THE TAO

Tao the natural order

MAR 23, 2021

V 1.1

https://thetao.cash

Table of Contents

Introduction

As time flies, it has been more than 10 years since the birth of Bitcoin, and the entire world of cryptocurrencies has been climaxing and troughing with the fluctuation of the price of Bitcoin. During this period, there exist many "rising stars", like Ethereum taking the beating or EOS being totally balls-up. Time flying away, although there have been no perfect coins yet, occasionally some fresh coins turn eye-poppers, such as BHD and Grin. This paper intends to elaborate on the situation of the coin circle in the past 10 years, in order to explain the current landscape and deficiencies of the cryptocurrency world. Then we try to propose a new minable coin based on zk-SNARKs technology, The TAO (DAO), which employs Proof of Capacity (aka, PoC) as its consensus model.

TAO (/daʊ/, /taʊ/) is a Chinese word signifying the "way", "path", "route", "road" or sometimes more loosely "doctrine", "principle" or "holistic beliefs". In the context of East Asian philosophy and East Asian religions, TAO is the natural order of the universe whose character one's human intuition must discern in order to realize the potential for individual wisdom. This intuitive knowing of "life" cannot be grasped as a concept; it is known through actual living experience of one's everyday being.

-- Wikipedia（https://en.wikipedia.org/wiki/Tao）

Background

There are three mainstream consensus models in the blockchain world: Proof of Work (PoW), Proof of Stake (PoS) and Proof of Capacity (PoC). Others are all evolutions on these three categories while DAG is not a blockchain at all and beyond the scope of this article. Some others are the home brew ones, which cannot to be mathematically verifiable, nothing but playing to the gallery.

The privacy coin proves its existence value from the perspective of Utility Token.

Three Consensus Models

PoW is the earliest and most mature consensus algorithm, representatives of which are Bitcoin and its fork, Ethereum and its fork, as well as Litecoin, Zcash, Monero, etc., and its main feature is that miners follow diverse types of cryptography hash algorithms specified by the blockchain networks to find a nonce value in some brute-force way within the certain time, in order to have the accounting right and get the reward by generating blocks. PoW is often criticized due to its significant waste of the electricity. The growing trend of hash algorithms built in ASICs has made PoW mining become heavy-asset projects, which daunts small and medium-sized retail investors while only capital predators can afford mining.

Proof of Stake, PoS, exists for quite a long time, but the price of PoS tokens has been flat. The reasons are as follows. First, all coins are generated and distributed in the "Genesis Block", and no longer needed to be mined. Secondly, excessive concentration of stakes is prone to bribery elections. The community differentiation of Steemit is the latest example. This is neither the first nor the last time. It is just a rather famous one. Thirdly, the total amount of PoS coins is usually huge. Considering the incentives of block producers, the inflation model is generally used. Fourthly, PoS has not find its practical application purposes so far. Even if Ethereum 2.0 is about to adopt to PoS, the community still cannot see a bright future for the time being. Fifthly, Staking cannot save PoS. However, PoS is not utterly useless. PoS is more suitable for being the stake of an enterprise-like community governance rather than a Utility Token.

PoC, Proof of Capacity, is destined to stand out from others since its birth. PoC is divided into two major categories. The first is represented by Burst, which leverages the principle of space-time conversion. The cryptographic hash value in PoW is computed in advance and written to harddisks. When mining, the miner program only needs to scan harddisks to obtain all nonce from special scoop and submit a smallest deadline of its own. The owner of the minimum deadline from the entire network in each round gets the accounting right, generates the block and gets the incentives at the same time. This type of PoC has a low entrance threshold. Ordinary computers can participate in mining as long as equipped with hard drives. The electricity consumption is extremely low while under certain computing power, it can be as safe as PoW. Burst, as the first complete and operational public chain based on PoC, proved the feasibility of PoC. But the poor operation in the later stage led to its failure. As a rising star of PoC, BHD testifies its value when nurturing the market in a large amount. The other is the PoC + PoW hybrid consensus model represented by Filecoin. Although belonging to the PoC category, it requires the capacity of harddisk to match enough PoW computational power (graphics cards or ASICs), which makes the entire consensus algorithm extremely complex. It sets a high technical threshold for miners, and the threshold of initial investment is as high as that of pure PoW consensus model.

Privacy Coins

When it comes to privacy coins, everyone will immediately think about of Zcash, Menero and Grin.

Zcash is a fork of Bitcoin, which changed the hash algorithm of consensus model and added zero-knowledge proofs (zk-SNARKs) into the business logic. It made a splash once and ranked the first place in privacy coins. But its good times did not last long. Currently it has reached the point where the computing power has dropped to be quite vulnerable to the 51% attack.

Menero chooses the privacy algorithm of ring signatures. It has stumbled over the past few years and been constantly patching up to fight against ASICs. It is also the preferred coin for zombie botnet mining networks in mixed reviews.

Grin is a promising new star of privacy coins in 2019. The basic principle is to add the additively homomorphic encryption to the Bitcoin codebase to realize privacy. Due to the privacy only established on the premise that everyone is online, which can be regarded as an online tumbler, that brings much trouble in practice.

The privacy coins above have one thing in common, that is, they all employ PoW consensus and require huge power consumption to maintain the security of chains.

But privacy coins have always been a "pretty rigid demand" in the cryptocurrency world! As a Utility Token, privacy should be a natural and inherent property of cryptocurrencies. Unfortunately, mainstream cryptocurrencies such as Bitcoin and Ethereum are neither private nor anonymous, which leads to the continuous chasing of privacy coins.

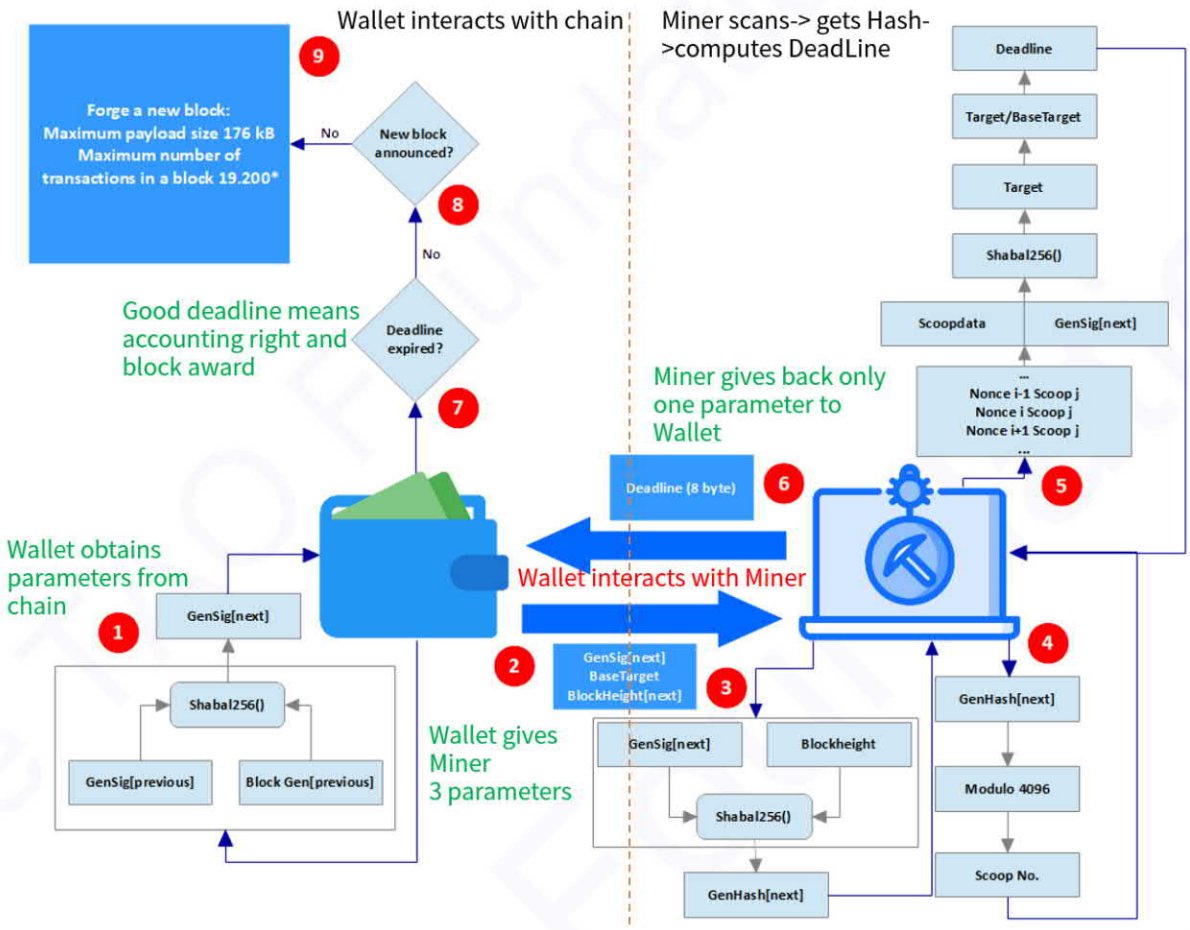
The TAO

Proof of Capacity (PoC) is an excellent consensus algorithm, with the attributes of low barriers to participation, low power consumption, ASICs tamperproof, and public transparency. Zero-knowledge proof (zk-SNARKs) has experienced the baptism of time and proved to be a reliable algorithm for privacy transactions. The TAO is a combination of their advantages. The underlying mining algorithm employs PoC consensus and the upper business logic adopts to zk-SNARKs. Although there are some public chains claiming to realize PoC + zk-SNARKs, it can be found to be a totally fake privacy coin after an easy verification. The TAO is the world's first privacy coin built with PoC + zk-SNARKs!

Technical Architecture

1. Mining Process

Mining is a process of having the accounting right and obtaining the token incentives by submitting minimum deadlines specified by blockchain networks. The mining process of The TAO is that miners plot harddisks in fixed time, get nonce, calculate deadlines and submit to wallet or pool. Miners Start mining with either solo or pool mode. They can choose as they want freely.

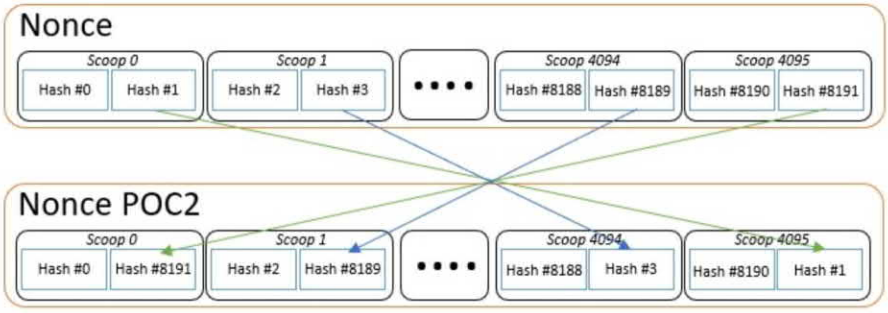


As shown in the figure, mining is divided into 9 steps: 1) Wallet interacts with the chain to obtain current chain parameters; 2) Wallet sends 3 chain parameters to the miner program; 3) Miner program calculates the nexthash value according to the current chain parameters; 4) Miner program calculates the current Scoop number according to the nexthash value; 5) Miner scans all nonce per the current Scoop number within his harddisk capacity and calculates the deadline value; 6) Submit the smallest deadline value of its own to the wallet; 7) The wallet makes a local judgment whether the received deadline value is greater than the default maximum value set by itself. If the deadline is greater than this value, it will be dropped directly and will not be submitted to the chain; 8) Wallet submits the deadline value to the chain; 9) Chain judges whether the accepted deadline is the smallest of all the current ones. If it is the smallest, the accounting right and token reward are given; Otherwise, the submitted deadline value is dropped.

During mining process, the wallet can be on the same physical machine with the miner program, which is called the solo mode. Or it can be on the remote site, thus it is in the pool mode. The TAO will make joint efforts with the community to run an official pool. It does not require all miners to participate in the official pool compulsorily while everyone is encouraged to join in it. There are two categories of miner programs: C ++ and Rust, which can be run under both Windows and Linux platforms respectively. The miner program will be available on the official website, and it is strongly recommended that the community only downloads the related programs from the official website.

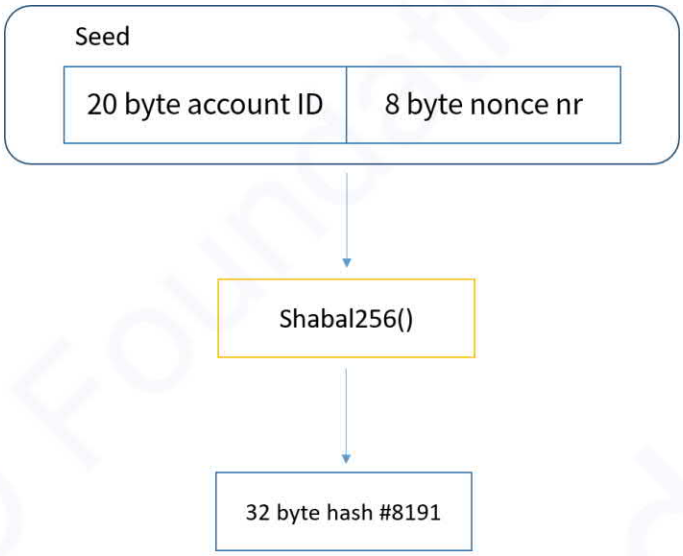
2. Plotting Process

The TAO adopts the PoC consensus, which means a process of making a plot file is required before mining. This process is called plotting harddisks. The plot files of The TAO use the PoC2 format, which is different from PoC format as follows:

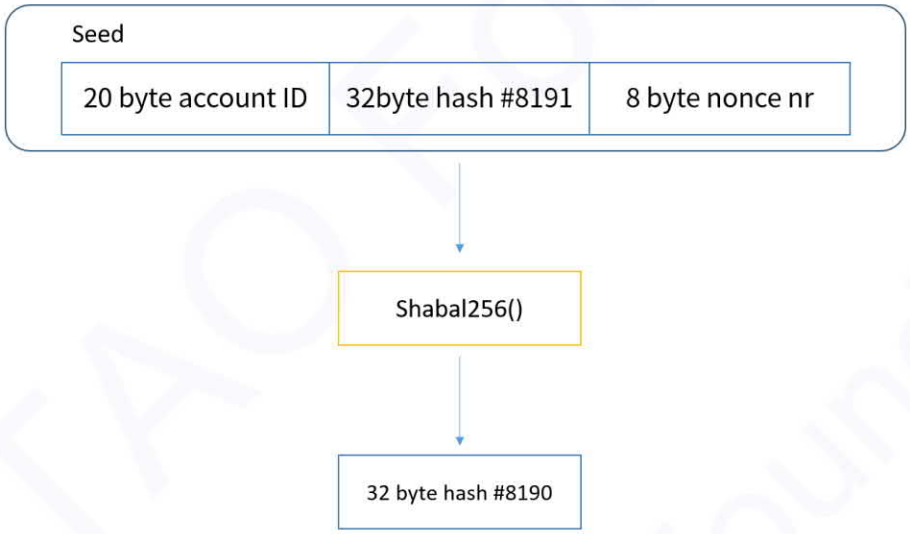


For the purpose of reducing the frequent addressing action of harddisk heads, PoC2 format is optimized. It puts the nonce of the same Scoop number together to improve efficiency and benefits lives of harddisks.

The nonce calculation of The TAO is slightly different from Burst, BHD, etc., which is embodied in the generation of seeds. The TAO uses a 20-byte account ID instead of the usual 8 bytes while is not simply extended from 8 bytes to 20 bytes. The specific algorithm will be reflected in the open source plot program. The cryptographic hash algorithm employs Shabal256. The nonce calculation process of The TAO can be listed as follows:



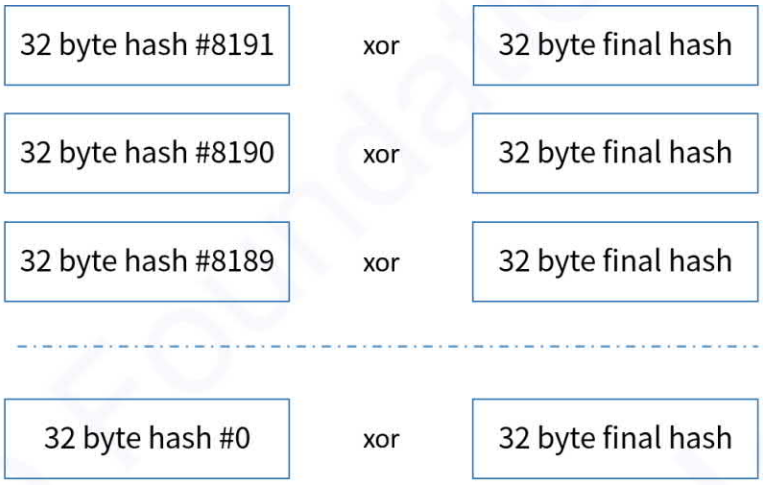
Leveraging the seed, use the Shabal256 algorithm to generate the first hash, that is, the hash with the number 8191. Then take the result as part of the seed to generate the second hash, #8190, by using the same algorithm, Shabal256:



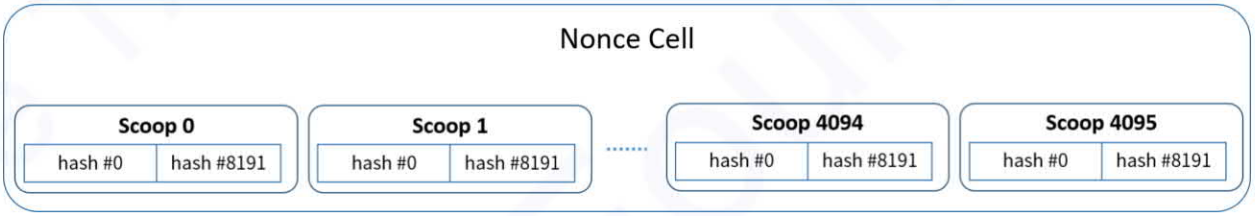
Make recursions consecutively until all 8192 hashes are generated. At this time, a hash called final hash needs to be generated, whose seed is all 8192 hashes plus the original seed with the earliest 28 bytes, namely:

Final hash = Shabal256（Hash #0~8191 + startedseed(20byte account ID+8byte hash nr）)

At last, final hash does xor calculation on each hash generated before, so as to get the final nonce:



The 8192 hash results are arranged in groups of two adjacent ones. Each group is called a Scoop then to get 4096 Scoops, which are filled in the Cell. Thus, the Cell structure is completed.



In the process of generating the Cell, the computer must use the cache to record all intermediate results for obtaining the final nonce result. As you can see, SSD can be used to improve the efficiency of harddisks plotting.

Since each Cell contains 8192 hash results of Shabal256 and the length of each nonce is 32 bytes, each Cell will take up fixed space of 256KB.

Repeat the operation of generating Cells, and then optimize the arrangement of all Cells to fill the Plot files.

The TAO will provide the plot program with graphic cards acceleration based on the Rust language to the community for free.

The core code of the miner in Rust language is described as the flowing:

```rust
use crate::miner::{Buffer, NonceData};
use crate::ocl::GpuContext;
use crate::ocl::{gpu_hash, gpu_transfer};
use crate::reader::ReadReply;
use crossbeam_channel::{Receiver, Sender};
use futures::sync::mpsc;
use futures::{Future, Sink};
use std::sync::Arc;
use std::u64;
use hex;

pub fn create_gpu_worker_task(
    benchmark: bool,
    rx_read_replies: Receiver<ReadReply>,
    tx_empty_buffers: Sender<Box<dyn Buffer + Send>>,
    tx_nonce_data: mpsc::Sender<NonceData>,
    context_mu: Arc<GpuContext>,
) -> impl FnOnce() {
    move || {
        for read_reply in rx_read_replies {
            let buffer = read_reply.buffer;
            // handle empty buffers (read errors) && benchmark
            if read_reply.info.len == 0 || benchmark {
                // forward 'drive finished signal'
                if read_reply.info.finished {
                    let deadline = u64::MAX;
                    tx_nonce_data
                        .clone()
                        .send(NonceData {
                            height: read_reply.info.height,
                            block: read_reply.info.block,
                            base_target: read_reply.info.base_target,
                            deadline,
                            nonce: 0,
                            reader_task_processed: read_reply.info.finished,
                            account_id: hex::encode(&read_reply.info.account_id),
                        })
                        .wait()
                        .expect("GPU worker failed to send nonce data");
                }
                tx_empty_buffers
                    .send(buffer)
                    .expect("GPU worker failed to cue empty buffer");
                continue;
            }

            // consume and ignore all signals
            if read_reply.info.len == 1 && read_reply.info.gpu_signal > 0 {
                continue;
            }

            gpu_transfer(
                &context_mu,
                buffer.get_gpu_buffers().unwrap(),
                *read_reply.info.gensig,
            );
            let result = gpu_hash(
                &context_mu,
                read_reply.info.len / 64,
                buffer.get_gpu_data().as_ref().unwrap(),
            );
            let deadline = result.0;
            let offset = result.1;

            tx_nonce_data
                .clone()
                .send(NonceData {
                    height: read_reply.info.height,
                    block: read_reply.info.block,
                    base_target: read_reply.info.base_target,
                    deadline,
                    nonce: offset + read_reply.info.start_nonce,
                    reader_task_processed: read_reply.info.finished,
                    account_id: hex::encode(&read_reply.info.account_id),
                })
                .wait()
                .expect("GPU worker failed to cue empty buffer");

            tx_empty_buffers.send(buffer).unwrap();
        }
    }
}

```

3. The structure of the Block

The structure of the block and the blockheader are defined in file block.h.

```
class CBlockHeader
{
public:
    // header
    static const size_t HEADER_SIZE=4+32+32+32+4+4+32;
    static const int32_t CURRENT_VERSION=4;
    int32_t nVersion;
    uint256 hashPrevBlock;
    uint256 hashMerkleRoot;
    uint256 hashFinalSaplingRoot;
    uint32_t nTime;
    uint32_t nBits;
    uint64_t nNonce;
    std::vector<unsigned char> nSolution;

    // poc
    uint256 genSign;
    uint160 nPlotID;
    uint64_t nBaseTarget;
    uint64_t nDeadline;

    CBlockHeader()
    {
        SetNull();
    }

    ADD_SERIALIZE_METHODS;

    template <typename Stream, typename Operation>
    inline void SerializationOp(Stream& s, Operation ser_action) {
        READWRITE(this->nVersion);
        READWRITE(hashPrevBlock);
        READWRITE(hashMerkleRoot);
        READWRITE(hashFinalSaplingRoot);
        READWRITE(nTime);
        READWRITE(nBits);
        READWRITE(nNonce);
        READWRITE(nSolution);

        READWRITE(genSign);
        READWRITE(nPlotID);
        READWRITE(nBaseTarget);
        READWRITE(nDeadline);
    }

    void SetNull()
    {
        nVersion = CBlockHeader::CURRENT_VERSION;
        hashPrevBlock.SetNull();
        hashMerkleRoot.SetNull();
        hashFinalSaplingRoot.SetNull();
        nTime = 0;
        nBits = 0;
        nNonce = 0; //uint256();
        nSolution.clear();

        genSign.SetNull();
        nPlotID.SetNull();
        nBaseTarget = 0;
        nDeadline = 0;
    }

    bool IsNull() const
    {
        return (nBits == 0);
    }

    uint256 GetHash() const;

    int64_t GetBlockTime() const
    {
        return (int64_t)nTime;
    }
};

```

And the block’ s structure is:

```
class CBlock : public CBlockHeader
{
public:
    // network and disk
    std::vector<CTransaction> vtx;

    // memory only
    mutable std::vector<uint256> vMerkleTree;

    CBlock()
    {
        SetNull();
    }

    CBlock(const CBlockHeader &header)
    {
        SetNull();
        *((CBlockHeader*)this) = header;
    }

    ADD_SERIALIZE_METHODS;

    template <typename Stream, typename Operation>
    inline void SerializationOp(Stream& s, Operation ser_action) {
        READWRITE(*(CBlockHeader*)this);
        READWRITE(vtx);
    }

    void SetNull()
    {
        CBlockHeader::SetNull();
        vtx.clear();
        vMerkleTree.clear();
    }

    CBlockHeader GetBlockHeader() const
    {
        CBlockHeader block;
        block.nVersion       = nVersion;
        block.hashPrevBlock  = hashPrevBlock;
        block.hashMerkleRoot = hashMerkleRoot;
        block.hashFinalSaplingRoot   = hashFinalSaplingRoot;
        block.nTime          = nTime;
        block.nBits          = nBits;
        block.nNonce         = nNonce;
        block.nSolution      = nSolution;

        block.genSign = genSign;
        block.nPlotID = nPlotID;
        block.nBaseTarget = nBaseTarget;
        block.nDeadline = nDeadline;

        return block;
    }

    // Build the in-memory merkle tree for this block and return the merkle root.
    // If non-NULL, *mutated is set to whether mutation was detected in the merkle
    // tree (a duplication of transactions in the block leading to an identical
    // merkle root).
    uint256 BuildMerkleTree(bool* mutated = NULL) const;

    std::vector<uint256> GetMerkleBranch(int nIndex) const;
    static uint256 CheckMerkleBranch(uint256 hash, const std::vector<uint256>& vMerkleBranch, int nIndex);
    std::string ToString() const;
};

```

4. Zero-Knowledge Proof（zk-SNARKs）

A zero-knowledge proof is a digital protocol that allows for data to be shared between two parties without the use of a password or any other information associated with the transaction.

In its most basic sense, a zero-knowledge proof (also commonly referred to as ZKP) can be thought of as a protocol through which a digital authentication process can be facilitated without the use of any passwords or other sensitive data. As a result of this, no information, either from the sender's or receiver's end, can be compromised in any way.

This is quite useful, especially since such a level of safety provides tech enthusiasts with an avenue to communicate with one another without having to reveal the content of their interactions with any third party.

The idea underlying zero-knowledge proofs first came to the fore back in 1985, when developers Shafi Goldwasser, Charles Rackoff and Silvio Micali presented to the world the notion of "knowledge complexity" — a concept that served as a precursor to ZKPs.

As the name suggests, knowledge complexity acts as a metric standard to determine the amount of knowledge required for any transaction (between a prover and verifier) to be considered valid.

The basic of Zero-Knowledge Proof protocol is interactive. It requires the verifier to constantly ask a series of questions about the "knowledge" the prover possess.

Non-interactive Zero-Knowledge Proof, as the name implies, do not require an interactive process, avoiding the possibility of collusion, but may require additional machines and programs to determine the sequence of experiments.

Zcash is the first widespread application of zk-SNARKs, a novel form of zero-knowledge cryptography. The strong privacy guarantee of Zcash is derived from the fact that shielded transactions in Zcash can be fully encrypted on the blockchain, yet still be verified as valid under the network's consensus rules by using zk-SNARK proofs.

The acronym zk-SNARK stands for "Zero-Knowledge Succinct Non-Interactive Argument of Knowledge," and refers to a proof construction where one can prove possession of certain information, e.g. a secret key, without revealing that information, and without any interaction between the prover and verifier.

zk-SNARK converts the transaction content that needs to be verified into a proof that two polynomial products are equal, combined with homomorphic encryption and other advanced techniques to protect the hidden transaction amount while performing transaction verification. Its process can be briefly described as:

- Disassemble the code into verifiable logical verification steps, then disassemble these steps into an arithmetic circuit consisting of addition, subtraction, multiplication, and division.
- Conduct a series of transformations to convert the code to be verified into a polynomial equation, such as t(x)h(x)= w(x)v(x).
- In order to make the proof more concise, the verifier randomly selects several checkpoints,s, in advance to check whether the equations at these points are true.
- By homomorphic encoding/encryption, the verifier does not know the actual input value when calculating the equation, but can still verify.
- On the left and right hand sides of the equation, multiply a secret value k that is not equal to 0. when verifying that (t(s)h(s)k) is equal to (w(s)v(s)k), The specific t(s), h(s), w(s), and v(s) cannot be known, so the information can be protected.

The core math model is:

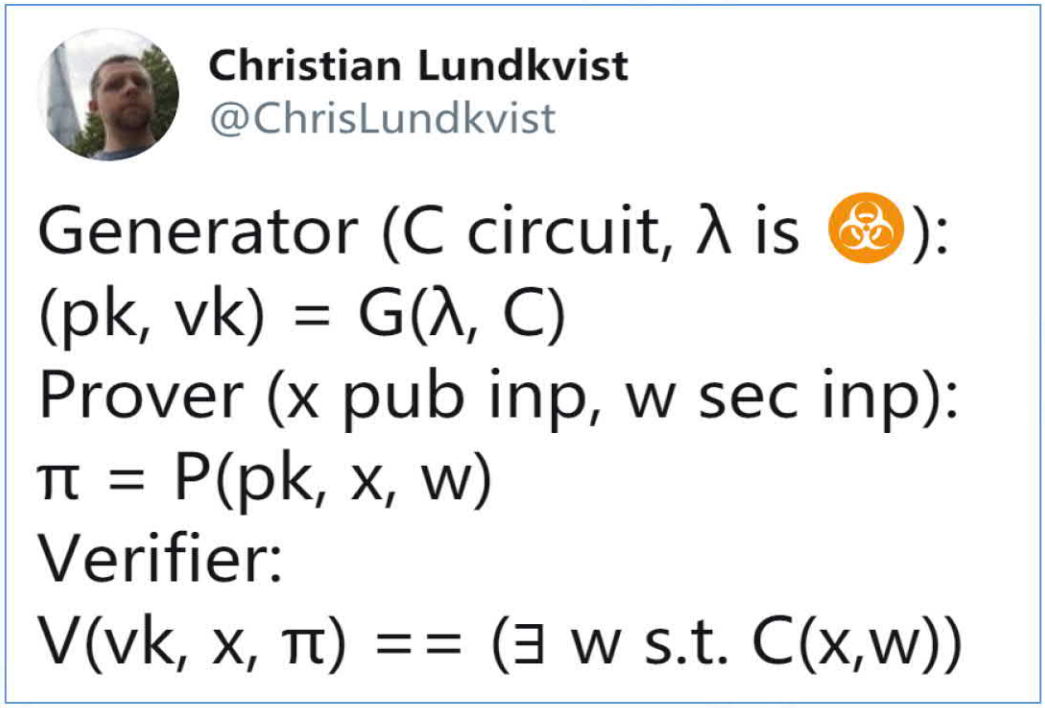
$$\Pr\left[\sigma \leftarrow \mathrm{Setup}(1^k) : \mathcal{A}^{\mathrm{Prove}(\sigma,\ldots)}(\sigma) = 1\right] \equiv \Pr\left[(\sigma,\tau) \leftarrow \mathrm{Sim}_1 : \mathcal{A}^{\mathrm{Sim}(\sigma,\tau,\ldots)}(\sigma) = 1\right]$$

Here $\mathrm{Sim}(\sigma,\tau,y,\omega)$ outputs $\mathrm{Sim}_2(\sigma,\tau,y)$ for $(y,\omega) \in R\sigma$ and both oracles output failure otherwise.

The TAO generates 5 parameters through the MPC (Multi-party Computation) protocol. The first 2 parameters are the proving key (pk for short) and the verifying key (vk for short). The following 3 parameters are the circuit parameters.

pk and vk are generated by generator G according to λ and C.

λ: Random factor, which is also called poison pill.



(From：https://twitter.com/ChrisLundkvist/status/799807876982251520)

During the process of generating pk / vk , multiple participants contribute their own random numbers, and the sum of shared randomness is λ. The MPC protocol works that as long as only one participant actually destroys the random number contributed by himself honestly afterwards, the security of pk and vk can be guaranteed, which means no one can forge the proofs.

C: Circuit transformation function, Circuit.

The code demonstrates as r1cs or cs, rank-1 constraint system.

P: Proving function.

V: Verifying function.

The three steps of zero-knowledge proof are as follows:

1. G calculates pk and vk according to the input λ and C. For some specific blockchains, pk and vk only need to be calculated once and revealed publicly.

2. The prover calculates the proving value π = P (pk, public input x, secure input w).

3. The verifier calculates V (vk, public input x, π) to be true, then confirms that the prover holds the secure input w to make C (x, w) true.

The r1cs parameter file is generated by running the Taod program with command-line parameters:

taod -savesproutr1cs -experimentalfeatures

We will hold a Trusted Setup Ceremony to generate privacy proving parameters for zk-SNARKs.

5. Wallet

We do not reinvent the wheel. Our wallet is compatible with the standard bip39/32/44 protocols, which means we have a universal Mnemonic and a Hierarchical Deterministic (HD) derivation sub-address protocol. Under the premise of guaranteeing the security, it not only greatly increases convenience and usability, but also lay a good foundation for wallet on mobile APPs. We will also launch our own hardware wallet as the first hardware wallet in PoC ecosystem to escort the community's assets.

Our wallet addresses are divided into two categories: transparent addresses starting with 7- and privacy addresses starting with tao-. All addresses support HD sub-address derivation.

6. Others

The TAO will provide the community with a complete ecosystem of tool chain and facility tools from plot programs to miner programs, from desktop wallets to mobile wallets, from block explorers to mining pools. All programs will be open source.

The TAO will be the first to creatively adopt the approach of delayed open-sourcing process of the source code. We package the source code and calculate its SHA512, and this hash value will be written to the genesis block. This move can not only make the source code delayed open-sourcing for about half a year to resist copycats in the TAO's young fragile ecosystem, but also prove that the code was genuine from the day of the genesis block generated.

Economic Model

The TAO adopts an economic model of deflation. It belongs to Utility Token, and the symbol is TAO.

- Total Supply：21,000,000 TAO
- Initial Block Reward：50 TAO
- Block Interval：5 minutes
- Deflation Strategy：Block reward halves every 210,000 blocks
- No pre-mining

Countermeasures

Any public chain is facing various types of attacks, and The TAO is no exception. We have taken the following countermeasure policies:

1. Gang of Miners Attack

In PoC ecosystem, gang of miners is not uncommon. They savagely loot PoC mining coins like locusts, grab short-term economic benefits, and cause serious damage to the ecosystem. Keep this in mind, we have made a countermeasure policy. First is the plot parameters. We have modified the plotting algorithm, the length of account ID turns from 8 bytes to 20 bytes which is not a simple digital expansion. It disables mining bully to switch on large amount of computing power for benefits by using harddisks already plotted in early stage when the computing power of entire network is not big enough.

Secondly, we split the mainnet into pre-mainnet and mainnet phases which is ruled by the whole net capacity power and block height.

2. 51% Attack

PoC algorithm has a 51% attack problem similar to PoW. To end this, we will adapt the latest algorithm.

3. Privacy Disclosure

We keep close track of the cutting-edge zk-SNARKs and zk-STARKs algorithm and adopting the latest research results in real time to maintain the robust stability and privacy of The TAO network.

4. Cold Startup

The startup of TAO mainnet splits into two stages, the pre-mainnet and the mainnet. The difference is the whole net's 'capacity-power'. The trigger value is 100P and the block height. The block reward during pre-mainnet is split:

(0, 1P) && height<2016 miner vs foundation 1:9

[1P, 10P) && height<6048 miner vs foundation 2:8

[10P, 50P) && height<14112 miner vs foundation 3:7

[50P, 100P) && height<22176 miner vs foundation 5:5

[100P, ∞) miner: 100%

That is to say, when we have 100P capacity-power, we enter mainnet stage.

Our Team

The TAO is the first anonymous coin from PoC ecosystem. we are geeks and white-hat hackers from all over the world. Although coming of a noble background, we would like to keep anonymous instead of letting the great fame behind become a burden to The TAO. We dedicate our math and crypto skills to the community of blockchain and the whole society.

Our email address is thetao[at]riseup[dot]net.

Don't call us, we will call you if you are special enough.

Community

The health, stability and sustainable development of ecosystem are closely associated with the community. We value and respect the community. The core team will make joint efforts with the

community to guide and promote the development of The TAO. The evangelists of the PoC have already laid a small space for us. With greeting to them, we start the global recruiting of community manager and volunteers.

Let us work together to create the glory of PoC + zk-SNARKs!

Roadmap

The TAO will launch the mainnet directly to make a privacy coin for PoC ecosystem and even the whole world.

In the next step, we will complete the integration with the private IM tool at a rate of a large version updating every 6 months, introduce NFT, DeFi and Oracle, realize a real privacy coin wallet with zero-knowledge proof for mobile app and a privacy stable coin base on PoC consensus, and so on.

In Q1 2021, we will list in DEX. Before the end of year 2021, launch in 1 ~ 2 renowned big exchanges.

All glorious things are happening quietly, so stay tuned ...

Acknowledgements

Greeting to Satoshi Nakamoto，Xiaoyun Wang，Vitalik Buterin，Zcash Core Team，Burst Core Team，BHD Core Team &Community，and countless friends and families behind us.

Love you, forever!

References

https://bitcoin.org/en/bitcoin-paper

https://ethereum.org/learn/

https://z.cash/technology/

https://www.burst-coin.org/

http://www.btchd.org/