Nutbox

Technical Whitepaper

Contents

1 Introduction 1
2 Background 2
3 Open Community Staking Protocol(OCSP) Contract 3
3.1 Contract Topology 4
3.2 Multi-Chain tToken 5
3.2.1 tToken Exchange Gateway（TEG） 5
3.2.2 Bridge Proxy Contract 6
3.2.3 tToken Issue Contract 6
3.2.4 tToken Burning Contract 6
3.3 Community Token(cToken) Mining & Distribution 7
4 Crosschain Bridge 8
4.1 What Is Crosschain Bridge 8
4.2 Light Trustless Blokchain State Verification(LTBSV) 9
4.2.1 Block Fetcher 9
4.2.2 Block Validator 9
4.2.3 Deposit Proof 10
5 Nutbox As Parachain 10
5.1 Nutbox Collator 11
5.2 XCM Based Transfer 11
6 Asset Price Oracle 11
6.1 Price Oracle Algorithm 12
6.2 Offchain Worker Based Price Oracle Flow 13
7 Nutbox Governance Contracts 14
7.1 Democracy 14
7.2 Collective 15
7.3 Elections 15
7.4 Membership 15
7.5 Treasury 15
7.6 Multisig 15
7.7 Sudo 15
8 References 16

1 Introduction

For a long time, whether in traditional communities or blockchain communities, community operators have been distressed about how to make the community active. Initially due to the rise of the blockchain token economy, the community will choose to issue its own community tokens to incentivize users participating in the community, and promote the community to have a higher degree of activity and value capture driven by the community tokens. For example, a community running a sharing application issues its own tokens, which can be earned by both sharing and renting items. In this way, community members receive additional financial incentives when using the product.

For now, however, it is not enough for the community to simply issue its own tokens. After the community issues its own tokens, it can play a role in stimulating community activity and increasing community value in the short term. But in the long run, community tokens often fail to bring substantial changes to the community due to the lack of a complete price support system or the lack of a good internal circulation of the economy. Therefore, enabling community tokens to truly circulate in the community has become a problem that the industry urgently needs to solve.

The popularity of the PoS consensus algorithm [1] provides us with a new set of solutions. The PoS consensus algorithm is used to solve the problems of low TPS and poor system scalability under the original PoW consensus algorithm [2] of the blockchain network. Unlike the PoW consensus algorithm, which enables anyone who provides hardware device can access the blockchain network to become a validator, in a blockchain network based on the PoS consensus algorithm, users need to stake a certain amount of network assets, and then become a validator to obtain rewards by verifying the block. When you exit as a validator, you can still get back your staked assets. Under the PoS consensus system, nodes participating in the network no longer need to compete in hardware, which consumes a lot of power resources. This economic model that stakes the original assets of the blockchain network and obtains income on this basis has a good economic closed loop and can make the blockchain network operate stably for a long time under a healthy economic cycle.

Unfortunately, this economic model is currently only used in the consensus system of the blockchain network itself. It does not bring direct value to the applications, that is, the communities running in the blockchain network.
What Nutbox is trying to do is to build an open asset staking protocol that allows the blockchain community to build an economic model of the staking for its community assets in a blockchain network based on the POS consensus algorithm.

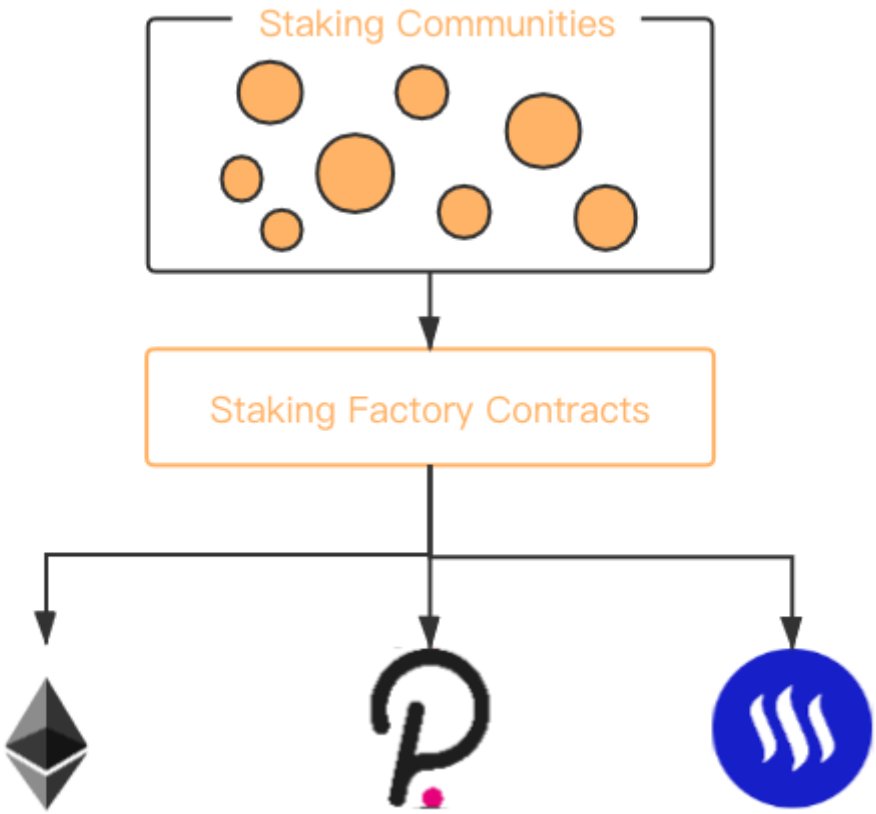

Figure1, A Staking Factory Runing In PoS Blockchain Network

2 Background

The ultimate goal of Nutbox is to support the collateralization of assets across multiple PoS chains and to enable the blockchain community to build their own staking economic business based on the staking model of the PoS chains they are running. Therefore, the Nutbox protocol needs to be deployed into a blockchain network that supports cross-chain interoperability, and we found that this is exactly what Polkadot does.

Polkadot was founded by Dr. Gavin Wood, creator of Ethereum Yellow Book and founder of Parity Inc. Unlike Ethereum2.0, which focuses on using sharding [15] technology to improve network scalability, Polkadot introduces relaychain and parachain.
Relaychain is developed and operated by Parity , and everyone can join as a network validator. The network selects the set of validators for each session based on the NPoS algorithm [3]. The selected validators are responsible for the verification of the block in this round and get rewards.

Parachain is developed by any team or individual. Unlike general blockchain networks, parachain does not have its own consensus system, and parachain blocks are verified by Relaychain. After the parachain block is constructed, it will be broadcast to the Relaychain

validator in PoV format, and the verified block will be stored in the parachain network. Polkadot Relaychain is like the security steward of each Parachain, responsible for the consensus security of each Parachain, so that Parachain can focus on its own business.

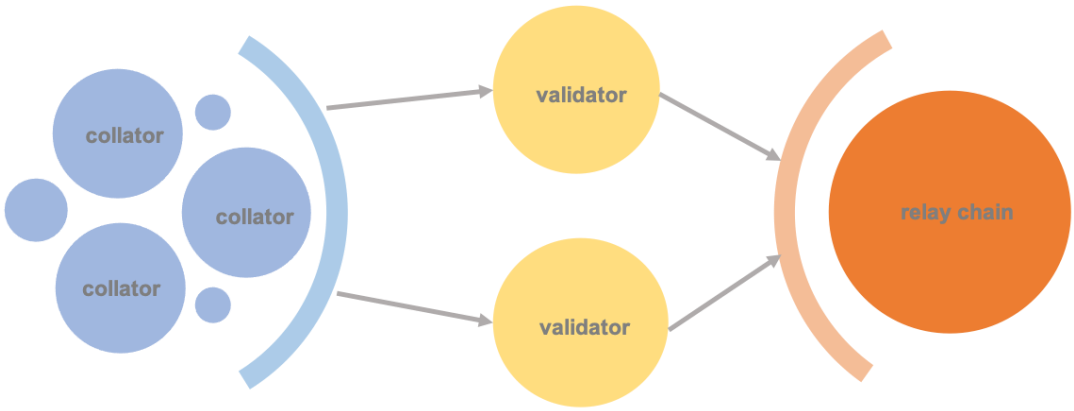

Figure2, A Staking Factory Runing In PoS Blockchain Network

Nutbox will run in either mode of "Nutbox Contracts" or "Nutbox Parachain".

The Nutbox Contracts mode is the operating mode when the Nutbox blockchain has not been developed and the Parachain slot of Polkadot has not been successfully obtained. It will be deployed as a smart contract to the blockchain network that supports EVM.

The Nutbox Parachian mode is that when the Nutbox blockchain obtains a Parachain slot through the Polkadot Parachain slot auction, it connects to the blockchain network as the Polkadot Parachain. It will not have its own consensus module. Instead, Polkadot provides consensus security and validates the blocks of the Nutbox blockchain.

Nutbox will support collateralization of multiple PoS chain assets in both modes, with slightly different implementations in both modes.

Nutbox Contracts

When the Nutbox doesn't run as a Polkadot Parachain, it will be deployed as a contract on a blockchain network that supports EVM virtual machines. At this point all cross-chain functions will be supported by the LTBSV cross-chain bridge protocol. And in the early stage of Nutbox design, seamless migration from contract to Parachain will be considered.

Nutbox Parachain

When Nutbox runs as a Parachain in the Polkadot ecosystem, part of Nutbox's cross-chain asset transfer will be implemented directly using Polkadot Cross-Consensus Message protocol, namely XCM [4]. If the other chain supported by Nutbox happens to be a Polkadot Parachain, then the two chains will directly complete the cross-chain transfer through the

instructions of the XCM protocol. If the other chain is not a Polkadot Parachain, then Nutbox will use the LTBSV cross-chain bridge protocol described below to achieve cross-chain transfer of assets.

3 Open Community Staking Protocol (OCSP) Contract

OCSP is a standard set of contract templates for deploying a user's asset staking business. With the OCSP contract, a tToken-based asset exchange gateway and a cToken distributor are created, that is, the business logic for users to exchange underlying assets and tTokens, as well as obtain cToken income through tToken-based staking. Here, both tToken and cToken represent two asset categories in the Nutbox network and do not refer to a specific asset.

3.1 Contract Topology

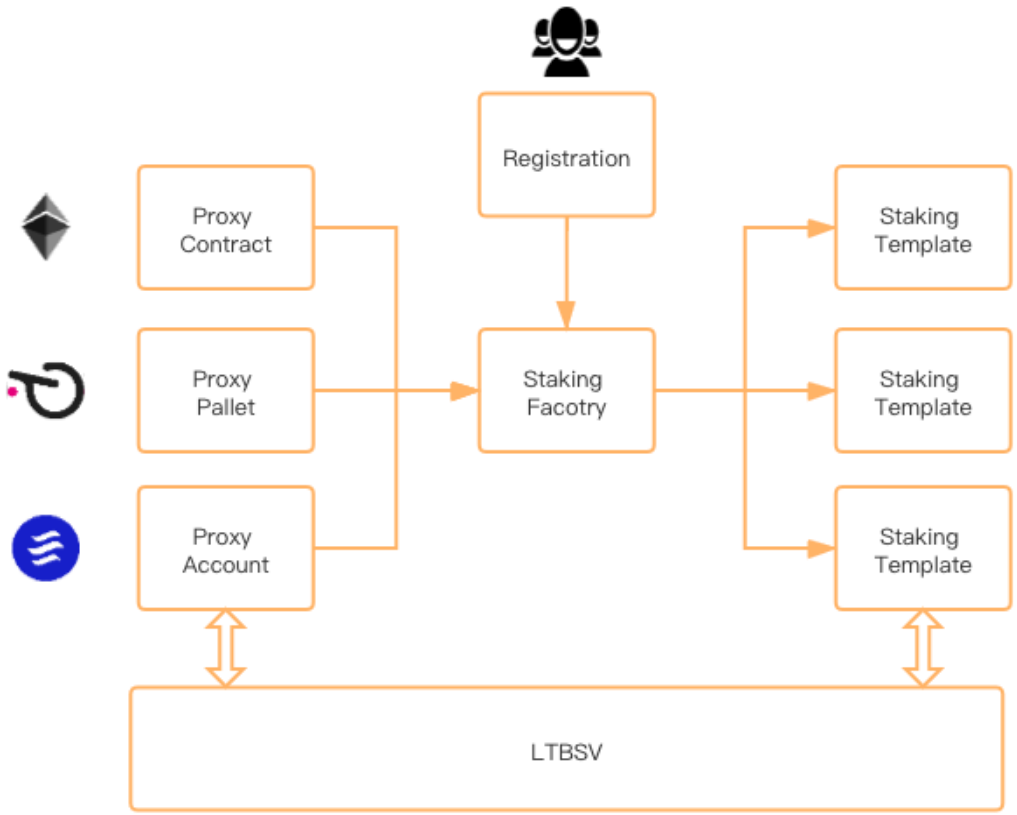

Figure3, Contract Topology

As shown in the figure above, OCSP is mainly composed of four main modules: Registration, StakingTemplate, StakingFactory, and BridgeProxy.

- **Registration** contract is used by individuals or organizations to submit registration information to Nutbox.

- **StakingTemplate** contract contains the tToken-based staking economic model implemented by Nutbox. With this contract template, it is possible to create a staking tToken or provide liquidity for tToken-cToken trading pairs to mine cToken staking business.
- **StakingFactory** contract is used to generate the StakingTemplate instances.
- **BridgeProxy** contract varies according to the implementation of the blockchain network. In a blockchain network that supports smart contracts, it will be implemented through smart contracts. For example, in the Ethereum network it is a Solidity contract, while in the Steem network it is just an account in the Steem network. It is primarily used by cross-chain Bridges to capture Nutbox-related asset transfer behavior in the original blockchain network.

3.2 Multi-Chain tToken

On Nutbox, users stake the assets of the original PoS blockchain network to obtain the corresponding tTokens in the Nutbox network, which are exchanged 1:1 with the underlying assets.

For example, a user in the Nutbox network can exchange DOT holdings for tDOT, which can participate in the Nutbox staking mining of all Polkadot ecological projects, i.e., stake tDOT to obtain the corresponding community's cTokens. Nutbox will combine cross-chain bridge technology and Polkadot XCM to support asset exchange across multiple PoS blockchain networks.

3.2.1 tToken Exchange Gateway（TEG）

TEG contract (Nutbox is deployed as a contract) and TEG pallet [5] (Nutbox is used as Polkadot Parachain) are responsible for implementing the cross-chain asset exchange gateway of the Nutbox network.

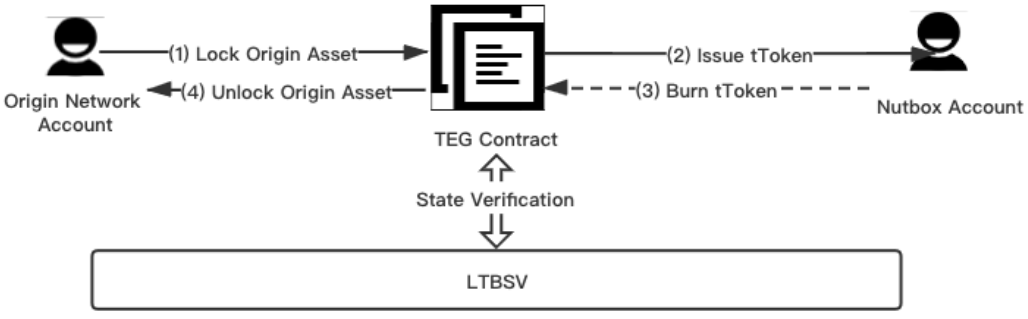

Figure4, tToken Exchange Gateway

As shown in the figure above, when a user exchanges a native asset for tToken, the native asset will be locked in the TEG Contract, and LTBSV verifies that if the user's

asset is locked [shown in the Deposit proof section for details], when the verification is passed, it will generate a corresponding amount of tToken to the user In the Nutbox account; the TEG contract verifies the user's tToken burning status when the user redeems it [shown in the tToken Burning Contract section for details], and when the verification is passed, the user's native assets locked in the TEG Contract will be unlocked.

3.2.2 Bridge Proxy Contract

The bridge proxy contract is a smart contract deployed by Nutbox to the original blockchain network and is responsible for storing and transferring assets in the original blockchain network. Asset transfer is completely decentralized. Anyone who needs to transfer assets from Proxy Contract needs to provide Withdraw proof. After verification, you can transfer assets from the contract to your own account.

3.2.3 tToken Issue Contract

The issuance of tToken requires the user to provide a deposit proof [shown in 4.2.3]. The deposit proof is issued by the LTBSV to the user when the user deposits the original blockchain assets into the Nutbox proxy contract. After the user submits the Deposit proof, TEG's tToken Issue Contract verifies the signature information of the Deposit proof firstly, and the verified Deposit proof is parsed as Deposit MetaData, which contains the amount of the original blockchain assets staked by the user.

TEG's tToken Issue Contract then issues the corresponding amount of tToken, and the exchange gateway creates a tToken Issurance Pair for this , which contains the amount of tToken issued, the user's public key, and the unlock time. The tTokens in the lock-up period cannot be exchanged for tokens in the original blockchain network.

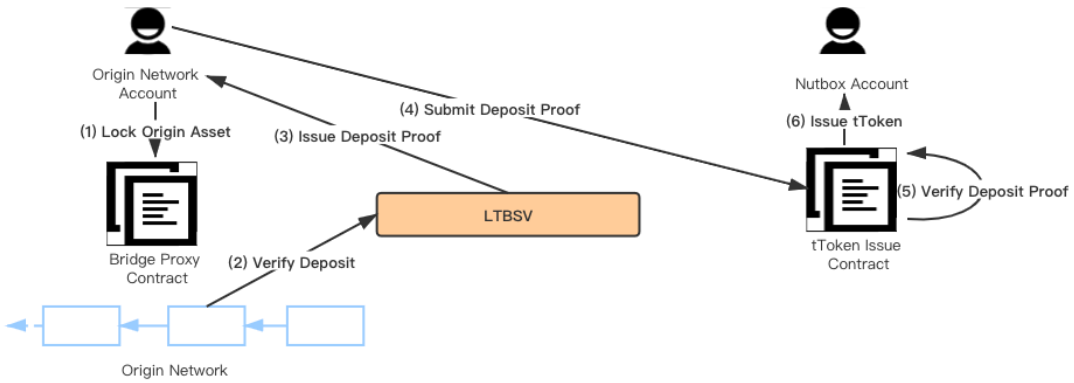

Figure5, tToken Issurance

3.2.4 tToken Burning Contract

When the unlocking time is reached, users can choose to redeem their unlocked tTokens at any time (users who use tTokens to participate in community token mining on Nutbox will also be locked). After the user chooses to burn a certain amount of tToken, TEG's tToken Burning Contract will burn the same amount of tToken, and Nutbox will issue a Withdraw proof to the user. The proxy contract will verify the Withdraw proof signature information. After the verification is passed, the Withdraw MetaData will be parsed out, which contains the amount of tTokens that the user needs to unlock and the user's public key in the original blockchain. Then, the proxy contract transfers the specified amount of original blockchain assets to the user's original blockchain network account.

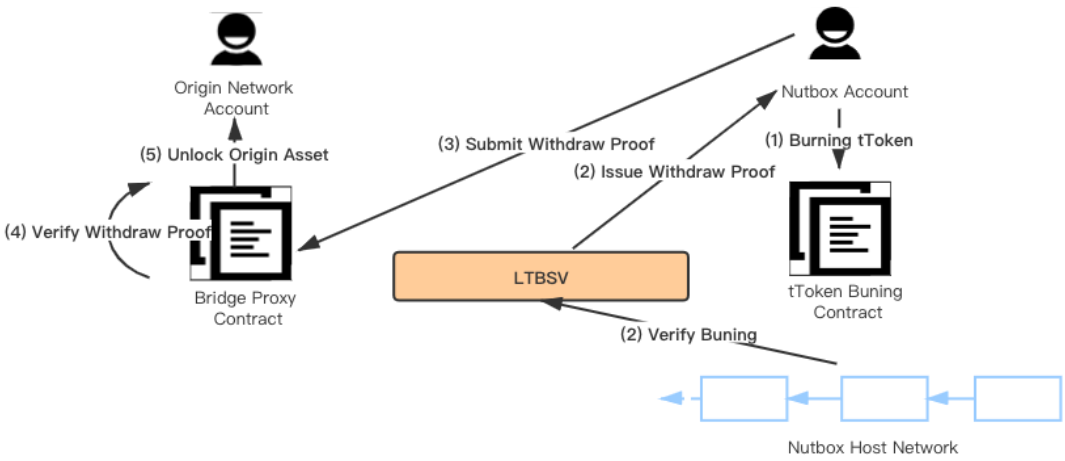

Figure6, tToken Burning

3.3 Community Token(cToken) Mining & Distribution

cToken is the community token specified when any organization or individual creates their own staking economy. A basic distribution strategy is to stake tToken to mine cToken. Nutbox supports a variety of cToken distribution strategies, which users can specify by themselves.

In the cToken mining algorithm, we have introduced two main variable factors.

shareAccumulate

Shared cumulative factors, global variable factors, will be used for all user reward calculations, represented by pool.shareAcc. The pool corresponds to one type of staking asset. For example, staking tDOT and staking the LP of the tDOT-USDT trading pair correspond to two pools respectively, and the profit sharing ratio of each pool can be configured.

debtRewards

User private variable factor, which represents the amount of reward that needs to be excluded when calculating the user's reward, represented by user.debtRewards.

$$shareAcc_{pool} = shareAcc_{pool} + \frac{f_{rewards}\{from, to\}}{totalStakingAsset_{pool}}$$

In the above formula, the initial value of $shareAcc_{pool}$ is 0 and $f_{rewards}\{from, to\}$ represents the newly generated cToken at a certain block height. This value is specified when creating a staking economy. $totalStakingAsset_{pool}$ represents the total amount of staking assets in the current pool.

$$user.debtRewards = user.amount * shareAcc_{pool}$$

In the above formula, $user.amount$ represents the amount of staking assets of the user.

$$user.rewards = user.amount * shareAcc_{pool} - user.debtRewards$$

In the above formula, $user.rewards$represents the cToken rewards that the user can get under the current block.

After setting up the cToken distribution strategy, the users participating in the staking will get the corresponding amount of cToken rewards according to the block.

4 Crosschain Bridge

Nutbox needs to handle the bi-directional transfer of assets in multiple blockchain networks. Since different blockchain networks have different block consensus algorithms and block verification logic, Nutbox needs to verify the legitimacy of transactions in each blockchain network to achieve cross-chain asset staking. Nutbox's cross-chain bridge service is specifically used for data verification of different blockchain networks and as an infrastructure for cross-chain asset transfer.

4.1 What Is Crosschain Bridge

A cross-chain bridge is a technical means to communicate between two different blockchain networks. Through the cross-chain bridge, users can transfer assets in different blockchain networks.

For example, if a user exchanges 1BTC for ERC20 assets xBTC on Ethereum (if 1BTC = 1xBTC), the user's Bitcoin account will be reduced by 1BTC, and at the same time the

ERC20 assets under the user's account in the Ethereum network will increase by 1xBTC.Users can exchange ERC20 assets into BTC at some point in the future.

The cross-chain bridge technology has multiple implementation methods according to different application scenarios, and the technical implementation details and degree of decentralization of each method are different. Nutbox adopts cross-chain bridge technology based on LTBSV (as below) to realize the transfer of cross-chain assets and the verification of block data.

4.2 Light Trustless Blokchain State Verification (LTBSV)

LTBSV is a trustless block data verification protocol that can verify block data without trusting any third-party block data source. That is, for any new block $Block_n$, the latest block $VBlock_{n-1}$ that has been verified, and the LTBSV validator f_{LTBSV} can verify whether the latest block is legal:

$$VBlock_n = f_{LTBSV}\{VBlock_{n-1}, Block_n\}$$

LTBSV is different from the full nodes in the blockchain network. It does not need to store the entire block data, nor does it need to verify every transaction in the block. The only difference from the light node is that it does not need to have a wallet function, which means, it does not need to store any user's transfer information, and it does not need to maintain an account balance based on the UTXO ledger model like Bitcoin. LTBSV includes the following components.

4.2.1 Block Fetcher

Get the latest block data from any third party without trusting the third-party data source. Usually only need to provide Block Fetcher corresponding to the full node link of the blockchain network, Block Fetcher will poll the latest block according to the set frequency. If for some reason Block Fetcher is suspended and restarted, it will still obtain all block data up to the latest block of the network based on the block height from the latest block verified last time and verify the legitimacy of each block one by one. If the new block data cannot match the original block data, the verification will fail.

4.2.2 Block Validator

Block Validator is responsible for the verification of block data. Since LTBSV does not care about every transaction in the block, it only cares about the transactions related to the proxy contract to some extent, as a result, LTBSV only verifies the block header in combination with the Merkle Root Hash of the transaction. In light nodes, all block headers need to be stored, that is, when the node is started, all previous block headers need to be synchronized. Even just synchronizing the headers would take several hours for most existing blockchain networks. LTBSV uses the Check Point mechanism,

which was first introduced in the Bitcoin system. The Bitcoin wallet electrum uses the Check Point mechanism[11], that is, developers hard-code the verified block header in advance. In the Bitcoin system, due to the difficulty, the value is adjusted every 2016 blocks, therefore, only the block hash with the block height of $2016 * n\{n = 1,2,3,,,n\}$ and the corresponding difficulty value[12] are needed to encode the block height. After adopting the Check Point mechanism, the verification of the latest block can be completed within a few minutes after LTBSV is started.

4.2.3 Deposit Proof

After LTBSV completes block data verification, it issues a deposit proof for valid transaction data. LTBSV uses ECDSA's elliptic curve encryption algorithm[14] to sign data. Deposit proof is used to prove the user's asset staking behavior in a specific blockchain network, that is, when the user stake assets to Nutbox's Proxy Contract in the blockchain network, they need to provide their own data signature$sig_{(amount,nonce)}$ for the staking amount and the account nonce value, here

$$sig_{(amount,nonce)} = f_{signer}\{sha3\{(amount, nonce)\}\}$$

Proxy Contract will verify the user's signature data and parse out $amount$. If $amount$ is equal to the received user transfer amount, the transaction is successful, otherwise the transaction fails.
Then LTBSV parsed the specific content of the staking transaction from the block, including the user's signature and the amount of staking assets. Then attach their own signature data sig_{LTBSV}

$$sig_{LTBSV} = f_{signer}\{sha3\{currentBlock + confirmedBlock\}\}$$

and finally constitute Deposit proof,

$$proof_{LTBSV} = (sig_{(amount,nonce)}, sig_{LTBSV})$$

The user then submits a deposit proof to the TEG module for verification. The TEG module will analyze the block height of the user's staking transaction and compare it with the current height. If the current height is not below the set confirmation block height, the transaction will fail. After the verification is passed, the equal amount of tToken will be distributed to the user's Nutbox account.

5 Nutbox As Parachain

Nutbox Blockchain is built on the framework of substrate FRAME [6] and uses substrate to build a blockchain network. It can reuse many mature, stable and reliable functional modules, and has an upgradeable Runtime. Upgradable Runtime means that the blockchain network no longer needs a hard fork to achieve the purpose of upgrading the system, ensuring the stability of the blockchain network. After Nutbox obtains the parachain card slot through the public slot auction, it can directly access the Polkadot parachain network.

5.1 Nutbox Collator

In the Polkadot network, Collator is different from the Polkadot relaychain full node or Validator. The Collator is used to run the parachain network. It does not contain its own consensus system (PoA or NPoS). It has a built-in Polkadot Relaychain full node, and it packets the PoV data of Parachain's transactions to send to Relaychain.The block corresponding to the PoV data verified by the relaychain is considered a valid Parachain block and is finally stored in Parachain's own block database.

After the Nutbox Collator is launched, anyone can join the Nutbox network to run the Nutbox Collator node. Although the block data does not need to be verified, the node will also be rewarded for running it.

5.2 XCM Based Transfer

Once Nutbox is connected to the Polkadot parachain network, it can realize cross-chain asset transfer based on the XCM protocol. For other blockchain networks that are also connected to the parachain network, cross-chain asset transfers can be realized between each other through the following methods:

- Use XCM instructions to construct an XCM message for cross-chain transfer
- Configure XCM Executor to execute XCM instructions and forward xcm messages to the target parachain through UMP or HRMP

At present, XCM V0 has been implemented and running on the Rococo test network. In the V0/V1 version, the asset is described as MultiAsset, and an address in the parachain network is described as MultiLocation. Among them, MultiAsset can be either a concrete asset or an unconcrete asset; it can be a fungible asset or an unfungible asset. MultiLocation can be used to represent a parachain network, it can also be used to represent an account in the network or even a smart contract deployed in the network.

6 Asset Price Oracle

6.1 Price Oracle Algorithm

From the above, we can see that there are various types of Assets. In this chapter, we will only discuss the price of Fungible Asset.

The price acquisition oracle has always been a branch of technology that is worth exploring in the industry. It usually includes two stages: price calculation and price feed. There are many implementations that can meet basic business needs. Nutbox refers to Uniswap's Price Oracle implementation for asset price acquisition [7]. For general assets such as BTC and ETH, we can derive the price of the corresponding asset through Uniswap trading pairs. Since trading pairs in Uniswap have a constant product K, each trading pair can be finally deduced as a trading pair relative to USDT.

Therefore, the asset price can be expressed as:

$$price_{token} = pairInstance.token0Price$$

The asset balance can be expressed as:

$$f_{balance} = token.balanceOf(address)$$

Create a trading pair,

$$pairInstance = f_{pair}\{f_{balance}(f_{pairAddress(token,usdt)}) * USDT, f_{balance}(f_{pairAddress(token,usdt)})\}$$

In the above formula, f_{pair} and $f_{pairAddress}$ are from Uniswap V2 SDK[8]. After the price is calculated, the price is written into the contract on the chain through the delayed price feed n_{block} strategy. Since Nutbox needs to support asset price acquisition of multiple blockchain networks, the block generation speed of different blockchain networks is not the same, so n_{block} needs to be set according to the block generation speed $t_{newBlock}$, then there is

$$n_{block} = \frac{t_{delay}}{t_{newBlock}}$$

Nutbox not only supports the staking of general assets, but also the staking of LP tokens. LP token staking relies on the price oracle of LP token, and the price acquisition of LP token in the realization of the price oracle has certain peculiarities. Usually the price calculation of LP token can be expressed as:

$$price_{lptoken} = \frac{r_0 * price_0 + r_1 * price_1}{totalSupply}$$

In the formula, $r0$ and $r1$ are the quantities of the two assets in Uniswap's corresponding trading pair, and price0 and price1 correspond to the prices of the tokens represented by $r0$ and $r1$.The above formula is simply to calculate the sum of the total value of the two tokens in the trading pair, and then divide it by the total number of LP Tokens to get the price of LP tokens. In practice, this price calculation method is easy to be attacked by hackers, because hackers can easily manipulate $r0$ and $r1$ through trading pool transactions. Therefore, referring to Alpha Finance's LP Token price acquisition algorithm[9], a better LP Token price acquisition algorithm can be expressed as:

$$price_{lptoken} = 2\frac{\sqrt{r0 * r1} * \sqrt{price_0 * price_1}}{totalSupply}$$

It can be seen from the above formula that the consequences of hackers' attacks on $r0$ and $r1$ are reduced from $r0 + r1$ to $\sqrt{r0} + \sqrt{r1}$. This algorithm realizes the constant product K of the trading pair curve based on Uniswap, so it is only applicable to the price calculation of the trading pair LP Token based on this trading model.

6.2 Offchain Worker Based Price Oracle Flow

The current substrate provides the Offchain Worker [10] mechanism, which allows nodes to obtain external data through http and perform complex calculations such as encryption and decryption. Offchain Worker has Offchain Storage, which can be accessed directly by runtime, and other pallets can be accessed in Offchain Worker's pallet. At the same time, Offchain Worker can also submit data that requires consensus verification through Signed Transaction and Unsigned Transaction.

The price oracle based on the Offchain Worker module uses the http protocol to obtain price information from a third-party price provider, calculates the asset price according to the price calculation algorithm, and then submits the price to the Nutbox network.

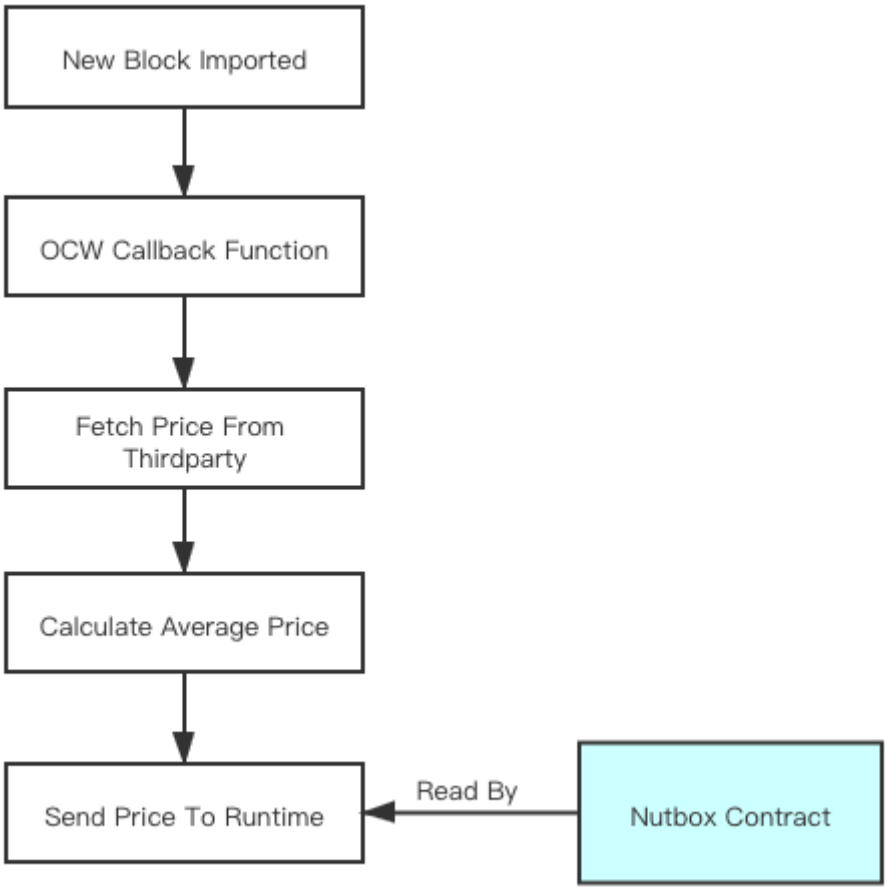

Figure7, Offchain Worker Based Price Oracle Flow

7 Nutbox Governance Contracts

Nutbox DAO will operate through on-chain governance. After the stable operation of the project, all decisions related to the project will be voted on-chain. The implementation of Nutbox Governance consists of a series of related contract modules, including:

7.1 Democracy

Democracy Contract can implement Nutbox's on-chain proposal voting system and is the main module of on-chain governance. Democracy contains two Proposal Queues, namely Internal Proposal Queue and External Proposal Queue. Proposals submitted by Nutbox will enter the Internal Proposal Queue, and user-initiated Proposal will enter the External Proposal Queue. Different voting strategies can be set for the two types of proposals.

7.2 Collective

Collective Contract maintains a set of accounts and only accounts in the Collective set are eligible to vote.

7.3 Elections

Elections Contract can implement a decentralized voting system based on dynamic weights. The user's voting weight is a combination of the amount of staking funds and other conditions.

7.4 Membership

Membership Contract manages the accounts in the Collective Contract account set and sets the restrictions in the Collective account set.

7.5 Treasury

Treasury Contract manages the funds of Nutbox DAO and each fund needs to be signed by the relevant Nutbox Multisig account.

7.6 Multisig

Multisig [13] Contract is used to implement Nutbox's multi-signature accounts.

7.7 Sudo

Nutbox Sudo is used to manage and set up Nutbox contracts, such as configuring LTBSV, TEG signature verifier, etc.

8 References

[1] PoS Consens Algorithm https://en.wikipedia.org/wiki/Proof_of_stake
[2] PoW Consens Algorithm https://en.wikipedia.org/wiki/Proof_of_work
[3] NPoS Consens Algorithm https://w3f-research.readthedocs.io/en/latest/Polkadot/NPoS/
[4] XCM https://github.com/paritytech/xcm-format
[5] Substrate Pallet https://substrate.dev/docs/en/knowledgebase/runtime/pallets
[6] Substrate FRAME https://substrate.dev/docs/en/knowledgebase/runtime/frame
[7] Uniswap Oracles https://uniswap.org/docs/v2/core-concepts/oracles/
[8] Uniswap SDK https://uniswap.org/docs/v2/API/overview/
[9] Alpha Finance LP Price https://blog.alphafinance.io/fair-lp-token-pricing/
[10] Substrate Offchain Worker https://substrate.dev/docs/en/knowledgebase/runtime/off-chain-workers
[11] Eelectrum Check Point https://electrumx.readthedocs.io/en/latest/protocol-basics.html#block-headers
[12] Bitcoin Check Point Format https://github.com/spesmilo/electrum/blob/master/electrum/checkpoints.json
[13] Multi Signature https://en.bitcoin.it/wiki/Multisignature
[14] EllipticCurve Cryptography https://en.wikipedia.org/wiki/Elliptic-curve_cryptography
[15] Ethereum Sharding https://eth.wiki/sharding/ethresearch-sharding-compendium