Overview

Background

Since the birth of Bitcoin, tens of thousands of blockchain-based digital currencies have come out. However, only a few projects have their fundamental chain and can truly promote the development of blockchain technology. We are very interested in the progress made in the privacy and anonymity-related technologies of blockchain. With an address, Bitcoin can label an identity. With this identity, users can deliver value on Internet, thus to carry out various value exchange activities, without exposing too much privacy information. Today, the Internet-based value exchange is quite common in our life, such as cinema ticket booking, online hotel reservation, online shopping and so on. People have left too much privacy information on Internet. The transparency nature of Bitcoin leads to failure in solving this problem well. However, it is believed that the current anonymous currencies still have much room for improvement in the range of application and in the convenience of use. In addition, due to an inadequate combination with other technologies, anonymous currencies have failed to be applied in more scenarios.

Solution

Therefore, we are looking forward to designing a new anonymous currency.

1. Anonymous mechanism is provided based on the MimbleWimble technology.
2. With X17r algorithm, it allows the participation of more common users.
3. It supports multi-assets.
4. It integrates with the relatively mature technologies in this industry, such as lightning network, cross-chain atomic exchange and smart contract, which expands the application of this new anonymous currency.
5. Anti-quantum algorithm is used to ensure a high security of the blockchain-based cryptographic algorithm.

X17r Mining Algorithm

Mining algorithm is a part of consensus. Here we choose the X17r mining algorithm which is anti-ASIC. With this algorithm, users can participate in mining with common PC. In early stages, there were no special rigs, and CPU was the only unit that could mine fairly. At this time, sufficient user participation can well ensure the decentralization of fundamental chain. Besides, different algorithms prevent plenty of BTC or hash power of other primary networks from attacking the UFO789 network.

X17r algorithm is another transformation algorithm after X11, X13, X15 and X17.

X11 algorithm refers to the use of 11 kinds of fixed Hash algorithms in series, thus to increase the R&D cost of ASIC rig. X13 algorithm is to connect 13 kinds of fixed Hash algorithms. However, to purely increase the quantity and variety of algorithm can only delay the appearance of ASIC rigs.

X17r algorithm is to continuously disrupt the series connection order of Hash algorithm, thus to have the computing difficulty increase exponentially.

X17r chooses 17 verified algorithms of X17. However, these 17 algorithms are not in fixed arrangement, but change dynamically based on the hash value of the previous block.

To do Hash algorithm with a fixed order, the chip utilization ratio of ASIC rig can reach 100%. However, to do unordered Hash algorithm, the chip utilization ratio of ASIC rig is only 64.38%.

Take X11 algorithm as an example. It is only necessary for rig developers to develop ASIC rigs corresponding to 11 kinds of Hash algorithms, and then to assemble them in a fixed order, just like the operation in workshop. Every Hash algorithm is computed by corresponding ASIC rig, which goes in endless cycles. No chip will be left idle, but will operate at full load.

Two uncertainties are added to X17r.
1. The uncertainty in the algorithm order.
2. The uncertainty in the frequency of algorithm occurrence.

Every block will do 17 operations, but it is uncertain what Hash algorithm will appear in these 17operations.

Suppose only one and the same algorithm appears in the 17 operations, such probability is $17/\ (17^{17}) = 2.055025702114\ e\text{-}20$.

Suppose two different algorithms appear in the 17 operations, such probability is [17*16/2 (two from 17 algorithms) $*2^{17}$] / $(17^{17}) = 2.15485063062e\text{-}14$

......

Suppose 16 algorithms appear in the 16 operations, such probability is 17！ / (17^{17})=4.299687097977e-7

Through weighted averaging, it is found out that every block will involve 10.3 kinds of functions. Suppose the chips doing every kind of hash computation have the same area, the utilization ratio of chip is 10.3/17=64.38%. It means that at least 35.62% chips will be wasted in the X17r computation by ASIC rigs.

X17r also has another feature. As the time spent on every kind of hash computation varies, the time of block production also varies. From the long run, the block production time will be averaged due to the match of hash functions.

Anonymity

MimbleWimble is an anonymity technique that has been proven quite reliable. With this technology, we can protect the privacy of users on the blockchain. Compared with other Zero-Knowledge Proof plans, it is more lightweight, and has maintained its security.

Mimble-Wimble is dependent on the elliptic curve cryptography (ECC). In ECC, a large number k is usually chosen as the private key. If H is a point on the elliptic curve, k*H is taken as the public key. Given the nature of elliptic curve, it becomes very hard to deduce the private key k from the public key, because the division of curve point is very difficult. Based on this property, we can conceal the actual transaction amounts in transactions, as shown below.

Suppose the transaction amount is v, it is necessary to verify v1+v2=v3, when the node checks that the output of transaction equals to input. As shown in the following formula, it is to multiply both sides of the equation by H, a point on the elliptical point.

v1*H+v2*H=v3*H，

Though it has become very hard to deduce the actual transaction amount in this way, it is still possible for attackers to deduce the value of v1, for the reason that the number of attemptable sets is still limited. Thus, point G, the second point on elliptic curve, and the private key r are introduced. The value of any input and output in the transaction is expressed by r*G+v*H. Given the nature of elliptic curve, neither r and v can be deduced. The equation to be verified is shown as follows.

r1G+v1H+r2G+v2H=r3G+v3H，

In addition, it is required that r1+r2=r3, so the actual transaction amount will be properly concealed. In actual transactions, the transaction amount is available only to both parties of transaction. The information seen by blockchain nodes is encrypted digits, and the private key is available only to the user himself. To verify that the input of transaction equals to output, and to protect sender’s private key from being cracked by receiver, the sender needs to select an excess value, and add it to his own private key. The receiver can only see the sum of both, and will never know the value of private key which is available only to the user himself. It is only necessary to verify the output of transaction equals to the input, and only necessary for traders to know the excess value (building an ECDSA signature with it to prove that you know it). Therefore, excess value serves as the private key of a transaction. Verifying UTXO can prevent double spending.

In the process of merger, the verification process is similar. In a single transaction, what is to be verified is that output equals to input, so in merged (including mixed transaction and deletion of intermediate state) transaction, what is to be verified is still that total / final output equals to input in nature. For the transaction of intermediate state, it is only necessary to verify its signature. In case of double spending, nodes can easily check out that the total transaction output does not equal to input, just as in Bitcoin. In Mimble-wimble, nodes can compare all the funds from mining with the total sum held, to check whether the total currency supply is correct. Range proof can ensure no currency overissuing occurs in transactions.

Multi-asset

Multi-asset allows more users to use anonymous blockchain. When users want to issue digital assets, they don’t have to establish a new public chain and to pay maintenance cost and contribute computing power to maintain it. The exchange and circulation of assets of different value will also become very convenient.

Atomic Exchange

The Hash-locked atomic exchange has been applied in some blockchains. Though this is a small improvement, it provides the minimum safety guarantee for the exchange and circulation of cross-chain assets. We will first have atomic exchange applicable to BTC and LTC, and then to other main-chain assets as needed in a longer time.

If Alice wants to exchange her ADA for Bob's BTC, she will do as follows.

1 Alice creates a random password s, and works out the hash value h of the password, that is, h = hash(s). Alice sends the hash value h to Bob.

2 Alice and Bob jointly lock their assets successively (Alice first, and then Bob) through the smart contract. The smart contract implements the following logic.

Condition 1: If anyone can supply a random value s' to the smart contract in H hours, and once the smart contract verifies that hash(s') == h (when s' equals to the original password), Bob's BTC will automatically be transferred to Alice, and will be returned to Bob in case of timeout.

Condition 2: If anyone can send the original password s to the smart contract in 2H hours, Alice's ADA will be automatically transferred to Bob, and will be returned to Alice otherwise.

3 Condition 1 is set for Alice. To get Bob's BTC, Alice will definitely provide the smart contract the random password s before time runs out, and the smart contract will surely have s pass successfully, thus to transfer Bob's assets to Alice. When the deal is done, the original password s provided by Alice will be broadcast, and recorded in the blockchain. At this time, Bob can send the publicized password s to the smart contract, and get Alice's ADA which is locked in the smart contract as specified in Condition 2. Theoretically, Bob has 1H to 2H hours to complete the operation (depending on how long Alice can complete her operation). It satisfies both.

Of course, Alice has the initiative and more advantages in the Hash-locked transaction. In more complex transactions, such as transactions with fluctuate exchange rate, Alice can choose to wait until a more satisfying exchange rate comes out to trigger the exchange, and Bob can only accept it passively. Of course, Alice may miss the best rate in the end. In general, Alice has more choices, thus getting a bargain in the transaction.

Generally, the cross-chain hash-locking will be matched with the "state channel", which achieves a faster transaction, thus to avoid the above-mentioned problem.

Lightning Network

The lightning network has a high-speed transaction channel, and has the computed result verified on the blockchain. It can greatly improve the transaction velocity, and save the storage cost and computing cost of the fundamental chain. At present, lightning network has been applied in BTC. When UFO789 network develops to a certain level, such improvement will inevitably be made. We will carry out this improvement as planned, so as to avoid network congestion in the future and to reduce cost.

Smart Contract

Smart contract is the best way to expand the function of blockchain. In current situation, it is very possible for blockchain to combine with the traditional industrial applications in many scenarios, but such combination is far from mature. Smart contract is the solution that allows us to make various experiments at the fastest speed and at the lowest cost. Based on the mature virtual machine technology, we will transplant the smart contract technology that supports multiple development languages to UFO789 project, and will make explorations for this industry.

Smart Contract and Virtual Machine

It uses a Turing-complete and smart-contract-oriented bytecode specification as the implementation specification for smart contract virtual machine. It provides the compilers of static-type high-level programming languages such as C#, Java and TypeScript, to form high-level languages into smart contract bytecodes.

- Smart contract virtual machine:
  The smart contract virtual machine is realized as a Turing-complete bytecode virtual machine which can operate definitively, control the logic execution and monitor the change of state.

- Smart contract language:
  The subsets of the popular programming languages, such as C#, Java and TypeScript, are taken as the high-level programming languages of smart contract, and are compiled to the bytecodes that are in compliance with smart contract bytecode specifications.

- Built-in library of smart contract:
  Smart contract provides the basic libraries for common numeric operation, string operation and so on, as well as the built-in function libraries for on-chain query, transactions and so on. In the smart contract, the built-in libraries can be called.

- Mutual calling of smart contract:
  The smart contract, after being deployed on the chain, can call other smart contracts / built-in native contracts, or be called by other smart contracts, besides being directly called by users or having access to assets.

  Some functional logics can be implemented in the form of smart contract, and be deployed on the chain and used by other smart contracts as the third-party library to expand the function of blockchain.

- Functional scope and limitation of smart contract:

Smart contract can compile business logic with Turing-complete programming languages, query the on-chain data, definitively have access to the state storage of the contract, call other smart contracts / native contracts, and output the return information to the caller.

Limitation: It cannot read data that is not on the chain, and cannot definitively produce logics where nodes are inconsistent with each other. The number of instructions executed and the memory space used are controlled by blockchain. The blockchain can terminate the performance of smart contract immediately and any time, such as when expenses exceed the budget.

- State storage of smart contract:

  Each smart contract has an independent state storage space which is referred to as storage. The format of storage is non-structural. Only the changed part of smart contract storage is stored on the chain, instead of the latest and the entire storage. For instance, in a contract calling when the smart contract storage is modified from *{"name": "chain" }* to *{ "name": "chain", "count": 123 }*, only the part modified, *{ "count": 123 }*, is recorded on the chain. Even when the smart contract calls the service charge, only the modified part, instead of the entire storage, will be charged. Therefore, even though the storage of smart contract is large, the increase of data on the chain and the service charge will not be great, if the amount of change incurred by contract calling is not large.

- State query of smart contract:

  Smart contract can both directly query some values of its storage, and use SQL-like programming language to get some data from the nested data structure. When the storage of smart contract is large, data load can be reduced and query speed can be improved in this way. This method avoids the full-table scanning, and raises the cap of performance of smart contract's data access part.

  For instance, the storage of a smart contract has a similar structure.

```
{
 "name": "blockchain",
 "userBalances": [
  { "userAddress": "a", "amount": 10000, "freeze": false },
  { "userAddress": "b", "amount": 20000, "freeze": true },
  { "userAddress": "c", "amount": 30000, "freeze": false },
  ……. More data, such as hundreds of thousands of records
 ]
}
```

The SQL-like syntax, like *var freezedUsers = storage.query("select userBalance.userAddress from userBalances as userBalance where freeze=true"*), can be used to query all the addresses of users whose accounts are frozen in this smart contract. In this way, the amount of data read and written will be significantly reduced, and the full-table scanning will be avoided. It is applicable to business scenarios which have mass data stored in the smart contract but have a little data read each time, such as simple push exchange, smart contract asset, contractual insurance and so on.

- Life cycle of smart contract:
  - The smart contract bytecode files are produced with high-level programming languages or by manually constructing bytecodes.

- Smart contract bytecodes are deployed to blockchain. They can either be created as smart contract, or be created as smart contract template for future use.
- Calling smart contract API, or making bank transfer to smart contract address.
- Every time when blockchain calls the smart contract, it first initializes the independent and lightweight smart contract sandbox execution environment, and then loads and performs the smart contract.
- After the smart contract is performed, the result of performance and the change of smart contract storage will be stored, according to whether the exit status is normal.

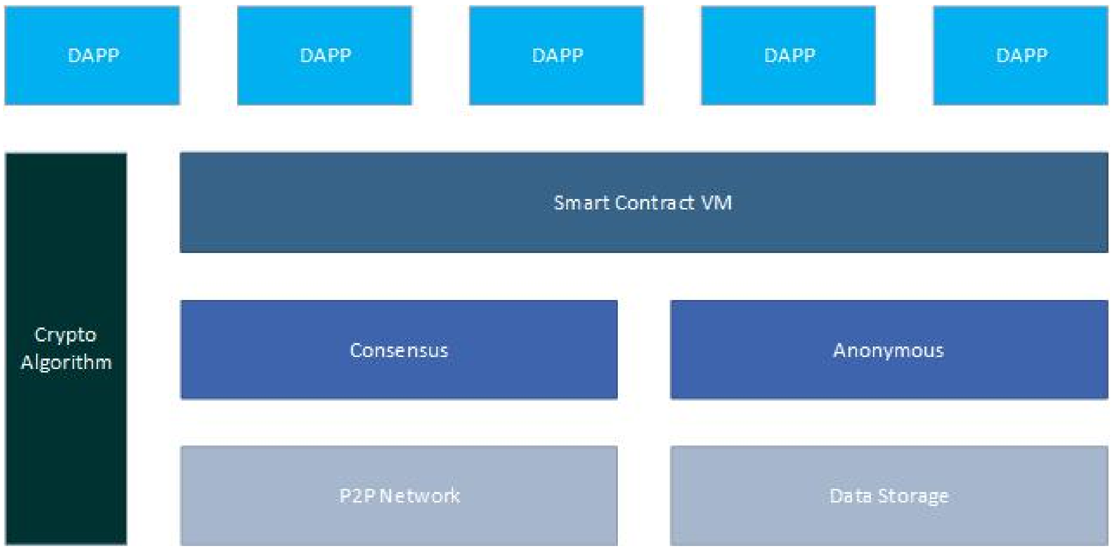
Anti-quantum Computation

As breakthroughs are made in the quantum algorithm technology, the algorithms used in the field of blockchain have become insufficient to protect users' information, assets and transactions. We will, according to the development of quantum computation, add anti-quantum algorithm in due time, in order to protect the security of user information, assets and transactions. The international "academic conference on anti-quantum computation cryptography" has yielded a number of research findings. This is one of the conferences we will give constant attention to.

System Design

We will design and implement the system which is similar to the mainstream blockchain systems in a hierarchical way.

1. The underlying stable and efficient network and the data layer lay a solid foundation for the basic protocols of blockchain.
2. The consensus layer and the anonymity protocol use the network to disseminate data, and use the data layer to verify and store data.
3. Smart contract is built on the unalterable blockchain layer. The virtual machine is used to expand the function of blockchain.
4. The security and reliability guarantee provided by cryptography is everywhere on all the layers.
5. Based on the smart contract virtual machine, every smart contract or every group of contracts can provide reliable data foundation or flexible autonomous logic for DAPP.


DAPP
DAPP
DAPP
DAPP
DAPP
Crypto Algorithm
Smart Contract VM
Consensus
Anonymous
P2P Network
Data Storage