01

Design concept

"Starting from transaction, with the vision of realizing complex business process and diversified assets on-chain, and carrying COAC ecological business"

As an infrastructure public chain for the digital economy era, COAC Chain aims to use blockchain technology to optimize COAC's strategic layout in different time periods and fields, and help COAC's industrial development in the digital age.
First of all, COAC Chain realizes the transaction and clearing of digital assets under the premise of transparency and supervision through the transaction on the chain. Specifically, in terms of transparency design, through technical means, the on-chain transaction order book, limit order and settlement are realized, so that COAC Chain can self-certify that it has no right to use transaction assets; in terms of supervision, COAC Chain increases In order to achieve efficient transactions on the chain, market-making robots will be added to the transaction chain to match transactions to improve liquidity. sex.
Secondly, with the completion of the transaction design of the public chain, in order to adapt to the future development of the digital economy, COAC Chain will support the on-chain processing of complex business processes and the registration of more abundant asset types. At this stage, the digital asset market is still in its early stages. At the same time, compared with the stock market, the design of digital assets is relatively simple. It can be said that the digital asset market still has a lot of room for development in terms of scale and products. With the further development of the social economy, more mature models of the capital market will gradually be combined with the token economy. In the near future, from currencies to commodities, from real estate contracts to various financial assets, there will be more assets that can be tokenized. Assets in the real world are mapped to the digital world in the form of tokens, which can be traded and managed in the blockchain system, and the market size of digital assets will also expand rapidly. To this end, COAC Chain will develop a new type of token contract to meet the needs of complex business scenarios such as additional issuance and mergers and acquisitions.
Finally, COAC Chain will also become the underlying support for the entire industry ecology of COAC. Many of COAC's businesses such as super node voting, COAC mining pool, COAC wallet, COAC ecological fund, etc. can be processed on the chain; and COAC's business layout in terms of talents and laws can also be based on COAC Chain to develop corresponding DApps. After the community ecology and technical foundation of COAC Chain matures, it will expand a wider range of business scenarios globally.

Technology Architecture

COAC Chain is the first to create a dual-chain architecture, build a dual-chain parallel model of transaction chain and contract chain, and realize the information exchange between the dual main chains through cross-chain technology. The dual-chain architecture ensures that the COAC Chain system can combine high scalability, high security and high efficiency. The transaction chain does not need to support smart contracts. The specific needs in the financial field can be achieved by adding special financial-related transactions, which can increase TPS to a certain extent to meet the needs of high-speed, high-frequency, and low transaction fees on the chain. ; The contract chain mainly supports smart contracts, and the demand for TPS is low to realize complex transactions, business contracts, financial contracts, logic and verification content when the contract is on the chain.
The double-chain structure is mainly based on the relevant design of existing engineering practice cases, such as thunder network's fast and slow chain solution, which can be traced back to the BTC period. The main function of BTC is payment and settlement, but it is limited by the 10-minute block time. In practical applications, the user experience is poor, so a layer of hub-based network is required for relatively faster settlement. As a second-layer solution, the lightning network not only guarantees the settlement function of the BTC main chain, but also increases scalability. It can be seen that the lighting network maintains network transactions of 450 BTC through more than 4,000 nodes and more than 12,000 channels. Based on this technical solution case, in the initial design of the public chain, we can ensure the settlement function through the double-chain structure while adding scalability, and in it, we need to design the relevant settlement public chain and expand the function of the public chain, and SPV (Simplified Payment Verification) and CLTV (Check Lock Time Verify) are used as the main cross-chain methods on the connection channel.
Since the primary goal of COAC Chain is to ensure high-speed transactions, smart contracts consume a lot of effective resources in the system. In order to ensure the breadth of the contract while fast settlement, COAC Chain will adopt a double-chain structure in the design, which is divided into two parts: transaction chain and contract chain:

Transaction chain: Provide payment and settlement functions, such as pending orders, matching, depth and other functions are responsible for it;
Contract chain: support smart contracts, meet the application of complex scenarios, and improve the scalability of the public chain.

Among them, transactions on the contract chain need to be registered on the transaction chain, COAC Chain will provide cross-chain relay to achieve, and the contract chain will provide a built-in contract for unified accounting to be responsible for receiving.
Specifically, in the blockchain, to ensure high-speed characteristics, it is necessary to simplify the transaction KV pair (Key Value Pair), and at the same time optimize the machines of the relevant nodes, otherwise the transaction processing capability (TPS) of the blockchain network will be reduced. It will be reduced due to the loss in transmission and packaging time. Since COAC Chain requires the introduction of a variety of functional features in design, it needs the support of non-Turing script or more advanced virtual machine parts, and some special transactions that can be built-in can be implemented on the terminal at a lower cost. If the method of gas limit or control of node machines is adopted, when any DApp is sought after by the market and becomes popular, it will inevitably affect the operation of the entire public chain. Therefore, COAC Chain will adopt a dual-chain architecture design. In addition, in order to ensure the availability of double-chain data (especially the availability of data on the contract chain), COAC Chain will use the cross-chain technology method of SPV verification + HTLC (Hashed Timelock Contracts)-like main chain locking, and the main chain will be used in the contract. Token, the transaction chain mainly guarantees the KV pair, the balance on the contract chain will only be generated after the transaction chain only has transactions that enter the contract chain. In summary, COAC Chain will ensure that the use of the contract chain will not affect the TPS of the transaction chain through the dual-chain architecture design and the above-mentioned cross-chain technology.
The double-chain architecture of COAC Chain has significant advantages: First, all transactions in the chain can be regarded as changes to global variables. At this time, a static method can be used to ensure a certain scope, and when there are more scopes in the future, The same cross-chain protocol can be used for contact; secondly, after the contract chain has SPV, the steps of payment verification on the contract chain and the main chain can be asynchronous, which greatly enhances the usability and scalability. For example, when a contract chain is occupied by several applications, another contract chain can be opened through the same structure.

02

Blockchain system

Multi-signature account model

Different from the Bitcoin blockchain, COAC Chain introduces the design of a multi-signature account model. Bitcoin has no concept of accounts, each user's balance is calculated from the UTXO (unspent transaction output) on the blockchain, and all legitimate Bitcoin transactions can be traced back to the output of the previous transaction or transactions, The source of these chains is mining rewards, and the end is the current unspent transaction output. All unspent outputs are UTXOs for the entire Bitcoin network. On the COAC Chain, each address corresponds to an account, and the global state consists of a mapping between account addresses and account states, which is stored in the data structure of the Merkle tree. Since COAC Chain has the concept of an account, it has real-time performance in terms of transaction visualization and account status query, and can view the current account status and transaction status in real time according to an address.

digital signature technology

Digital signature is an important part of the blockchain, which ensures the security of the blockchain system. Digital signatures have two important properties. First, only the owner can make their own signature, but anyone who sees it can verify whether it is valid. Second, the signature is only associated with a specific file, which A signature cannot be used to indicate that the owner supports a different document.
The digital signature scheme consists of the following three algorithms:

***(sk,pk):=generateKeys(keysie);**

the generateKeys method takes keysize as input to generate a pair of public and private keys. The private key sk is kept securely and used to sign a message; the public key pk is available to anyone and can be used to verify the signature.

***sig:=sig(sk, message);**

the signature process takes a message and private key as an input, and the corresponding message output is a signature.

***isValid:=verify(sk, message, sig);**

the verification process is to take a message and the signed message and the public key as input, if the returned result is true, the signature is proved to be true; if the returned result is false, the signed message is proved is false.

At the same time, the following two properties are required:

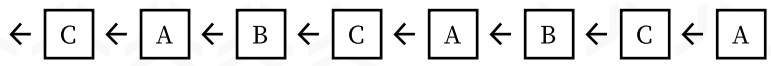
*A valid signature can be verified, ie: Verify(pk, message, sign(sk, message)) == true
*The signature cannot be forged.

COAC- DPoS consensus mechanism

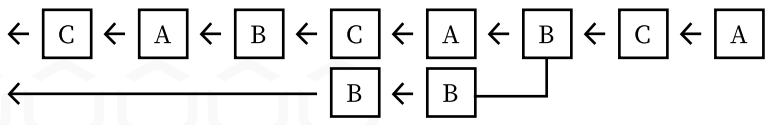
The choice of the public chain consensus mechanism will consider many factors. First, the consensus mechanism generally follows the CAP principle, that is, consistency, availability, and partition tolerance are difficult to achieve optimally at the same time. Secondly, when choosing and designing consensus, it is also necessary to consider the basic nature of consensus: (a) agreement, all honest nodes agree with a result; (b) validity, the agreed result must be a legitimate (c) can be terminated (termination), a consensus must be reached within a certain period of time, and it will not go on endlessly. From the perspective of combining theory and practice, both the contract chain and transaction chain of COAC Chain choose BFT-DPoS (Byzantine Fault Tolerance - Delegated Proof of Stake, Byzantine Fault Tolerance Delegated Proof of Stake) as the consensus mechanism. In order to achieve the goal of supporting transaction clearing, asset on-chain and other multi-functional goals of COAC Chain, the availability of its system needs to be in the first place. Specifically in the CAP principle, the design of COAC Chain needs to ensure the availability and partition fault tolerance of the system, and appropriate compromises can be made for consistency. There is no need to guarantee strong consistency, only the final consistency is required, and the characteristics of the COAC-DPoS mechanism are in line with it.
The main consensus mechanisms of existing blockchain projects are PoW (Proof of Work, workload proof mechanism), PoS
(Proof of Stake, proof of stake mechanism) and DPoS (Delegated Proof of Stake, delegated proof of stake mechanism). From the perspective of efficiency and energy consumption, both PoW and PoS mechanisms have some problems at the design level.
The problems of PoW mechanism mainly exist in the centralization of computing power and energy consumption. On the one hand, the PoW mechanism uses the computing power of nodes to compete for computing power. With the gradual upgrade of CPU mining to ASIC mining, there is a trend of computing power centralization, which is consistent with the concept of blockchain decentralization. Conflict; PoW, on the other hand, wastes a lot of power for computation. The PoS mechanism has improved in terms of energy consumption, but there are still hidden dangers in terms of centralization. Specifically, the PoS mechanism will show a trend that those who hold more coins will receive more coins, and the entire network may become more and more centralized as the running time increases. Therefore, although the PoS mechanism saves energy compared to PoW, its bottom layer still relies on PoW, and it does not improve performance and security well.

The DPoS mechanism is similar to the system in which the board of directors is elected by the general meeting of shareholders, and the super node election mechanism is introduced. This design mechanism makes the generation of blocks faster and more energy efficient. In addition, the DPoS mechanism makes full use of the votes of token holders to reach consensus in a fair and democratic manner. The N super nodes elected by voting have exactly equal rights, and token holders can vote to replace super nodes at any time. Although there is still centralization in the DPoS mechanism, this centralization is controlled because each user has the right to decide which nodes can be trusted. The DPoS mechanism can theoretically reach a transaction speed of 10,000 times per second, and can also reach the level of 1,000 times per second in the case of high networ

k latency, which is more suitable for enterprise-level applications. Since COAC Chain is designed to serve the digital economy, it has extremely high requirements for data exchange and calculation in a trusted environment and its stability, so DPoS is a more suitable choice.
Although DPoS is relatively superior to other mechanisms in terms of performance, its high performance is based on low failure and low latency. However, it is difficult to guarantee low failure and low latency for a long time in the current application scenarios. Therefore, there may be potential problems with non-BFT DPoS mechanisms.
In the DPoS algorithm, block producers take turns generating a block within a certain period of time. Assuming no nodes miss their turn, the longest chain will be produced. Block producers are invalid for any time period outside the scheduled round.
However, there may be cases where malicious or faulty nodes create a small number of forks. In order to ensure that the chain where the honest nodes are located becomes the longest chain, the number of honest nodes is required to account for more than 2/3 of the total. For example, there are three nodes A, B and C, where A and C are honest nodes, and B is a malicious node. It takes 2 seconds to generate a block under DPoS, so malicious node B can only generate one block every 6 seconds. The honest node can generate 2 blocks every 6 seconds, so the chain generated by the honest node is always longer than the attack chain, as shown in the figure.



Schematic diagram of the normal production process of blocks under the general DPoS consensus mechanism



Schematic diagram of the fork of a few nodes under the general DPoS consensus mechanism

Similarly, consider other failure scenarios: such as dual production of offline minority nodes, network fragmentation, dual production of online minority nodes, insufficient number of legal super nodes, fraud by most producers, etc. 1, the number of honest nodes needs to account for more than 2/3 of the total. Therefore, under the traditional DPoS mechanism, in order to prevent forks and ensure the irreversibility of transactions, 2/3 of the super nodes need to confirm by continuing to produce blocks after the block. For example, there are 18 super nodes in a system. The node produces a block every two seconds, so to achieve irreversibility of the transaction, it needs to continue to produce 12 blocks later, which takes a total of 26 seconds (1+12 blocks).
In order to improve the performance of the COAC Chain system, the COAC protocol can be introduced on the basis of the original DPoS to realize the confirmation of the block signature when the block is generated, and shorten the time required for the irreversibility of the transaction. Specifically, the super node packages the transaction into a block and signs the block with its own private key, and broadcasts it to all nodes. When the super node receives at least 2/3 of the signed blocks of other super nodes, the The block completes the verification of all nodes and becomes an irreversible block added to the blockchain. Since each block is broadcasted to the whole network immediately after production, the production of the new blockchain and the receipt of confirmation of the old block can be carried out simultaneously. Therefore, from the moment a block is generated to become an irreversible block, it only takes the block to be generated at the longest. plus the time for signature confirmation of other supernodes (according to the EOS team testing, the process can be completed within 1 second), continuing the above example, it only takes 3 seconds. Under the COAC-DPoS mechanism, the shortening of the block generation interval of the system reduces the delay of cross-chain communication, and at the same time, the number of transactions that can be confirmed per unit time is increased, which improves the overall performance of the blockchain system.

Contract layer design

The contract layer of COAC Chain is composed of multiple built-in contracts and smart contracts customized for transactions, and on this basis, cross-chain relay is realized, so that the two chains of COAC Chain can achieve credible cross-chain interaction. COAC Chain-VM uses WebAssembly (WASM) to execute smart contracts. WASM can support a variety of programming languages, uses binary encoding, takes up less storage space, and has superior performance during program execution. WASM will generate intermediate language - bytecode, which can be compiled using the compilation tools provided by COAC Chain. When calling the contract, the deployment interface deploys the bytecode on the chain. After successful deployment, a smart contract account will be created on the blockchain, in which the bytecode of the contract and the corresponding ABI (Application Binary Interface) are stored. The user uses the ABI to interact with the smart contract through the specified contract account name and contract method to realize the call to the smart contract. Finally, in order to prevent the problems caused by the failure of contract logic execution, COAC Chain will refer to the process of Ethereum and use require and assert to solve it.

KYC&AML

Digital identity

To implement KYC and AML management on the transaction chain of COAC Chain, a credible digital identity standard H-UID (COAC Chain - User Identity) must first be established on the chain. H-UID is unique. Citizens of various countries register their personal information on the chain after KYC authentication. Users can manage their personal information and digital assets based on H-UID.
H-UID consists of the following parts:

- A. Basic information, such as name, gender, nationality, certificate type, certificate number, contact information, etc.;
- B. Advanced information, such as credit, education, work, social and other related data;
- C. Digital asset information, digital assets held by individuals;
- D.Account public and private keys, which are used to sign, encrypt and authorize C-UID data.

Note: Institutional accounts must be associated with the identity of a legal person. One legal person can register multiple institutional accounts.

Creation and verification of C-UID

The user submits information to create a C-UID, and the supervision node verifies the authenticity of the information. After the verification is passed, the verification content is signed, and the personal information is encrypted and registered on the chain. In the verification cycle of C-UID, the verification judgment is triggered when the C-UID needs to be used, and the re-verification period is 6 months. Generally, no verification is required.

Data authorization

In order to better protect personal privacy, except the supervisory node has the right to view the personal data of C-UID, any other
A person or institution can view the data of other people's C-UID only under the premise of obtaining their authorization. When users authorize others to view data, they can set the authorized user, authorization time, specific purpose and other elements in the smart contract, and require the authorized person to use the relevant data only in a trusted execution environment. All query records will be registered on the chain for accountability.

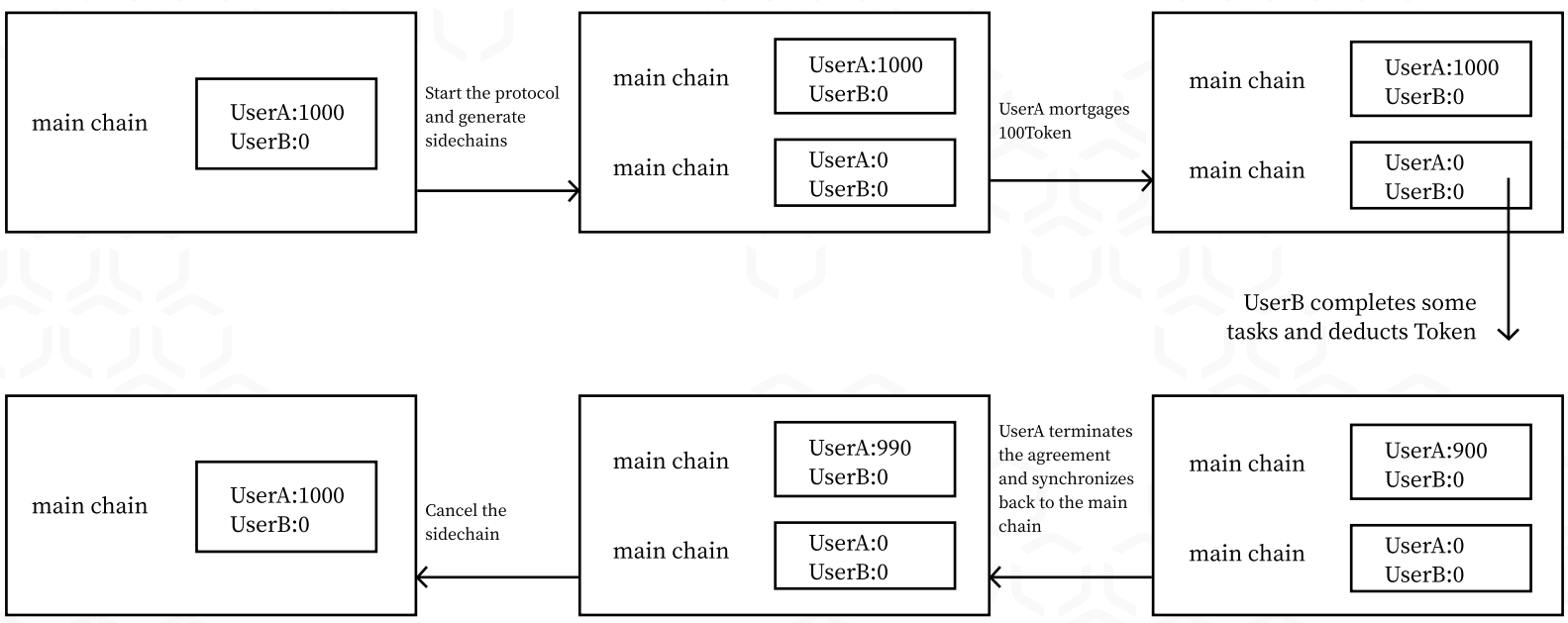
safety protection

In order to protect the security of the user's identity information, on COAC Chain, if the user loses the private key of the data, he will not lose his identity. The user can verify the identity through the supervisory node and reset the private key of the data; in order to prevent the private key from being stolen and published on the entire network , the user himself can also perform .

Cross-chain Technology

COAC Chain adopts a double-chain structure, so it involves cross-chain technology. COAC Chain will develop H Protocol, that is, using the cross-chain technology method of SPV verification + HTLC-like main chain locking. The traditional SPV verification mode often has the phenomenon of inefficiency due to too long confirmation time, but COAC Chain adopts the DPoS mechanism, which is clever This problem is greatly improved, and the extremely fast verification of cross-chain transactions can be completed.
Take the two users UserA and UserB on the transaction chain as an example, where UserA and UserB have a certain number of tokens. UserA and UserB reach an agreement (such as cross-chain data transfer), and then UserA needs to mortgage a certain amount of Tokens and use a side chain to complete the agreement. It is agreed in the agreement that UserB can obtain the token mortgaged by UserA on the side chain after completing a certain task (such as completing cross-chain data transfer). During this process, UserA can check the remaining amount of the pledged Token at any time, and UserB can also check the remaining amount within the range that can be changed at any time. And UserA can decide to terminate the entire agreement at any time. After UserA terminates the agreement, the remaining mortgage will be returned to UserA's main chain account, and UserB will also obtain UserA's Token deducted during this agreement.

For example, initially, the number of tokens owned by UserA and UserB is {UserA:1000, UserB:0}. First, UserA starts the protocol and generates a sidechain; then, UserA and UserB reach a cross-chain data transfer agreement, and UserA mortgages 100 tokens to the sidechain. UserB deducts 10 tokens on the side chain after completing some cross-chain data transfer. After completing the above process, UserA finds that there is no need to continue the transfer, and chooses to terminate this agreement, and the transfer of the token on the side chain will be synchronized to the main chain. Finally, UserA revoked the entire sidechain. The asset changes of UserA and UserB during the whole process are shown in Figure 6:



The above example is a simple contract between UserA and UserB. We can apply this protocol to multiple roles in the same way (UserB can be multiple users). The following describes the working process of the entire protocol with a specific example:

- A.Initialization: Suppose there are four users A, B, C, D, user A sends a special transaction Tx.init to initialize a data, including the permission table {user B: modify, user C: modify, user D: readonly} and Mortgage list {User B: 100, User C: 50} (that is, 100 Tokens are mortgaged to User B, and 50 Tokens are mortgaged to User C). During the special transaction process, User A's 150 Tokens are locked (deducted), and their value is the sum of the mortgage list;

- B.KDC (Key Distribution Center, Key Distribution Center) takes this Tx.init and saves the file ID, including the permission table {user B: modify, user C: modify, user D: readonly} and mortgage list {user B : 100, userC: 50};
- C.User B sends a modification HTTP request req-write to KDC. KDC judges that it has permissions according to the permission table, records the modification records of user B, and returns success;
- D.User C sends a read HTTP request req-read to KDC, KDC calculates the final value according to the initial value and all modifications, and returns it to user C;
- E.User A sends a termination HTTP request req-terminate to KDC, KDC stops serving the requested fileid, and then KDC sends a special transaction Tx.terminate to return User A's remaining mortgage amount and increase the user amount in the mortgage list.

Distributed storage

At present, distributed storage is an essential module of the new public chain. Since the gas limit per block of EVM of 4,700,000 can only accommodate 62kb of storage space, it is far from meeting the needs of users to establish static links or store original documents on the chain. Distributed storage can provide appropriate options for different scenarios. By storing data outside the chain in a distributed manner, it can not only make reasonable use of the storage resources of the main chain, but also allow the data to have a choice between private storage and public access.
To this end, COAC Chain uses decentralized storage developed by third parties to speed up development. In order to ensure the security and privacy of users, COAC Chain requires the decentralized storage provided by third parties to use the storage architecture solution of the serverless interactive system.

The selection of the above scheme is mainly based on the following points: (1) If only the use chain is involved, the plaintext that can be verified needs to be provided, that is, the entire file is disclosed to all users; (2) If only A's key pair is used for encryption, The contract can only be consulted after each time A is inquired, and there will be a single point of failure risk that the contract will not be recognized; (3) If only distributed storage is used, the file (as a distributed network resource) will still be in the form of plaintext. Stored in each node, this scenario generally occurs in the distributed storage of web resources. For the scenario of direct access to resources, a serverless resource service system can be guaranteed through plaintext access, but there are still all plaintext visible or only in interaction. For A-visible issues, A's files are still at risk of a single point of failure.

On the distributed storage implementation path provided by third parties, COAC Chain requires the use of DHT (distributed hash table) as the P2P communication structure, which is also a very mature technical solution for distributed storage. Specifically, COAC Chain requires the use of sentinel (SPoR) in PoR (Proofs of Retrievability) to provide timed storage of heartbeats to ensure that files can be retrieved after uploading; the fingerprint group is generated by engineering optimization of the blockchain version of sentinel , and provide it to the chain side through the relevant heartbeat generation time attribute to ensure block evolution. As the most widely used solution to trackerless, DHT has been maturely used in BitTorrent, kad network, and Ethereum (only including the part of communication to find neighbors).
COAC Chain will continue to use the third-generation DHT—Kadmilia as the structure of the P2P overlay network. The entire network is designed in the way of distributed storage of Key Value, accurate search through Key, and finally download through multiple XOR distance jump addressing. A Kad network with 2^n nodes is guaranteed to find the searched node or value in at most n steps in the worst case. In terms of ensuring downloads, PoR is also a method to ensure data integrity in cloud storage. Compared with PDP, PoR not only ensures data integrity, but also ensures data recoverability.

Supervisory Nodes

Supervision nodes mainly exist in the transaction chain, and supervision nodes are not introduced in the contract chain. The nodes of the blockchain can be simply divided into two categories: full nodes (non-mining) and block producing nodes. The main function of a full node is to trade and download blocks for signers to ensure the availability of the blockchain system. The main function of a block producer is to verify, package blocks, and allow other full nodes to download blocks.

Based on technical characteristics, the supervisory node can only be placed in the block-producing node during design. Because the full node has already downloaded the block, even if some special operations are performed on the reading of the block, transactions that do not want to be supervised can be avoided through technical methods. The supervision of writing nodes is relatively easy, as long as the blocks need to be supervised and encrypted with supervision keys before they are packaged: by setting the memory pool, the common block producing nodes and supervision nodes are both in the process of packaging. The source transaction can be seen, and after the chain is on, the ordinary block-producing node can only see a subset of the block content of the supervisory node. This not only satisfies supervisory nodes to view all information, and only supervisory nodes can view this information (non-supervisory nodes cannot), but also ensures that every record of relevant information is on the chain to facilitate supervision by different supervisory agencies.

On the transaction chain of COAC Chain, the block-producing nodes are mainly composed of nodes of COAC and regulatory authorities, while the general block-producing nodes are nodes controlled by COAC. In terms of specific implementation, the supervision node mainly has the following three functions:

Supervisory nodes can view the correspondence between KYC information and blockchain addresses, and generate a corresponding series of blockchain addresses through KYC information itself (such as name, photo, ID number, etc.) decentralized storage and KYC to achieve the corresponding relationship on the chain;

The supervisory node can verify the transaction entrusted book (order book), which is mainly realized by smearing the chain, that is, encrypting the data first and then uploading it to the chain. The supervisory node has the relevant key and can view it at any time, while the node without the key cannot View related information;

Supervision can view asset traceability, and other nodes have no authority. It is mainly realized by settlement and smearing on the chain, which is verified by supervision nodes.

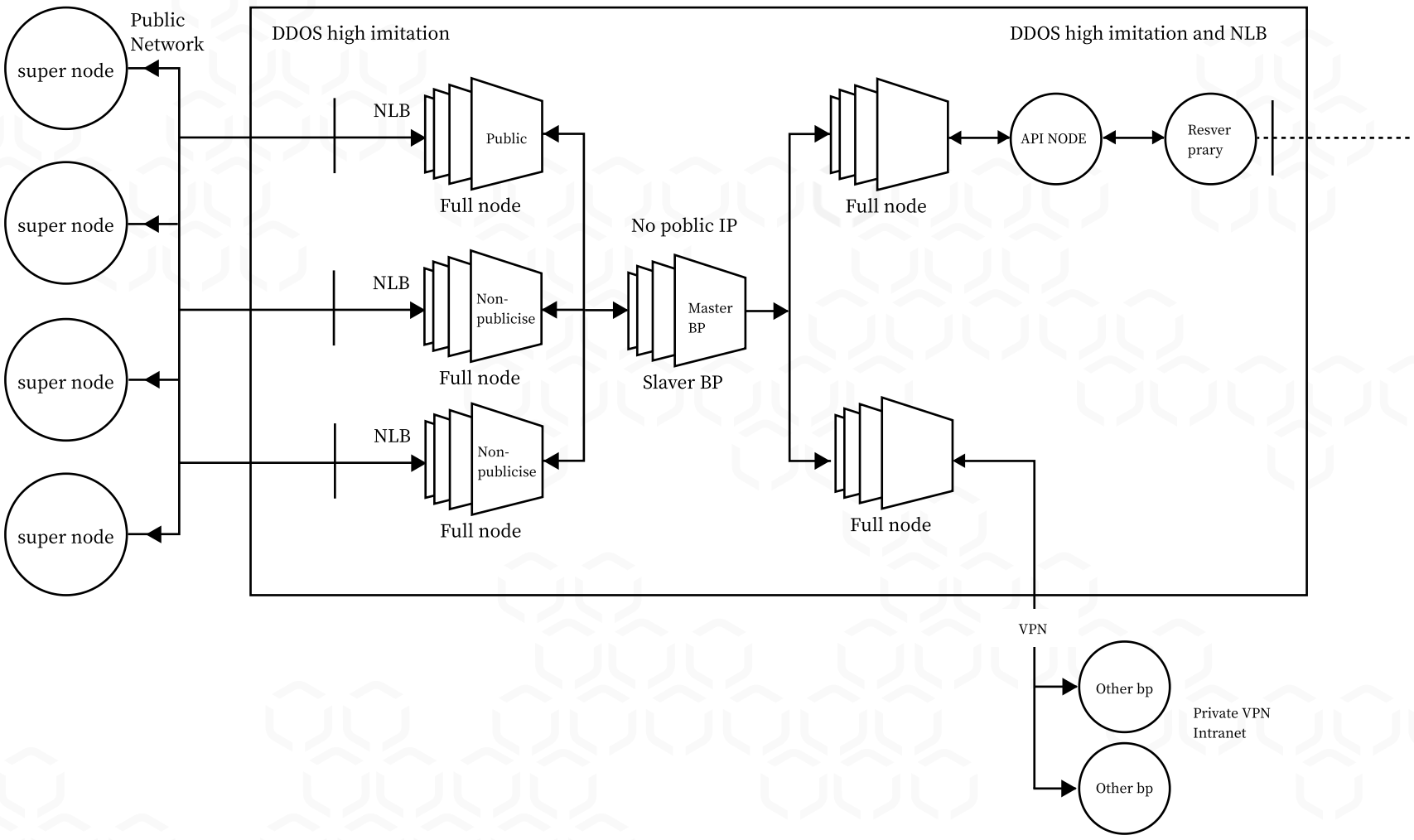
In order to ensure the access, verification, falsification and security of supervisory nodes, supervisory nodes adhere to the principle of independence in design, and plan to use two sets of gateways. In the query and tracking of information data, the supervisory node will design a safe and reasonable management and distribution mechanism. Specifically, COAC Chain will set up a special supervision node to be responsible for the public key authentication and authorization of other supervision nodes, and each supervision node has its own set of public and private keys. KYC is verified and signed by the supervisory node. You can view the user's basic information in the local off-chain database of the supervisory node without your authorization. However, more on-chain data such as advanced information and asset data need to be authorized by the user. to view.

Security

In the era of digital economy, the security of infrastructure public chains is particularly important. COAC Chain's security design mainly focuses on four aspects: security architecture, security audit, smart contract security and security bounty program.

Security Architecture Design

In terms of architecture design, the core goal of COAC Chain is to protect the normal communication and operation of the block server, enhance the overall anti-attack capability of the initial main network and protect the security of nodes. To this end, COAC Chain adopts the layout of super node server isolation, multi-hop node and multi-link communication in the architectural design to prevent DDoS attacks and ensure uninterrupted communication of super nodes. The specific architectural design is shown in Figure 7.



Under normal circumstances, the peripheral nodes communicate through the public nodes announced to the outside world; when the attacker attacks the public nodes through the public node list and the public nodes are unavailable, they can communicate through the private nodes; when the public network nodes are all discovered and attacked Attacks by attackers lead to all blocking of the public network full node servers, and finally the private VPN network communicates in the isolated virtual intranet to ensure the normal production of the most basic blocks; the full node where the query RPC is located is completely isolated from the super node and set up defenses , to ensure that the attack on the RPC from the external network cannot affect the super node.

Security Audit

(1)The core objectives of the audit

In terms of security audit, the core audit objectives of COAC Chain are as follows:

A.Audit the security status of super nodes exposed services on the public network, and find common security problems in traditional security;

B.Audit the anti-DDoS capability of the super node architecture;

C. According to the characteristics of this public chain, carry out a customized load attack test to check the robustness of the overall framework.

The core direction of auditing

The core audit direction of COAC Chain mainly includes the following four points:

A.Find loopholes that can cause the entire node to stop producing blocks;

B.The single-point blocking attack caused by architectural defects can lead to the problem of node paralysis;

C.The problem that the server can be remotely attacked and controlled due to the misconfiguration of the service;

D.The leakage of sensitive node information (especially the leakage of the private key of the server's SSH connection on GitHub) and other issues.

Audit content

The core audit direction of COAC Chain mainly includes the following four points:

A.Find loopholes that can cause the entire node to stop producing blocks;

B.The single-point blocking attack caused by architectural defects can lead to the problem of node paralysis;

C.The problem that the server can be remotely attacked and controlled due to the misconfiguration of the service;

D.The leakage of sensitive node information (especially the leakage of the private key of the server's SSH connection on GitHub) and other issues.

Architecture Audit

Whether the super node server is sufficiently isolated from the external network to ensure that if there is a malicious attack on the external network, it will not directly affect the block production of the super node server;

Whether the super node is designed with multi-links to prevent the occurrence of single point failure (or DDoS for a single point) causing the super node to be unable to synchronize with other nodes;

Whether the node has necessary security reinforcement (such as whether the high-defense defense is correctly deployed on the periphery of the core communication node to resist DDoS attacks, and the appropriate deployment of HIDS)

RPC Security Audit

Whether there are restrictions on unnecessary node RPC services;

If the RPC service is enabled, is it disabled unnecessary wallet_plugin,wallet_api_plugin andproducer_api_plugin;
Enable SSL for RPC.

Security Configuration Audit

Whether the Active multi-signature key is configured correctly;
Whether to enable logging, whether to enable more security logging plug-ins under conditions, etc.;
Whether the max-clients parameter configuration is reasonable, and whether it is easy to suffer from the full number of P2P connections, resulting in failure to synchronize;
Whether to start the node program with non-root privileges;Whether to change the default port of the SSH service, whether to configure a whitelist for the server SSH, and set to allow only key (and encrypt the key) login, and prohibit password login.

Security team audit

Infrastructure Audit

Whether the server provider is a high-quality security provider;

 The real open port service audit of the public network IP of the node prevents the operation and maintenance personnel from improperly configuring the service and security rules and causing the vulnerability to be exposed.

Anti-DDoS capability audit

Conduct practical tests against UDP Flood, TCP Flood, etc. of P2P ports (including various mainstream reflection attacks), and use real attack traffic to test the stability of nodes;

Conduct a practical test for the anti-CC attack of the RPC port, and use a large number of attack nodes with high concurrent requests to consume server performance to detect node stability.

Node Vulnerability Audit

The ability of audit nodes to resist the whole network scan and hide the real public network IP;

Audit whether the sensitive information of nodes is leaked on the public network, such as exposure on GitHub, etc.;
Audit the RPC port for malicious calls;
If the node deploys other programs other than the main program of the blockchain network, conduct vulnerability attack and defense audits for third-party programs;

Check whether the node has customized appropriate emergency response plan.

Smart Contract Security

In order to provide the security of smart contracts, COAC Chain will develop relevant contract templates based on business needs. Developers can call according to the interface parameters of the contract. For requests that do not meet the interface requirements, the call will be directly rejected. On the basis of this, the security of the contract is greatly improved.

In addition, in terms of smart contract security testing, COAC Chain will develop a smart contract verification platform specifically for COAC Chain to automatically detect conventional vulnerabilities and quickly and accurately find conventional security issues in smart contracts. At the implementation level, security testing of smart contracts consists of two parts:

A.Before and during: COAC Chain encourages the community to develop a standardized IDE, which will perform code auto-completion and syntax prompts, and can automatically integrate discovered security vulnerabilities, check the code in time, and point out possible security risks;

B. After the event: By automatically scanning the contract code of all DApps in the COAC Chain, using automated testing to match the vulnerability library rules with the contract code of DApps, and promptly announce the DApps contract code loopholes.

Threat Intelligence Bounty Program

COAC Chain will consider cooperating with a professional blockchain security team to implement a security bounty program, giving a certain amount of token rewards to teams or individuals who provide threat intelligence reports.

Processing flow

A. Reporting stage:
The reporter visits the secure website, enters the "Bounty Vulnerability Submission" page, and submits threat intelligence;

B. Processing stage:
Within one working day, the security team will confirm the received threat intelligence and continue to follow up the assessment;
Within three to ten working days, the technical team will deal with the problem, give a conclusion and score, and communicate with the reporter to confirm if necessary, and ask the reporter for assistance;

C.Repair phase:
The business department repairs the security problems reported in the threat intelligence and arranges the update to go online. The repair time depends on the severity of the problem and the difficulty of repair. Generally speaking, it is within 24 hours for serious and high-risk problems, and within three working days for medium-risk problems. Low-risk issues within seven working days. Client security issues are limited by version release, and the repair time is determined according to the actual situation;
The reporter reviews whether the security problem is fixed;
After the reporter confirms that the security problem has been fixed, the technical team informs the security team of the processing conclusion and vulnerability score, and issues a reward.

Severe vulnerabilities

Serious vulnerabilities refer to those that occur in core systems and business systems (core control systems, domain control systems, business distribution systems, and other management and control systems that can manage a large number of systems) and can cause large-scale impacts. Business system control authority or core system administrator authority and can control the core system. including but not limited to:

A.Block replay verification, the block does not fail to replay due to any factors;

B.System smart contract overflow, conditional competition vulnerability, permission control defect, double spending, consensus layer vulnerability;

C.Sandbox escape causes node command execution or system file reading;

D.The sandbox timeout detection mechanism is bypassed, resulting in DDoS local nodes;

E.Intranet multi-machine control;

F.The super administrator authority of the core background is obtained and causes a large-scale leakage of the core data of the enterprise, which can have a huge impact;

G. The communication layer DDoS other full nodes in a large area at a small cost.

High-risk vulnerabilities

A.System access (getshell, command execution, etc.);
B.SQL injection of the system (background vulnerabilities are downgraded, and packaging submissions are upgraded as appropriate);
C.Unauthorized access to sensitive information, including but not limited to bypassing authentication to directly access the management background, important background weak passwords, SSRF that obtains a large amount of sensitive information on the intranet, etc.);
D.Read any file;
E.Unauthorized operations involving money and bypassing of payment logic (required to be used successfully in the end);
F.Serious logic design flaws and process flaws. Including but not limited to any user login vulnerability, batch modification of any account password vulnerability, logic vulnerability involving the core business of the enterprise, etc., except for verification code blasting;
G.A large number of source code leaks;
H. Defects in smart contract authority control;
I. Abnormal smart contract attacks to avoid exhausting node resources;

J. Invade the server through a full-node program to gain control rights.

Medium critical vulnerability

A.Vulnerabilities that require interaction that can affect users, including but not limited to CSRFs involving core businesses;

B.Ordinary unauthorized operations, including but not limited to bypassing restrictions, modifying user information, performing user operations, etc.;

C. Denial of service vulnerabilities, including but not limited to remote denial of service vulnerabilities that cause denial of service of website applications, etc.;

C.Vulnerabilities that can be caused by successful blasting of sensitive system operations such as arbitrary account login and arbitrary password retrieval due to verification code logic;

E. The sensitive authentication key information stored locally is leaked, and it needs to be effectively used.

Low-risk vulnerabilities

A.Local denial-of-service vulnerabilities, including but not limited to client-side local denial-of-service (crash caused by parsing file formats, network protocols), exposure of Android component permissions, problems caused by ordinary application permissions, etc.;
B.Ordinary information leakage, including but not limited to Web path traversal, system path traversal, directory browsing, etc.;
C.Ordinary CSRF;
D.Reflected XSS (including DOM XSS / Flash XSS);
E.URL jump vulnerability;
F.SMS bombs, mail bombs (each system only accepts one vulnerability of this type);
G.Other vulnerabilities that are less harmful and cannot prove harmful (such as CORS vulnerabilities that cannot obtain sensitive information);
H. Unechoed and not deeply exploited SSRF.

Technical advantages

In terms of technical advantages, COAC Chain will achieve "ease of use + support activity + supervision + security", maintaining high performance and increasing ease of use on the basis of supervision and security.

Ease of use

For users: The current way of generating local private keys in mainstream public chain keystores will bring certain difficulties to users. When COAC Chain is used as the front end of blockchain account, it will adopt a mechanism similar to double login - keystore is aimed at developers and users. For professional practitioners, the centralized mailbox/phone account corresponds to the keystore generated corresponding to the users transferred from the Internet;

For developers: In order to enable technicians to use simpler components when developing smart contracts, that is, common smart contracts can be written by simply splicing templates, COAC Chain will classify common smart contracts in design, and use modules The way of transformation is provided in the template.

Support activity
Development environment and tools: Based on WASM, users can write smart contracts in C, C++, Rust and other languages, with low storage cost and high performance;
High TPS: Through the COAC-DPoS mechanism, COAC Chain is expected to achieve 10,000-level TPS, realize low-latency real-time block writing and query, and the block generation speed can reach the second level;
Asset on-chain: Realize the on-chain of various assets.

Supervisability

KYC&AML: Regulatory nodes can view KYC information;
Asset traceability: The supervisory node can know the traceability details of each transaction, and the other nodes only trace the transaction information.

Security

Adopt security architecture design to ensure the normal communication and operation of the network;
Cooperate with a professional blockchain security team to conduct security audits;
Develop an automated verification platform for smart contracts to ensure the security of smart contracts.

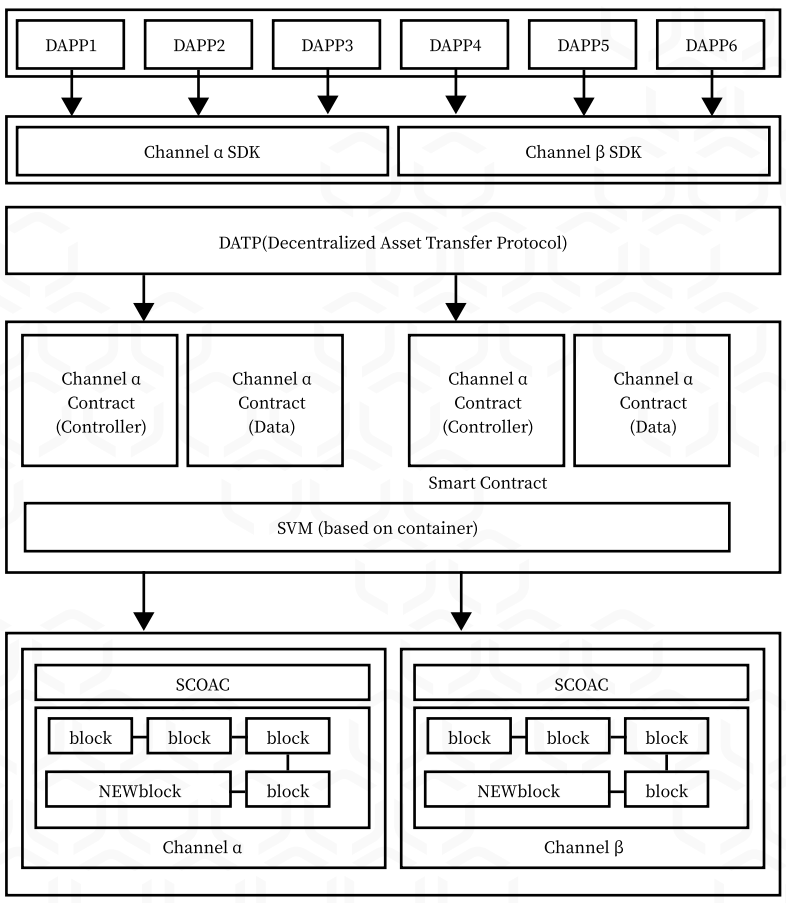
08

International trade application scenarios

International Business Alliance Ecological Chain 1.0 Issuing COAC Token and Providing Services Based on TRON Chain

The COAC Token of the International Business Alliance Ecological Chain will be issued and circulated using the TRC 20 standard of the TRON Chain. Local services and transaction matching services will be implemented based on the TRON chain first. At the same time, the ecological chain development team of the International Business Alliance will begin to evaluate the existing mainstream blockchain underlying technologies, including possible TRON 3.0, MOAC, EOS, Hyperledger, etc., and select one of the technologies for secondary development of the alliance. Chain construction. On this basis, middleware across the underlying platform of the blockchain will also be developed to facilitate subsequent secondary development.
International Business Alliance Ecological Chain 2.0 New Generation Smart Contract Alliance Chain

The International Business Alliance Ecological Chain 2.0 will be a consortium chain, which will support
cross-border payment services and financial services on the basis of the consortium chain. The choice of alliance chain is mainly based on the protection and support of financial and payment data security. After completing the development of the International Business Alliance Ecological Chain 2.0, local services and transaction matching services will be migrated to the International Business Alliance Ecological Chain 2.0 blockchain platform at the same time. In response to the above problems of the current blockchain, the International Business Alliance Ecological Chain will redesign the basic chain structure, integrate the current implementation of basic chains such as Tron and Fabric, and add a series of new features to become the International Business Alliance Ecological Chain 2.0 version.



These features include:

①Use more mature container technology to replace the relatively clumsy EVM virtual machine and build SVM (COAC VM).
②Introduce LLVM and WebAssembly runtime, and support Go, Node.js, Wren and other languages to write smart contracts. After optimization, it is expected to improve the performance by 10~100 times compared to the TRON Chain Solidity runtime.
②Use the improved speculative Byzantine algorithm Zyzzyva (sCOAC) as the main consensus mechanism to solve the problem of PoW energy consumption and the long-term tendency of PoS/dPoS to be centralized.
④In analogy to the concept of multi-tenancy in the Internet open platform, the channel link mechanism is introduced, so that different members/organizations can complete transactions through isolated private block links, and nodes in different links can reach consensus independently , the consensus performance is significantly improved without the need for the entire network consensus.
⑤ Introduce DATP (Cross-chain Digital Asset Exchange Protocol) to achieve consensus on asset transfer between different channels.
⑥ Establish a contract layering mechanism, and divide the original ambiguous smart contract system into three layers: control layer, data layer and business logic layer. The control layer and the data layer run on the chain, the former is equivalent to the business process, and the latter is combined. The business logic layer should run outside the chain, which is equivalent to the SaaS cloud service of the Internet.
⑦Introduce the off-chain encryption applet mechanism CryptoApps to replace the oracle mechanism, so that on-chain smart contracts can safely and efficiently interact with off-chain business logic.
⑧Introduce the WYSIWYG smart contract BPMN (Business Process Model and Notation) editor, which can automatically compile and generate smart contracts on the chain. Dapps developers can build their own business processes without writing contract code.
⑨ Introduce the member service of the alliance chain-like mechanism to manage the identity of the node, and introduce the concept of the witness node. The accuracy of the information on the chain is endorsed by the witness node.
The International Business Alliance Ecological Chain 2.0 is committed to creating a new generation of smart contract platform, adopting a similar alliance chain mechanism, and first landing in the international trade industry. By improving the virtual machine container, contract runtime, consensus mechanism and contract layering, it is expected that the optimized performance can reach 5000~10000tps, which can meet the application deployment needs of the international trade industry in the next few years.

International Business Alliance Ecological Chain Middleware System

KYC&AML

(1)Digital identity

To implement KYC and AML management on the transaction chain of COAC Chain, a credible digital identity standard C-UID (COAC Chain - User Identity) must first be established on the chain. C-UID is unique. Citizens of various countries register their personal information on the chain after KYC authentication. Users can manage their personal information and digital assets based on C-UID.
D-UID consists of the following parts:

A.Basic information, such as name, gender, nationality, certificate type, certificate number, contact information, etc.;

B.Advanced information, such as credit, education, work, social and other related data;

C.Digital asset information, digital assets held by individuals;

D. Account public and private keys, which are used to sign, encrypt and authorize C-UID data.
Note: Institutional accounts must be associated with the identity of a legal person. One legal person can register multiple institutional accounts.

(2)Creation and verification of C-UID

The user submits information to create a C-UID, and the supervision node verifies the authenticity of the information. After the verification is passed, the verification content is signed, and the personal information is encrypted and registered on the chain. In the verification cycle of C-UID, the verification judgment is triggered when the C-UID needs to be used, and the re-verification period is 6 months. Generally, no verification is required.

(3)Data authorization

In order to better protect personal privacy, except the supervisory node has the right to view the personal data of C-UID, any other

A person or institution can view the data of other people's C-UID only under the premise of obtaining their authorization. When users authorize others to view data, they can set the authorized user, authorization time, specific purpose and other elements in the smart contract, and require the authorized person to use the relevant data only in a trusted execution environment. All query records will be registered on the chain for accountability.

(4) Security protection

In order to protect the security of the user's identity information, on COAC Chain, the user will not lose his identity if he loses the private key of the data. The user can verify the identity through the supervisory node and reset the private key of the data; in order to prevent the private key from being stolen and published on the entire network , the user can also modify the private key.

Cross-chain technology

COAC Chain adopts a double-chain structure, so it involves cross-chain technology. COAC Chain will develop H Protocol, that is, using the cross-chain technology method of SPV verification + HTLC-like main chain locking. The traditional SPV verification mode often has the phenomenon of inefficiency due to too long confirmation time, but COAC Chain adopts the DPoS mechanism, which is clever This problem is greatly improved, and the extremely fast verification of cross-chain transactions can be completed.

Take two users UserA and UserB on the transaction chain as an example, where UserA and UserB have a certain number of tokens. UserA and UserB reach an agreement (such as cross-chain data transfer), and then UserA needs to mortgage a certain amount of Tokens and use a side chain to complete the agreement. It is agreed in the agreement that UserB can obtain the token mortgaged by UserA on the side chain after completing a certain task (such as completing cross-chain data transfer). During this process, UserA can check the remaining amount of the pledged Token at any time, and UserB can also check the remaining amount within the range that can be changed at any time. And UserA can decide to terminate the entire agreement at any time. After UserA terminates the agreement, the remaining mortgage will be returned to UserA's main chain account, and UserB will also obtain UserA's Token deducted during this agreement.

For example, initially, the number of tokens owned by UserA and UserB is {UserA:1000, UserB:0}. First, UserA starts the protocol and generates a sidechain; then, UserA and UserB reach a cross-chain data transfer agreement, and UserA mortgages 100 tokens to the sidechain. UserB deducts 10 tokens on the side chain after completing some cross-chain data transfer. After completing the above process, UserA finds that there is no need to continue the transfer, and chooses to terminate this agreement, and the transfer of the token on the side chain will be synchronized to the main chain. Finally, UserA revoked the entire sidechain.

The above example is a simple contract between UserA and UserB. We can apply this protocol to multiple roles in the same way (UserB can be multiple users). The following describes the working process of the entire protocol with a specific example:

A.Initialization: Suppose there are four users A, B, C, D, user A sends a special transaction Tx.init to initialize a data, including the permission table {user B: modify, user C: modify, user D: readonly} and Mortgage list {User B: 100, User C: 50} (that is, 100 Tokens are mortgaged to User B, and 50 Tokens are mortgaged to User C). During the special transaction process, User A's 150 Tokens are locked (deducted), and their value is the sum of the mortgage list;

B.KDC (Key Distribution Center, Key Distribution Center) takes this Tx.init and saves the file ID, including the permission table {user B: modify, user C: modify, user D: readonly} and mortgage list {user B : 100, userC: 50};

C.User B sends a modification HTTP request req-write to KDC. KDC judges that it has permissions according to the permission table, records the modification records of user B, and returns success;

D. User C sends a read HTTP request req-read to KDC, KDC calculates the final value according to the initial value and all modifications, and returns it to user C;

E. User A sends a termination HTTP request req-terminate to KDC, KDC stops serving the requested fileid, and then KDC sends a special transaction Tx.terminate to return User A's remaining mortgage amount and increase the user amount in the mortgage list.

Distributed storage

At present, distributed storage is an essential module of the new public chain. Since the gas limit per block of EVM of 4,700,000 can only accommodate 62kb of storage space, it is far from meeting the needs of users to establish static links or store original documents on the chain. Distributed storage can provide appropriate options for different scenarios. By storing data outside the chain in a distributed manner, it can not only make reasonable use of the storage resources of the main chain, but also allow the data to have a choice between private storage and public access.

To this end, COAC Chain uses decentralized storage developed by third parties to speed up development. In order to ensure the security and privacy of users, COAC Chain requires the decentralized storage provided by third parties to use the storage architecture solution of the serverless interactive system.

The selection of the above scheme is mainly based on the following points: (1) If only the use chain is involved, the plaintext that can be verified needs to be provided, that is, the entire file is disclosed to all users; (2) If only A's key pair is used for encryption, The contract can only be consulted after each time A is inquired, and there will be a single point of failure risk that the contract will not be recognized; (3) If only distributed storage is used, the file (as a distributed network resource) will still be in the form of plaintext. Stored in each node, this scenario generally occurs in the distributed storage of web resources. For the scenario of direct access to resources, a serverless resource service system can be guaranteed through plaintext access, but there are still all plaintext visible or only in interaction. For A-visible issues, A's files are still at risk of a single point of failure.

On the distributed storage implementation path provided by third parties, COAC Chain requires the use of DHT (distributed hash table) as the P2P communication structure, which is also a very mature technical solution for distributed storage. Specifically, COAC Chain requires the use of sentinel (SPoR) in PoR (Proofs of Retrievability) to provide timed storage of heartbeats to ensure that files can be retrieved after uploading; the fingerprint group is generated by engineering optimization of the blockchain version of sentinel , and provide it to the chain side through the relevant heartbeat generation time attribute to ensure block evolution. As the most widely used solution to trackerless, DHT has been maturely used in BitTorrent, kad network, and Ethereum (only including the part of communication to find neighbors).

COAC Chain will continue to use the third-generation DHT—Kadmilia as the structure of the P2P overlay network. The entire network is designed in the way of distributed storage of Key Value, accurate search through Key, and finally download through multiple XOR distance jump addressing. A Kad network with 2^n nodes is guaranteed to find the searched node or value in at most n steps in the worst case. In terms of ensuring downloads, PoR is also a method to ensure data integrity in cloud storage. Compared with PDP, PoR not only ensures data integrity, but also ensures data recoverability.

10 Supervision node

Supervision nodes mainly exist in the transaction chain, and supervision nodes are not introduced in the contract chain. The nodes of the blockchain can be simply divided into two categories: full nodes (non-mining) and block producing nodes. The main function of a full node is to trade and download blocks for signers to ensure the availability of the blockchain system. The main function of a block producer is to verify, package blocks, and allow other full nodes to download blocks.
Based on technical characteristics, the supervisory node can only be placed in the block-producing node during design. Because the full node has already downloaded the block, even if some special operations are performed on the reading of the block, transactions that do not want to be supervised can be avoided through technical methods. The supervision of writing nodes is relatively easy, as long as the blocks need to be supervised and encrypted with supervision keys before they are packaged: by setting the memory pool, the common block producing nodes and supervision nodes are both in the process of packaging. The source transaction can be seen, and after the chain is on, the ordinary block-producing node can only see a subset of the block content of the supervisory node. This not only satisfies supervisory nodes to view all information, and only supervisory nodes can view this information (non-supervisory nodes cannot), but also ensures that every record of relevant information is on the chain to facilitate supervision by different supervisory agencies.

On the transaction chain of COAC Chain, the block-producing nodes are mainly composed of nodes of COAC and regulatory authorities, while the general block-producing nodes are nodes controlled by COAC. In terms of specific implementation, the supervision node mainly has the following three functions:
Supervisory nodes can view the correspondence between KYC information and blockchain addresses, and generate a corresponding series of blockchain addresses through KYC information itself (such as name, photo, ID number, etc.) chain;

The supervisory node can verify the transaction entrustment ledger (order book), which is mainly realized by smearing the chain, that is, encrypting the data before uploading the chain. The supervisory node has the relevant key and can view it at any time, while the node without the key cannot view it. Related Information;
Supervision can view asset traceability, and other nodes have no authority, which is mainly realized by settlement and smearing on the chain, which is verified by supervision nodes.

In order to ensure the access, verification, falsification and security of supervisory nodes, supervisory nodes adhere to the principle of independence in design, and plan to use two sets of gateways. In the query and tracking of information data, the supervisory node will design a safe and reasonable management and distribution mechanism.

Specifically, COAC Chain will set up a special supervision node to be responsible for the public key authentication and authorization of other supervision nodes, and each supervision node has its own set of public and private keys. KYC is verified and signed by the supervisory node. You can view the user's basic information in the local off-chain database of the supervisory node without your authorization. However, more on-chain data such as advanced information and asset data need to be authorized by the user. to view.

On the COAC Chain, if the supervisor wants to obtain the authority and execution capability of the supervisory node, it must maintain the supervisory node. Because the supervision node needs to use its own supervision public and private keys to control the smeared source data. If the supervisor does not maintain the nodes, the security and controllability are difficult to guarantee.
For practical consideration, the maintenance of supervisory nodes in the initial stage is performed by COAC. COAC will proactively provide data to regulators for their oversight.

safety

In the era of digital economy, the security of infrastructure public chains is particularly important. COAC Chain's security design mainly focuses on four aspects: security architecture, security audit, smart contract security and security bounty program.

Security Architecture Design

In terms of architecture design, the core goal of COAC Chain is to protect the normal communication and operation of the block server, enhance the overall anti-attack capability of the initial main network and protect the security of nodes. To this end, COAC Chain adopts the layout of super node server isolation, multi-hop node and multi-link communication in the architectural design to prevent DDoS attacks and ensure uninterrupted communication of super nodes. The specific architectural design is shown in Figure 7.
Figure 7 COAC Chain security architecture design

Under normal circumstances, the peripheral nodes communicate through the public nodes announced to the outside world; when the attacker attacks the public nodes through the public node list and the public nodes are unavailable, they can communicate through the private nodes; when the public network nodes are all discovered and attacked Attacks by attackers lead to all blocking of the public network full node servers, and finally the private VPN network communicates in the isolated virtual intranet to ensure the normal production of the most basic blocks; the full node where the query RPC is located is completely isolated from the super node and set up defenses , to ensure that the attack on the RPC from the external network cannot affect the super node.

Security Audit

The core objectives of the audit

In terms of security audit, the core audit objectives of COAC Chain are as follows:

A.Audit the security status of super nodes exposed services on the public network, and find common security problems in traditional security;

B.Audit the anti-DDoS capability of the super node architecture;

C. According to the characteristics of this public chain, carry out a customized load attack test to check the robustness of the overall framework.

The core direction of auditing.

The core audit direction of COAC Chain mainly includes the following four points:

A.Find loopholes that can cause the entire node to stop producing blocks;

B.The single-point blocking attack caused by architectural defects can lead to the problem of node paralysis;

C.The problem that the server can be remotely attacked and controlled due to the misconfiguration of the service;

D. The leakage of sensitive node information (especially the leakage of the private key of the server's SSH connection on GitHub) and other issues.

Audit content

COAC Chain will focus on self-audit of nodes (many sensitive servers and operations should not be directly exposed to third parties, and need to rely on node self-audit), and the security team will provide professional guidance and cooperation to achieve the most comprehensive and efficient audit. Effect.

11

Node self-audit

Architecture Audit

Whether the super node server is sufficiently isolated from the external network to ensure that if there is a malicious attack on the external network, it will not directly affect the block production of the super node server;

Whether the super node is designed with multi-links to prevent the occurrence of single point failure (or DDoS for a single point) causing the super node to be unable to synchronize with other nodes;

Whether the node has necessary security reinforcement (such as whether the high-defense defense is correctly deployed on the periphery of the core communication node to resist DDoS attacks, and the appropriate deployment of HIDS).

RPC Security Audit

Whether there are restrictions on unnecessary node RPC services;

If the RPC service is enabled, is it disabled unnecessary wallet_plugin,wallet_api_plugin and producer_api_plugin;
Enable SSL for RPC.

Security Configuration Audit

Whether the Active multi-signature key is configured correctly;

Whether to enable logging, whether to enable more security logging plug-ins under conditions, etc.;
Whether the max-clients parameter configuration is reasonable, and whether it is easy to suffer from the full number of P2P connections, resulting in failure to synchronize;
Whether to start the node program with non-root privileges;
Whether to change the default port of the SSH service, whether to configure a whitelist for the server SSH, and set to allow only key (and encrypt the key) login, and prohibit password login.

Security Configuration Audit

Security team audit
A. Infrastructure Audit
Whether the server provider is a high-quality security provider;
The real open port service audit of the public network IP of the node prevents the operation and maintenance personnel from improperly configuring the service and security rules and causing the vulnerability to be exposed.

B. Node Vulnerability Audit
The ability of audit nodes to resist the whole network scan and hide the real public network IP;
Audit whether the sensitive information of nodes is leaked on the public network, such as exposure on GitHub, etc.;
Audit the RPC port for malicious calls;
If the node deploys other programs other than the main program of the blockchain network, conduct vulnerability attack and defense audits for third-party programs;
Check whether the node has customized appropriate emergency response plan.

C. Anti-DDoS capability audit
Conduct practical tests against UDP Flood, TCP Flood, etc. of P2P ports (including various mainstream reflection attacks), and use real attack traffic to test the stability of nodes;
Conduct a practical test for the anti-CC attack of the RPC port, and use a large number of attack nodes with high concurrent requests to consume server performance to detect node stability.

Smart Contract Security

In order to provide the security of smart contracts, COAC Chain will develop relevant contract templates based on business needs. Developers can call according to the interface parameters of the contract. For requests that do not meet the interface requirements, the call will be directly rejected. On the basis of this, the security of the contract is greatly improved.

In addition, in terms of smart contract security testing, COAC Chain will develop a smart contract verification platform specifically for COAC Chain to automatically detect conventional vulnerabilities and quickly and accurately find conventional security issues in smart contracts. At the implementation level, security testing of smart contracts consists of two parts:

A.Before and during: COAC Chain encourages the community to develop a standardized IDE, which will perform code auto-completion and syntax prompts, and can automatically integrate discovered security vulnerabilities, check the code in time, and point out possible security risks;

B. After the event: By automatically scanning the contract code of all DApps in the COAC Chain, using automated testing to match the vulnerability library rules with the contract code of DApps, and promptly announce the DApps contract code loopholes.

Threat Intelligence Bounty Program

COAC Chain will consider cooperating with a professional blockchain security team to implement a security bounty program, giving a certain amount of token rewards to teams or individuals who provide threat intelligence reports.
Processing flow
A. Reporting stage:
The reporter visits the secure website, enters the "Bounty Vulnerability Submission" page, and submits threat intelligence;
B. Processing stage:
Within one working day, the security team will confirm the received threat intelligence and continue to follow up the assessment;
Within three to ten working days, the technical team will deal with the problem, give a conclusion and score, and communicate with the reporter to confirm if necessary, and ask the reporter for assistance;
C. Repair phase:
The business department repairs the security problems reported in the threat intelligence and arranges the update to go online. The repair time depends on the severity of the problem and the difficulty of repair. Generally speaking, it is within 24 hours for serious and high-risk problems, and within three working days for medium-risk problems. Low-risk issues within seven working days. Client security issues are limited by version release, and the repair time is determined according to the actual situation;
The reporter reviews whether the security problem is fixed;
After the reporter confirms that the security problem has been fixed, the technical team informs the security team of the processing conclusion and vulnerability score, and issues a reward.

critical vulnerability

Serious vulnerabilities refer to those that occur in core systems and business systems (core control systems, domain control systems, business distribution systems, and other management and control systems that can manage a large number of systems) and can cause large-scale impacts. Business system control authority or core system administrator authority and can control the core system. including but not limited to:

A.Block replay verification, the block does not fail to replay due to any factors;

B. System smart contract overflow, conditional competition vulnerability, permission control defect, double spending, consensus layer vulnerability;

C. Sandbox escape causes node command execution or system file reading;

D. The sandbox timeout detection mechanism is bypassed, resulting in DDoS local nodes;

E. Intranet multi-machine control;

F. The super administrator authority of the core background is obtained and causes a large-scale leakage of the core data of the enterprise, which can have a huge impact;

G. The communication layer DDoS other full nodes in a large area at a small cost.

High-risk vulnerabilities

A.System access (getshell, command execution, etc.);

B. SQL injection of the system (background vulnerabilities are downgraded, and packaging submissions are upgraded as appropriate);

C. Unauthorized access to sensitive information, including but not limited to bypassing authentication to directly access the management background, important background weak passwords, SSRF that obtains a large amount of sensitive information on the intranet, etc.);

D. Read any file;

E. Unauthorized operations involving money and bypassing of payment logic (required to be used successfully in the end);

F. Serious logic design flaws and process flaws. Including but not limited to any user login vulnerability, batch modification of any account password vulnerability, logic vulnerability involving the core business of the enterprise, etc., except for verification code blasting;

G. A large number of source code leaks;

H. Defects in smart contract authority control;
I. Abnormal smart contract attacks to avoid exhausting node resources;
J.Invade the server through a full-node program to gain control rights.

Moderate vulnerability

A. Vulnerabilities that require interaction that can affect users, including but not limited to CSRFs involving core businesses;

B. Ordinary unauthorized operations, including but not limited to bypassing restrictions, modifying user information, performing user operations, etc.;

C. Denial of service vulnerabilities, including but not limited to remote denial of service vulnerabilities that cause denial of service of website applications, etc.;

D. Vulnerabilities that can be caused by successful blasting of sensitive system operations such as arbitrary account login and arbitrary password retrieval due to verification code logic;

E. The sensitive authentication key information stored locally is leaked, and it needs to be effectively used.

low severity vulnerability

A. Local denial-of-service vulnerabilities, including but not limited to client-side local denial-of-service (crash caused by parsing file formats, network protocols), exposure of Android component permissions, problems caused by ordinary application permissions, etc.;

B. Ordinary information leakage, including but not limited to Web path traversal, system path traversal, directory browsing, etc.;

C. Ordinary CSRF;

D. Reflected XSS (including DOM XSS / Flash XSS);

E. URL jump vulnerability;

F. SMS bombs, mail bombs (each system only accepts one vulnerability of this type);

G. Other vulnerabilities that are less harmful and cannot prove harmful (such as CORS vulnerabilities that cannot obtain sensitive information);

H. Unechoed and not deeply exploited SSRF.

Technical advantages

In terms of technical advantages, COAC Chain will achieve "ease of use + support activity + supervision + security", maintaining high performance and increasing ease of use on the basis of supervision and security.

(1)Ease of use

For users: The current way of generating local private keys in mainstream public chain keystores will bring certain difficulties to users. When COAC Chain is used as the front end of blockchain account, it will adopt a mechanism similar to double login - keystore is aimed at developers and users. For professional practitioners, the centralized mailbox/phone account corresponds to the keystore generated corresponding to the users transferred from the Internet;

For developers: In order to enable technicians to use simpler components when developing smart contracts, that is, common smart contracts can be written by simply splicing templates, COAC Chain will classify common smart contracts in design, A modular approach is provided in templates.

(2)Support activity

Development environment and tools: Based on WASM, users can write smart contracts in C, C++, Rust and other languages, with low storage cost and high performance;

High TPS: Through the COAC-DPoS mechanism, COAC Chain is expected to achieve 10,000-level TPS, realize low-latency real-time block writing and query, and the block generation speed can reach the second level;

Asset on-chain: Realize the on-chain of various assets.

(3) Supervisability

KYC&AML: Regulatory nodes can view KYC information;

Asset traceability: The supervisory node can know the traceability details of each transaction, and the other nodes only trace the transaction information.

(4) Security

Adopt security architecture design to ensure the normal communication and operation of the network;

Cooperate with a professional blockchain security team to conduct security audits;

Develop an automated verification platform for smart contracts to ensure the security of smart contracts.