super-node overlay network.

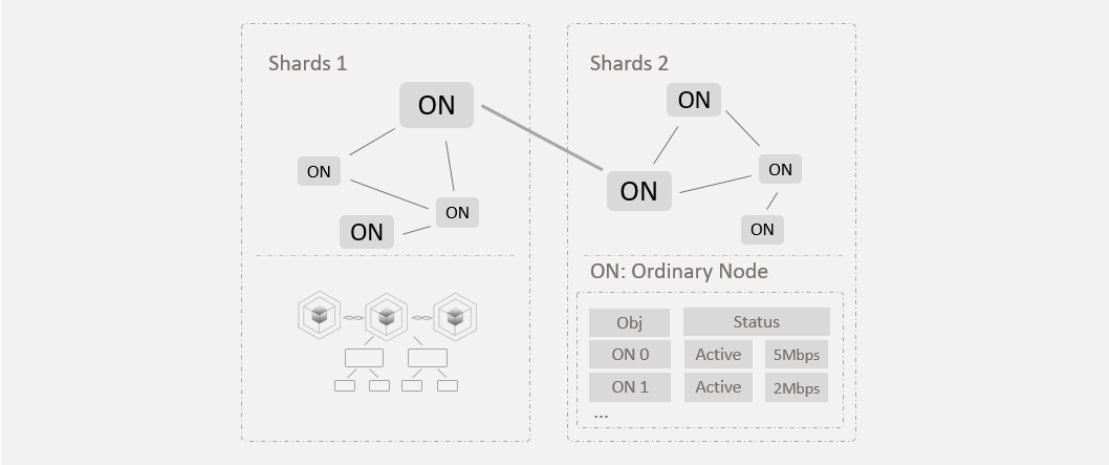


Figure 6-1 P2P Network Node

6.2 Network Sharding

Network sharding is a mechanism for managing nodes. According to certain sharding rules, nodes are automatically divided into different network shards. The nodes in each shard network are fixed, and all network shards can be parallelized. Dealing with transactions, the total network throughput will increase linearly.

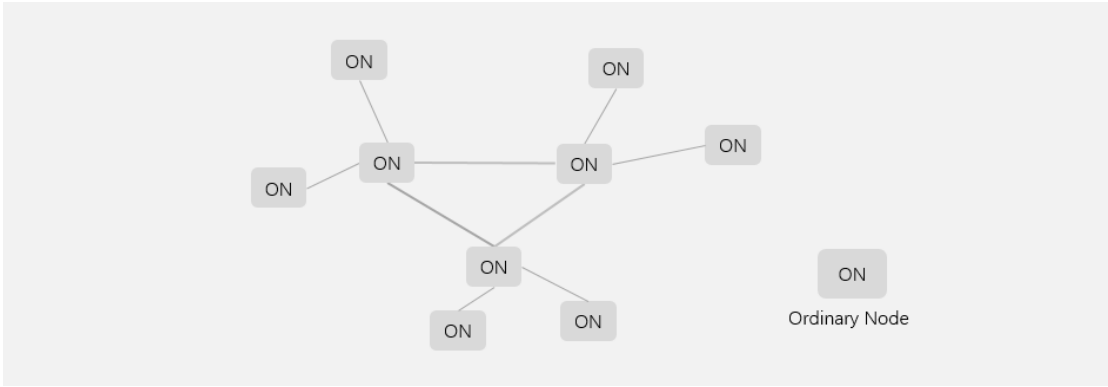


图 6-2 Network Sharding

The nodes in the network shard dynamically increase or decrease according to different bandwidths. Nodes in a certain network state can spontaneously form network shards. When a node in a network shard reaches a certain number, other nodes can spontaneously transfer to other network shards.

7. Smart Contracts and Virtual Machines IVM

Usechain is a decentralized application platform that supports multiple different types of identity blockchain applications in addition to Usechain, and allows users and companies to create their own application systems. Financial transactions, credit certificates, or other more complex applications can be implemented and run on Usechain automatically and reliably. This will bring innovations in areas such as degree certification systems, shared insurance, microfinance, and shared investment. change.

Taking into account the use of the public chain, the project will design an enhanced ERC20 interface that will contain the relevant calls for identity information, thus enabling the design and development of a standard identity token.

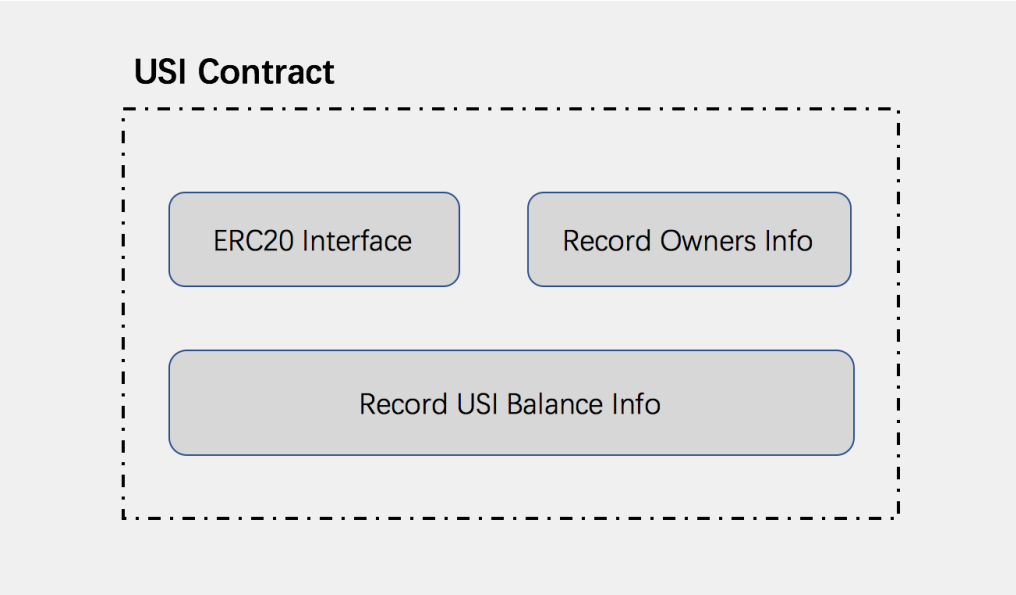


Figure 7-1 ERC20 interface improvements

7.1 Contract

The contract is deployed and implemented by all nodes in the decentralized Ethereum network to ensure that the contract is correct, secure, and non-repudiable. The content of the contract is specified by the code of the contract and is executed by the EVM. The person who proposes the execution of the contract needs to pay a small amount of gas to stimulate the Ethereum network to work properly.

7.1.1 Application Layer Development

Usechain provides a rich set of application layer protocols and components that support a variety of different requirements for identity blockchain applications. Application developers can quickly develop a decentralized application without paying attention to the underlying data interactions.

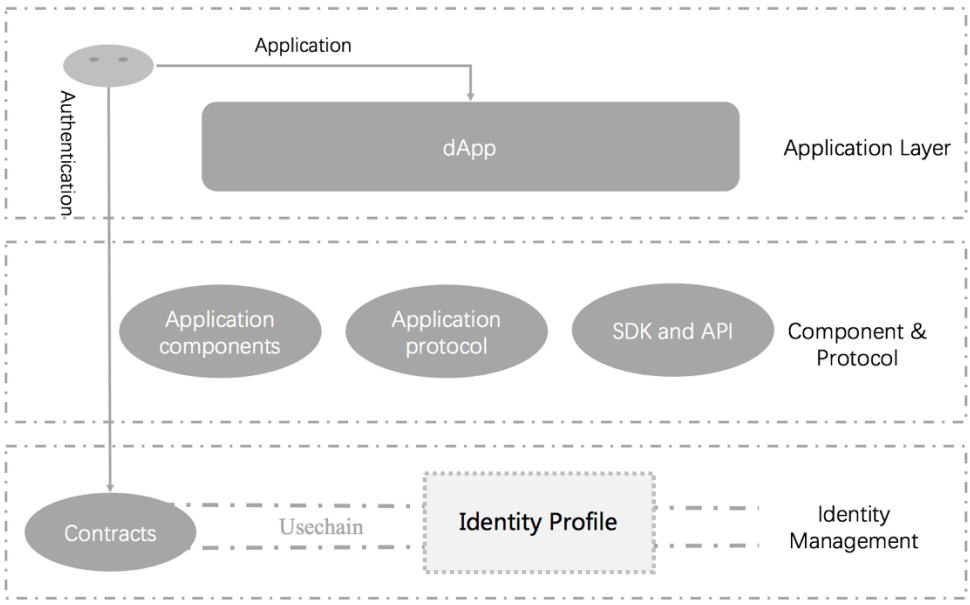


Figure 7-2 Usechain Application Framework

1) Usechain builds distributed trust on the basis of decentralized mirror identity system and smart contracts;

2) Provides interface to the upper layer of application to use the functions of Usechain through protocols, components, and API;

3) Usechain focus on DApp in different scenarios, provides authentication services, and solve trust issues from the bottom layer.

7.1.2 Contracts Speed-up (Code Layer)

Smart contract is the most common and powerful feature of Ethereum. In addition to basic peer-to-peer transfer for external accounts, any other transfer or logic-involved transaction (such as multisig, withdrawal delay, agency hosting, etc.) can use smart contract. Therefore, smart contract is considered as a high frequency function of Ethereum, and the codes within smart contract are in direct relation to the execution efficiency of the contract, thus affect the actual processing capacity of Ethereum.

7.1.3 Solidity: Defects & Improvements

Smart contract in Ethereum consists of Bytecode, and Bytecode, which operates the bottom layer based on Stack, is not applicable for writing complex contracts. Thus, Solidity, an object-oriented high-level language similar to JS, which is easy to learn and program, was introduced by Ethereum. Solidity is easy to start, supports inheritance, library calls, etc. so users can easily write contracts using Solidity.

The latest version of Solidity been published is version 0.4.21, it’ s not a stable version, and the existing issues of standardization and expansion in Ethereum smart contract are still unsolved. For these reasons, Solidity has some weaknesses compared to other high-level languages, which are:

1) Solidity cannot directly fetch the original content in Calldata, Callcode;

2) The variable-length type Bytes and the fixed length type Bytes32 Uint256

cannot be directly converted in Solidity. Even though 256bit of memory space has been allocated in Bytes, it can only be filled byte by byte in Solidity, which is a huge waste of Gas.

These issues in Solidity can be improved or solved using Assembly, which will reduce gas consumption and raise execution efficiency of code.

7.1.4 Assembly: Usage & Risks

As a high-level language, Solidity performs work by its compiled bytecode, so just like other high-level languages, users have the right to embed bytecode directly in Solidity to write the contract logic.

In Solidity, Assembly code block can be used to embed with bytecode. To increase the readability of the code, most bytecodes within Assembly can be written in form of function, and this avoids the difficulty of writing bytecode in a direct way.

To the problems of Solidity mentioned above, they can be effectively solved by Assembly code:

1) Direct access to original data in TX is allowed due to the direct use of opcode (such as Calldataload, Calldatacopy, etc.)

2) For variable-length arrays (such as Bytes, String) that have been allocated space, you can assign values directly by locating their data segments , eg. :

```
1.  bytes32 b32 = 0x12345678;
2.  bytes memory b = new bytes(32);
3.  assembly {
4.      mstore(add(b, 32), b32)
5.      re := add(b, 32)
6.  }
```

"32" added here is because the first Slot in the variable-length array in Memory is the length. Similarly, to read data from an access dynamic array and stored in the corresponding fixed length type or normal type, you need to:

```
1. assembly {
2. b32 := mload(add(b, 32))
3. }
```

In addition, viable-length types can convert with each other directly in Solidity, and the same is true for fixed length types (different lengths may truncate or zerofill).

By using Assembly, some restrictions of Solidity can be bypassed, so there is no need to use a large amount of Gas to fill the target byte by byte, ultimately saving Gas and improving the efficiency of contract code at the same time.

Solidity, as a high-level language, was designed not just for the convenience of developers to quickly develop applications, but to take security as the prime important consideration, just like any other high-level languages and virtual machine do. In the case of abusing bytecode or machine code to write logic by developers, it is likely to see serious yet hard to find problems such as memory leak, index overflow, and data corruption. In order to make a better balance between security and performance, the widely recognized Assembly code block should be used only under the corresponding condition, to prevent the great reduction of security for a slight performance improvement.

7.2 Virtual Machine

In the current blockchain industry, Ethereum Virtual Machine (EVM) has

been widely used. EVM is used in most of the existing smart contracts that supports execution of code with any algorithm complexity. Blockchain database is jointly maintained and managed by numerous nodes connected to the blockchain network, each node runs EVM to execute the code of smart contracts, such decentralized consistency ensures high fault tolerance of the whole network.

Take the *Call()* function in EVM for example:

First, we call the *Transfer()* function, set *caller* as receiver account, and *addr* as sender account;

Second, create a Contract object and initialize its member variables *caller*, *self(addr)*, v*alue*, and *gas* ;

Third, assign the member variables *Code*, *CodeHash*, and *CodeAddr* of the Contract object;

Lastly, call the *run()* function to execute the contract's instructions, and then return from the *Call()* function.

The whole implement processes can be well adapted to the current Ethereum block structure.

Relevant code for reference:

```
1. func (evm *EVM) Call(caller ContractRef, addr common.Address, input []byte, gas uint64, value *big.Int) (ret []byte, leftGas *big.Int, error){
2.     var snapshot = evm.StateDB.Snapshot()
4.     contract.SetCallCode(&addr, evm.StateDB.GetCodeHash(addr), evm.StateDB.GetCode(addr))
5.     ret, err = run(evm, snapshot, contract, input)
6.     return ret, contract.Gas, err }
```

In the manner of simplicity, deterministic, and security, EVM was designed specially for blockchain system. But it is incompatible with the current mainstream technologies and design paradigm, there are some

design and implementation flaws exist. The current smart contracts with EVM have a low execution efficiency, in terms of realization of cryptographic computation, you won't choose Solidity but go, C/C++ or other languages with high efficiency. Meanwhile, on memory allocation model, the use of the allocated memory by function in smart contract depends entirely on the programmer to check, if you reuse temporary memory without a complete testing, contracts could be facing potential bugs. In addition, smart contracts in EVM are hard to debug and test, the only condition EVM would report an error is the lack of gas. There is no debug logging in the execution of contracts, nor can it call external code. Furthermore, there is also a lack of standard library, which means for most of the contract developers, they could only continuously copy and paste code from open-source software under such circumstance. Deploying and running smart contracts requires a large amount of gas consumption in the Ethereum network, which not only makes it difficult to write good code, but also makes it very expensive.

EVM, as the first mature and widely used virtual machine in blockchain network, is a pioneer in this field. But given its existing problems, optimization and improvement must be done to have better performance, and that's what Usechain Virtual Machine aims to do based on the specific advantages of identity chain, to make the function of the blockchain virtual machine more powerful, more easier to use, and more safer.

In the early stage of the Usechain project, we continue to use the VM environment based on EVM, writing and running the Ethereum smart contract is very simple and applicable to develop some primitive application. In the later stage of the project, we will build Identity Virtual Machine (IVM). IVM can be defined and sandboxed with a small amount

of adaptation, which will be a new type of standard to write high-performance smart contract. Identity chain will have the intermediate communication layer been built, so that IVM can interact with the bottom layer of blockchain on the basis of the intermediate communication layer. In this way, it makes up the shortfall of the Ethereum Virtual Machine, enables to build high execution efficiency smart contracts which are allowed to interact with the external data, as well as expanding the application scenarios of smart contracts. In order to reduce the vulnerability of smart contract, we will redesign the IVM compiler with built-in contract vulnerability detection function to report every potential bugs in contracts.

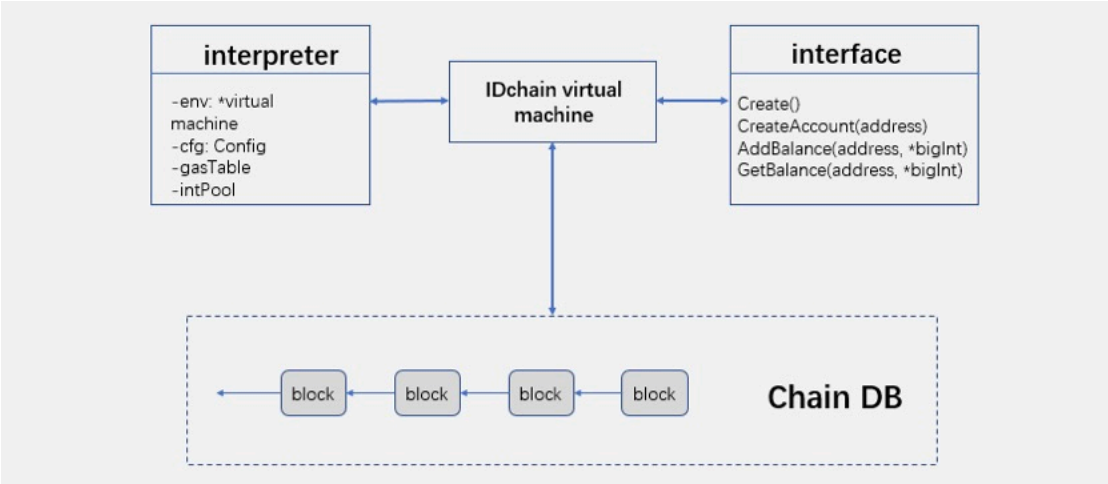


Figure 7-3 Identity Virtual Machine

In the future, we will possibly use WASM as development standard to develop the Usechain Virtual Machine. WASM is a new type of web standard which enables to build high-performance web applications, it could be clearly defined and sandboxed by a small amount of adaptation as well. The advantages of WASM is that it has obtained widely supports in the industry, so developers can develop smart contracts using languages they expert in, such as Go or C languages.

8. Light Nodes Protocol

In the blockchain network, the full amount of the blockchain data will be huge, but for the average user, he is only concerned with relevant information to him other useful information. So for the light client end, there is no need to synchronize all the blockchain data. However, light clients need to ensure the reliability of data in real time. To achieve this goal, our project will use lightweight Meckerl proofs, so that the whole system does not have to rely on all miners on every node, and the miners can also minimize their work of proof.

In the bitcoin system, a light client can download only the header of each block, each block contains only the hash value, the time stamp, the difficulty of mining, transaction blocks, random numbers and the root hash value of the Merkel tree, and its size is 80 bytes. Bitcoin light client has the information of a light block header, it can prove that a transaction is occurred, but cannot provide the information on the current state (the balance of digital assets, and so on). In order to provide the balance information of a node, it is necessary to be able to continuously retrieve information from the blockchain of the entire nodes.

In some application situations of Usechain, it is necessary to quickly obtain the address for the balance, and the proof of an identity. To solve

this problem, with reference to the existing Ethereum implementations, we adopt Merkel - Patricia tree [12] (Merkle Patricia Tree, MPT) data structure to store the transaction information, and the receipt status. So each blockchain header contains not only the root data of a Merkle tree, but the root data of three Merkle trees.

The light node will save the header information for each block, send a data request (via Bloom filter) to the entire network node through the P2P network, and store only a small amount of transaction content to be verified. These transaction contents are all transactions corresponding to the private key in the wallet. This way we can significantly reduce blockchain data storage. At the same time, Usechain will based on the cryptography principles , optimize the data storage structure and further improve the efficiency and security of data.

Light node data query and verification:

1) If the light node wants to get the status of an account: random value, balance, you can recursively downloads the transaction tree from State Root until it finds the result;

2) If the light node wants to verify whether a transaction has been confirmed, it can go to the neighboring P2P network to inquire the block where the transaction is located and download the transaction tree of the State Root. Based on the calculated transaction Hash, a Meckel-Patricia tree root is obtained together with the Hash corresponding to the

transaction tree. The comparison of the two values, if the same, can verify that the transaction is confirmed and at the same time the number of blocks that can be confirmed.

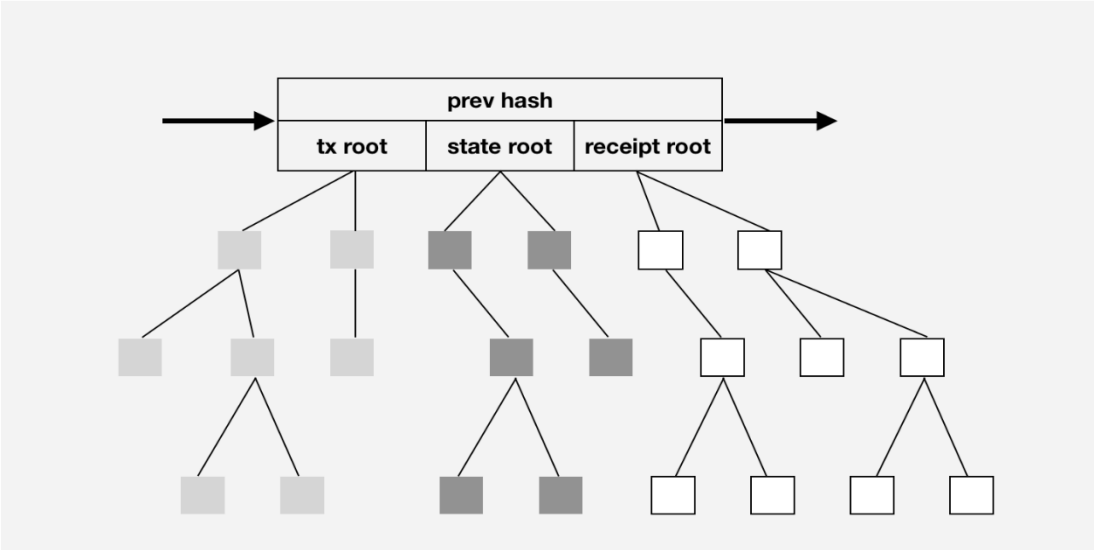


Figure 8-1 Merkle Patricia State Tree

9. General function

9.1 Voting system

Usechain establishes a voting system that allows the entire community to vote for consensus to determine which nodes are selected as the block production committee, which features should be implemented, and what order should be implemented. Anyone can initiate a vote according to his own needs. The person who initiated the vote needs to determine the content of the vote and the voting period (associated with the number of blocks). Users can use it to solve all problems, such as selecting new icons. Adding new features must be decided through

voting by shareholders, and voting can also be used to decide whether to destroy (freeze) particular coins, especially thief or hackers' coins, or even to determine which node to stop the malicious attack through the ballot vote. Further, the community can vote to decide whether to initiate a vote to consider individual users or nodes.

Voting is calculated based on the number of USI/USN owned. Users with more USI/USN coins, such as centralized exchanges or mine pool owners, have greater voting power in this system. To put some restrictions on the voting power, an upper limit for the voting power will be created. It may be difficult to achieve for an anonymous blockchain, but for the real-name identity used in Usechain, we can define the upper limit of the number of votes for each identity.

The Usechain voting system is one of the important components of decentralized currency. There is no leader, no centralized entity, and all decisions depend on the democratic vote. Moreover, in addition to solving global problems, voting systems can also be used by asset owners to trade. It can help shareholders to reach consensus.

9.2 Malicious Address Discovery and Punishment Mechanism

Usechain is an identity public blockchain, and all addresses in the chain are authenticated. A user by zero-knowledge proof scheme, provides

Usechain the identity proof already certified by a third-party certification body. There is no identity information for the account on the chain. In most cases, the user's identity cannot be tracked . However, if there are some evil acts in the chain, such as the behavior of piracy, malicious miners, etc., a special review committee will initiate a proposal to deal with illicit actions. After obtaining the consensus from more than half of the users in the entire network, they will vote to freeze the fund with the relevant address. At the same time, the review committee unlocks the identity verification through the malicious address, submits an application for assistance to the third-party identity verification agency, inquires the true identity information of the malicious user, and applies for the punishment under relevant law.

Summary

As the most promising blockchain ecosystem, Usechain will combine the advantages of public and private chains, solve the inherent defects of the existing blockchain systems and build an ecosystem that is based on true identity. Usechain will continue to develop and iterate through the research and development efforts for its basic platform, as well as the development and commercialization of various product and commercialization projects. It will gradually form a blockchain-based identity economy, improve industry efficiency, and promote the fast, collaborative development for our society.

Appendix

1. Elliptic Curve Cryptography

Elliptic curve cryptography was created independently by American scholars Neil Koblitz and Victor Miller in 1985[2]. In the RSA and ElGamal encryption schemes, a basic security level can be achieved using a 1024-bit modulus. For the elliptic curve (ECC) encryption scheme[13, 14], a it requires only 160 bits in length for a similar cryptographic attack to achieve the same level of security.

Assume G represents a finite field on which an elliptic curve is defined. Actually, this curve can be expressed as a collection of points E, plus an infinity point. The set formed together is regarded as an elliptic curve determined by this equation.

$$E/G: \{(x,y)|y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, a_1, a_2, a_3, a_4, a_6, x, y \in G\} \cup \{O\}$$

In the elliptic curve we define the addition operation , $P(x_1, y_1), Q(x_2, y_2)$ are two points on the elliptic curve E , then the addition of these two points has $P + Q$, R is (x_3, y_3). Her R represents the interaction point of a line passing through points P and Q and the curve E.

Since P, Q, $-R$ are on the same line, so the equation can be expressed as：

$$y = kx + b$$

If $P \neq$ Q(P, Q two points are not overlapping) , the slope of the line：

$$k = (y_1 - y_2)/(x_1 - x_2)$$

If $P = Q$,(P, Q overlapping) , the line that will to tangent the elliptic curve has a slope

$$k = (3x^2 + 2a_2x + a_4 - a_1y)/(2y + a_1x + a_3)$$

Since

$$(kx + b)^2 + a_1x(kx + b) + a_3(kx + b) = x^3 + a_2x^2 + a_4x + a_6$$

When the third order is ; $-x_1x_2x_3$ is the constant coefficient , $x_1x_2 + x_1x_3 + x_2x_3$ is the coefficient for the first order , $-(x_1 + x_2 + x_3)$ the coefficient for the second order.

Hence：

$$\begin{cases} x_3 = k^2 + ka_1 + a_2 + x_1 + x_2 \\ y_3 = y_1 - k(x_1 - x_3) \end{cases}$$

2. Secp256K1

Secp256k1 is the elliptic curve based on F_p the finite field [1], due to the special construction, its performance after optimization can be 30% higher than the performance of other curves, there are two obvious advantages:

1) take up very little bandwidth and storage resources, the length of the key is very short;

2) All users can use the same operation to complete the domain operation.

Bitcoin and Ethereum use a special elliptic curve based on secp256k1 standard. This standard is established by the US National Institute of Standards and Technology (NIST). The secp256k1 curve is defined by the following function, which produces an elliptic curve:

$$E: y^2 = x^3 + ax + b \ over \ F_p$$

Among them F_p is prime field , p is a prime number , G is a basis point。The parameters in Secp256K1 are set as follows:

= FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F

= 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

= 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000007

Compressed basis point is

= 02 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798

And the uncompressed form is:

= 04 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8

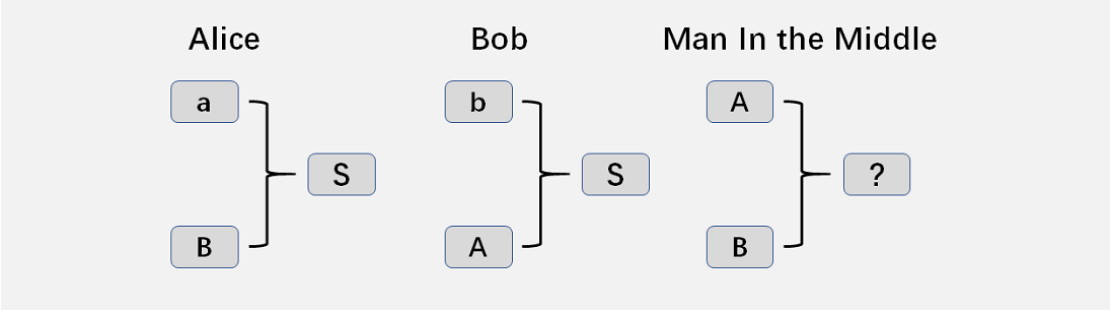
3. Elliptic Curve Diffie-Hellman key Exchange, ECDH)

ECD H is an anonymous Key-agreement protocol. Under this

agreement, both parties use the Diffie-Hellman key exchange algorithm to use a public key and private key pair established by elliptic curve encryption to establish secure shared secret data in an insecure channel. This is a variant of the Diffie-Hellman key exchange, using elliptic curve encryption to enhance security.

Suppose Alice and Bob each generate public and private key pairs (a, A), (b, B), then $A = [a]G$, $B = [b]G, G$ for the same elliptic curve base point。. Alice and Bob calculate separately $S = [a]B$, $S = [b]A$, then :

$$S = [a]B = [a]([b]G) = [b]([a]G) = [b]A$$



Alice and Bob can easily calculate their shared secret S , but others can't figure it out.

4. Ring signature algorithm

The ring signature was first proposed by the three cryptographers Rivest , Shamir and Tauman in 2001. The ring signature belongs to a simplified group signature [15, 16, 19] .

The signer first selects a temporary set of signers, which includes the signer itself. The signer can then use his own private key and the public key of other people in the signature set to generate the signature independently without the help of others. Other members in the signer collection may not know that they are included. The ring signature can be divided into four steps: GEN, SIG, VER, LNK:

1) GEN: Collect public parameters and select randomly, $n-1$, public key,

contract user public key constituting a set of public key

$$\{P_i| = 1,2, \dots, n\}$$

For pair of user public and private keys (P, x), $x \in [1, l-1]$, l is the 点P的阶, generate the public key image I。

2) SIG: For required signature messages m, use a public key set:

$$S = \{P_i| = 1,2, \dots, n\}$$

Among them P_s is the true public key, Calculate the output signature: $ringsig$。

3) VER: based on message m, public key set S and signature $ringsig$, verify the validity of the signature, output True or False

4) LNK: use $J = \{I_i\}$, to verify if the signature was used.

Ring Signature Simple Explanation: A user can check a set of signatures with a set of public keys instead of a unique public key. The identity of the signer is indistinguishable from other users of the public key in the collection until the owner uses the same key pair to generate a second signature. The specific process is as follows:

1) GEN：Signer selects a random number $x \in [1, l-1]$, calculating the public key $P = xG$And public key image $I = xH(P)$。H is a hash function. At the same time randomly select the public key set on the blockchain $S = \{P_i| = 1,2, \dots, n\}$, $P_s = P$。

2)SIG：SIG: Signer Selects Random Number

$$\{q_i| = 0,1,\dots,n, q_i \in [1,l]\}$$

$$\{w_i| = 0,1,\dots,n, i \neq s, w_i \in [1,l]\}$$

Make the following transformation：

$$L_i = \begin{cases} q_i G, & i = s \\ q_i G + w_i P_i, & i \neq s \end{cases}$$

$$R_i = \begin{cases} q_i H_p(P_i), & i = s \\ q_i H_p(P_i) + w_i I_i, & i \neq s \end{cases}$$

The next step is to calculate：

$$c = H(m, L_1, \dots, L_n, R_1, \dots, R_n)$$

The final signer calculates

$$c_i = \begin{cases} w_i, & i \neq s \\ c - \sum_{k=0}^{n} c_i \ mod\ l, & i \neq s \end{cases}$$

$$r_i = \begin{cases} q_i, & i \neq s \\ q_s - c_s x \ mod\ l, & i \neq s \end{cases}$$

The final signature becomes:

$$ringsig = (I, c_1, \dots, c_n, r_1, \dots, r_n)$$

3) VER：When the verifier verifies the signature, based on the message m、public parameters and $S = \{P_i| = 1,2,\dots,n\}$, $ringsig$,

Calculate

$$\begin{cases} L'_i = q_i G + c_i P_i \\ R'_i = r_i H_p(P_i) + c_i I_i \end{cases}$$

The verifier calculates：

$$\sum_{k=0}^{n} c_i = H(m, L'_1, \dots, L'_n, R'_1, \dots, R'_n)$$

If they are equal, LNK is executed. Otherwise, the verifier refuses to sign.

4) LNK：For all that has been on the blockchain I, build set J , If current I appears in the set , it means that the public key has been used, and the signature transaction is considered illegal; if it does not appear, the signature transaction is considered legal, and put I into set J。

References

[1] Wood G. Ethereum: A secure decentralised generalised transaction ledger[J]. Ethereum project yellow paper, 2014, 151: 1-32.

[2] Cooper D. Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile[J]. 2008.

[3] Vukolić M. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication[C]//International Workshop on Open Problems in Network Security. Springer, Cham, 2015: 112-125.

[4] Luu L, Narayanan V, Zheng C, et al. A secure sharding protocol for open blockchains[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016: 17-30.

[5] Schlosser M, Condie T, Kamvar S. Simulating a file-sharing p2p network[C]//1st Workshop on Semantics in Grid and P2P Networks. Stanford InfoLab, 2003.

[6] Li H, Lu R, Zhou L, et al. An efficient merkle-tree-based authentication scheme for smart grid[J]. IEEE Systems Journal, 2014, 8(2): 655-663.

[7] Vukolić M. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication[C]//International Workshop on Open Problems in Network Security. Springer, Cham, 2015: 112-125.

[8] Joux A. A one round protocol for tripartite Diffie–Hellman[C]//International algorithmic number theory symposium. Springer, Berlin, Heidelberg, 2000: 385-393.

[9] Rackoff C, Simon D R. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack[C]//Annual International Cryptology Conference. Springer, Berlin, Heidelberg, 1991: 433-444.

[10] Gervais A, Karame G O, Wüst K, et al. On the security and performance of proof of work blockchains[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016: 3-16.

[11] Goldwasser S, Micali S, Rivest R L. A digital signature scheme secure against adaptive chosen-message attacks[J]. SIAM Journal on Computing, 1988, 17(2): 281-308.

[12] Modified Merkle Patricia Trie Specification. https://github.com/ethereum/wiki/wiki/Patricia-Tree.

[13] Johnson D, Menezes A, Vanstone S. The elliptic curve digital signature algorithm (ECDSA)[J]. International journal of information security, 2001, 1(1): 36-63.

[14] Koblitz N. Elliptic curve cryptosystems[J]. Mathematics of computation, 1987, 48(177): 203-209.

[15] Zhang F, Kim K. ID-based blind signature and ring signature from pairings[C]//International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2002: 533-547.

[16] Liu J K, Wei V K, Wong D S. A separable threshold ring signature scheme[C]//International Conference on Information Security and Cryptology. Springer, Berlin, Heidelberg, 2003: 12-26.

[17] Certicom Research, Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography, Version 2.0, May 21, 2009.

[18] Rahat Afreen, S.C. Mehrotra , A Review on Elliptic Curve Cryptography for Embedded Systems : International Journal of Computer Science & Information Technology (IJCSIT), Vol 3, No 3, June 2011

[19] How to leak a secret, Ron Rivest, Adi Shamir, and Yael Tauman, ASIACRYPT 2001. Volume 2248 of Lecture Notes in Computer Science, pages 552–565.