0Chain - decentralizing storage[1]

Saswata Basu, Tom Austin, Siva Dirisala and 0Chain team

Aug 4, 2019

Abstract

There are two relevant data trends. Data is expected to grow from 33 ZB (zettabytes) today to 175 ZB in 5 years[2]. And the cloud is moving to the edge for performance and availability, driven by IoT applications, multi-player gaming, autonomous vehicles, and content streaming. Decentralization would accelerate this change and adoption, as it lowers deployment, management, and scale-out cost. To this end, 0Chain is decentralizing storage.

0Chain dStorage is cheaper and higher performance than traditional cloud. The protocols provide a layer of privacy, security, transparency, and service assurance. For consumers, the benefits are privacy, anonymity, and transparency. Developers have better customer data protection at a lower cost. Enterprises can scale out their data protection at a lower cost. For MSPs, dStorage provides a higher revenue potential.

The dStorage platform is built on 0ChainNet, a permission-less, fast finality, scalable blockchain, built from scratch in Golang. 0ChainNet protects its network from Sybil with a Nonlinear Proof-of-Stake protocol, and prevents blockchain stalls from DDoS attacks by using multiple leaders. Client protection is accomplished by using a Serverless 2FA protocol for individuals, and a cryptographic multiple signature protocol for exchanges and businesses.

0ChainNet offers innovative token economics which enables users and developers to get “free” services, such as transactions and storage. Users can lock ZCN tokens[3], like a bank CD, to get interest tokens immediately. Storage providers (“blobbers”) need to stake ZCN tokens to receive expected payment. As more applications use our network, ZCN will grow in its intrinsic value relative to the data stored on the network, and tokens locked for interest, as users interactively lock and stake tokens to participate in the ecosystem. In this sense, ZCN is the *first crypto asset tied to data and interest.*

Unlike Bitcoin, Ethereum, and other projects, 0Chain inflation is primarily driven by token holders desire to mint “interest” tokens, which are given to token holders for locking or staking their tokens, rather than “reward” tokens given to miners only. 0Chain has recently re-tooled its token economics to provide on-going rewards to support the community of developers and ambassadors as well. In addition, 0Chain has allocated a portion of team tokens to fund the reward tokens for the first 4 years after mainnet to maintain low inflation. After 4 years, the rewards will be generated by the network and the interest will be set at 5% to maintain an average target inflation rate of about 3-4%. This provides benefits to the entire ecosystem - miners, developers, ambassadors, and token holders at a low inflation rate.

1 dStorage Platform

The motivation behind the design of the platform is to offer a cheaper, faster, more secure and more available storage than traditional cloud. Let’s discuss some inherent challenges with offering such a service in a decentralized environment.

a) How to protect against data loss?
b) How to ensure that even if one or some of the decentralized parties are down, the overall service is not effected for every single stored file or object?
c) How to make certain that the decentralized parties are really storing the data for which they are getting rewarded?
d) How to avoid currency fluctuation for our managed storage provider (MSP) customers?
e) How do dApps preserve data consistency among multiple parties to execute their smart contracts?

Some challenges are inherent and relatively new to the decentralized systems. Below, we will explore each of these challenges and the approach 0Chain dStorage has taken. This has been also discussed in the EthResearch forum[4].

---

[1] Several patents pending

[2] IDC projection for data

[3] 0Chain tokens or tokens in this white paper refer to the utility of an entity used for compute and storage resources

[4] https://ethresear.ch/t/dstorage-better-than-traditional-cloud/5907/3

1.1 Data Availability

Traditional storages offer two types of solutions. One is replication where each data is stored multiple times and in multiple locations so when one disk or location has a fateful incident, the data is recoverable from the other. This is what exists today at enterprises when they back up their storage at two separate locations, in addition to a local copy on premises.

The other option is to store the data using erasure coding technology. Erasure coding can be best understood as a set of equations required to solve a set of variables. If there are 4 variables, one needs 4 equations to solve them. Anything more either produces inconsistent result or is redundant. An example encoding is 10-15 erasure coding where 10 original data blocks go through expansion into 15 blocks. This gives a safety where even with the loss of any 5 of the 15 blocks, the original 10 blocks can be recovered.

0Chain storage uses the erasure coding technology, as shown in Fig 1, to ensure data is available, and benefits from cost savings of 50%. The availability cost is 1.5 times the original data, instead of 3 copies, and the availability gained is 14 nines compared to 4 nines in a 3-copy replication. More on this in Section 6 as to how it improves cost dynamics for enterprises and SMEs.

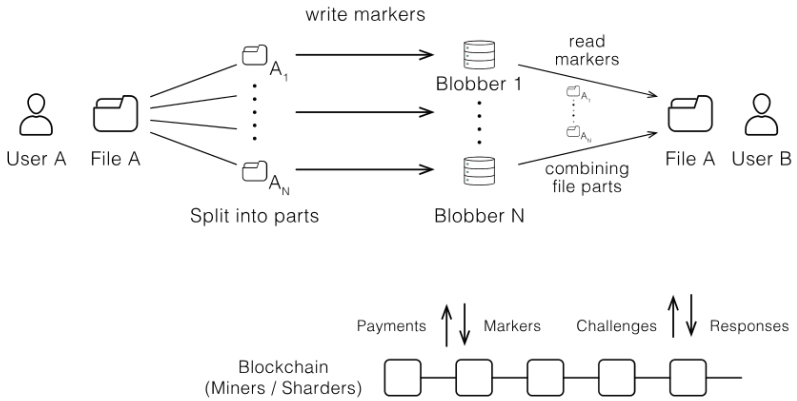

Fig 1: 0Chain Storage protocol where the file is split into parts and sent to different storage providers (blobbers)

1.2 Data Performance & Reliability

A 10/15 erasure coding helps us with data performance and reliability.

1) With 5 blobbers down, the original data can still be restored from the remaining 10. As these are independent blobbers, their failure (hardware, network, natural disaster) probability is mutually exclusive, resulting in a very high reliability.
2) Data can be uploaded/downloaded in parallel to all the 15 blobbers. This results in a tremendous network time savings. For example, a 10 MB file expands into 15 MB due to erasure coding, but each blobber is uploaded only a 1 MB encoded file. So, in effect, the upload time will be as if the client is uploading just a 1 MB file. Of course, the assumption here is that the client has enough network bandwidth to upload 15 MB simultaneously to 15 different service providers. But the point is, the upload time is now dependent on the bandwidth of the client rather than the bandwidth of a single server on a traditional storage platform.
3) When downloading data from multiple servers of 0Chain storage, the client tries to download it from all the 15 blobbers but it only needs to wait for the first 10 to reconstruct the data. This results in a fast and smooth download experience.

1.3 Always-on Data Protection

Data protection is an issue for a traditional storage providers as well, but in a different context than Byzantine conditions, and it is more due to bit rotting and issues related to hardware. For this, the traditional storage systems have a means of self-validating and recovering.

In a decentralized system, there is no way to enforce the operational practices and quality of each service provider just by policy. This assurance needs to be part of the protocol. Just like in Proof Of Stake based consensus algorithms that offer rewards as well as slashing of stake, the service providers are rewarded and punished for passing or failing a challenge that ensures that the provider is doing what is required. 0Chain uses smart contract driven challenges for the blobbers. The challenge itself is completely random (but deterministic for a given block to ensure consensus). The blobber and the specific file and a fragment within the file is picked for a challenge in a random manner. Such randomized systems have statistical outcomes that can be tuned to achieve a desired level of quality of service.

Ensuring high quality requires doing frequent challenges. Hence the challenge protocol should be as light as possible in terms of bandwidth requirements. At the same time, it is important to be able to validate as much as possible. Say a file is stored in 64KB blocks and the file size is 10MB. Challenging the entire file content stored at a blobber (1MB after erasure coding) is a waste of lot of network bandwidth. Challenging any of the random blocks reduces it to just 64KB overhead. However, this has a potential for a blobber to download the content, reconstruct it and then serve the 64KB he is supposed to have stored. Note that 0Chain storage offers both private and public content. The private content can only be read by the owner or any user authorized by the owner. Hence, a blobber will not be able to download the private content to pass the challenge. The following attack scenario and the defense against is valid only for public content although the challenge behaves exactly the same in both scenarios.

1.4 Outsourcing Attack

Our protocol[5] avoids outsourcing attack by ensuring that the content provided for verification is 64KB but the content required to create this verified content is the full file fragment. This is done as follows. The file fragment of 1MB stored with the blobber is divided into fifteen 64KB blocks. Each of these 64KB blocks are further are divided into 64 byte chunks (just for discussion and final chunk size will be fine tuned or dynamic). There are 1024 such chunks in each 64KB block that can be addressed using an index of 1 to 1024. Now, imagine that the data at each of these indexes across the blocks is treated as a continuous message and hashed. Then these 1024 hashes serve as the leaf hashes of the Merkle Tree. The root of this Merkle tree is used to roll up the file hashes further up to directory/volume level. The Merkle proof provides the path from the leaf to the file root, and from the file root to the volume level. With this model, in order to pass the challenge for a file for a given index (between 1 and 1024) a dishonest blobber first needs to download all the content and do the chaining to construct the leaf hash. This discourages them to avoid storing the content and engage in an outsourcing attack.

1.5 Validators

In addition to blobbers, the dStorage also relies on a special role called Validators. Their role is to validate and give verification signatures that the blobbers can collect and with enough signatures, they establish their validity to the blockchain. The fact that there are multiple validators is beneficial for the following reasons

1) Scales with storage growth and number of blobbers on the network
2) Individual validators cannot get the blobbers punished
3) Increases the reliability of the network in case of validator downtime

1.6 Parallel Async blockchain transactions

All reads and writes have to get registered on the blockchain for the following reasons

1) the payments to the blobbers are happening on the blockchain via the storage smart contract (hence completely decentralized)
2) there is an audit trail of every read and write to the storage. This audit trail is useful in creating not only security driven applications but also monetization applications that needs an indisputable proof of what actions have been performed by users for which the content owners need to be paid.

Because we need to record every read and write on the blockchain at first it might indicate a bottleneck compared to a traditional storage model. 0Chain protocol is designed specifically to reduce this bottleneck. This is done by making the blockchain transactions asynchronous to the actual read/write operations. This is done by making use of read and write markers that can be cryptographically verifiable. The users present these markers to the service providers and get started right away and the service providers redeem these markers for rewards offline. As the blockchain itself is trusted, as long as the markers are valid, the service providers are guaranteed to be paid.

1.7 Preserve data consistency among validators for oracles and ledger constructs

A decentralized app (dApp) needs to provide a auditable transaction history and provide data consistency among multiple parties in order for them to execute the data on the smart contract. This is detailed on the Hyperledger forum[6].

Proxy ReEncryption[7]

Proxy re-encryption is a technique where a party can encrypt and save the content with a 3rd party and later share the encrypted content securely with anyone by providing *proxy re-encryption* keys. These keys are tied to the public/private keys of the

---

[5] Storage on the 0Chain Network:the Blockchain Observable Storage System (BOSS), P.Merrill, T. Austin. https://drive.google.com/file/d/1tNjb_OvX93s47QQ2gFrfeUBdmP0PcL9k/view

[6] https://lists.hyperledger.org/g/fabric/topic/does_hyperledger_have_a_way/32685809?p=,,,20,0,0,0::recentpostdate%2Fsticky,,,20,2,0,32685809

[7] Sharing of Encrypted files in Blockchain Made Simpler, S. Selvi and A. Paul, S. Dirisala, S Basu, and C. Rangan, https://eprint.iacr.org/2019/418

receiving party and the re-encryption process is performed by the 3rd party storing the content before serving the content to the receiver.

dStorage usage by an app leveraging decentralized storage

The diagram in Fig 2 provides an overview of how an app can integrate with dStorage to leverage decentralized storage for HyperLedger (same is applicable for oracles that work with smart contracts on blockchains such as Ethereum and EOS, and other ledgers such as Corda, Ripple, Steller, Iota). The use case is an enterprise app that is collaborating with multiple parties using HyperLedger wants to share the documents while submitting the HyperLedger transaction. However, since HyperLedger is not suitable for storing large amounts of data like documents, the document is first uploaded to dStorage, and then the hash of the content is submitted to in the HyperLedger transaction along with the PRE keys for each endorser that needs to verify the content. The endorsers then download the content from dStorage and confirm that the hash of the content matches with that provided in the transaction. This ensures that all the parties can trust that they have received the same document pertaining to that transaction.

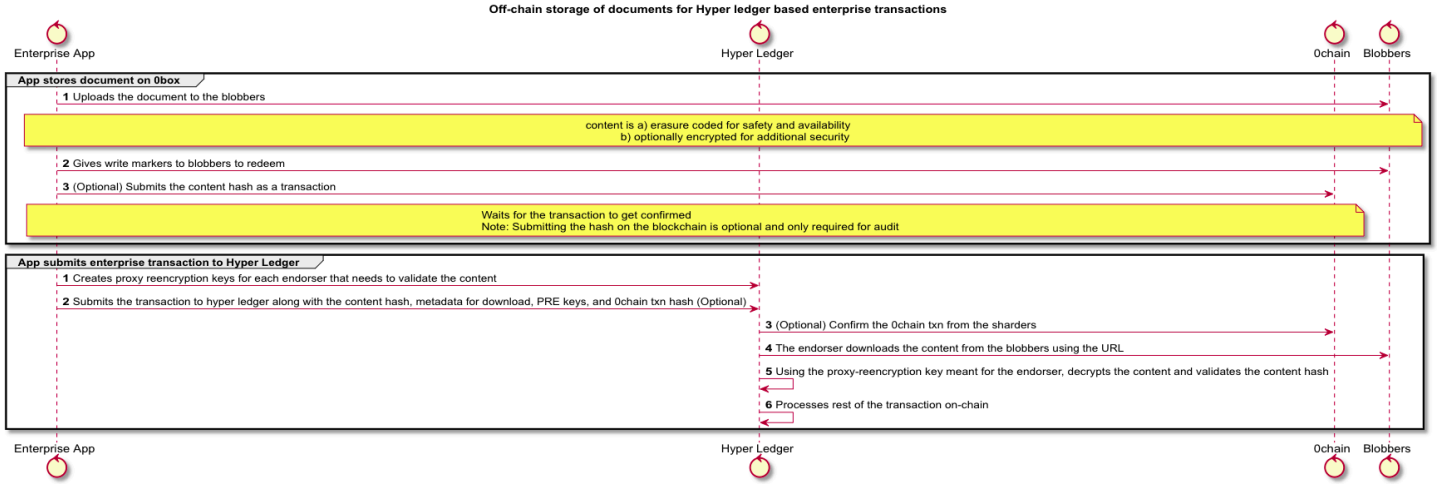

Fig 2: Off-chain storage of documents for Ethereum and Hyperledger based enterprise transactions

1.8 MSP payments in USD

When offering decentralized storage to enterprises, they may prefer to pay for the services using fiat currency. Similarly, service providers might want to receive some of their payments as fiat currency while keeping the rest in the native token value of the blockchain. While smart contracts on the blockchain can keep track of rewards in the native token, there is no standard provision to allow the rewards to be tracked as a mix of tokens and fiat currency.

0Chain offers such a mechanism of accumulating the rewards in dual units. This is achieved by tracking an exchange rate between the native token and any fiat currency, say USD on a regular basis. The exchange rate itself is tracked via a smart contract that allows changing the rate using some type of governance, for example, a multi-signature voting system, there by ensuring that the exchange rate used is publicly verifiable and trusted. Once such an exchange rate is available for the smart contracts, then upon computing the rewards in the native token, the smart contract can split the rewards into native token amount and the rest tracked as fiat currency using the currently tracked exchange rate on the blockchain. The native tokens are directly transferred to the service providers wallet on the blockchain while the fiat currency balances are tracked separately on a smart contract. Using this method, the actual payment using the fiat currency can happen periodically, say once a month, off-chain. When the payment happens, the corresponding fiat balance needs to be adjusted on the blockchain. The transaction that does the adjustment can provide the off-chain payment as a proof as part of the transaction payload. It can be in the form of a digital image, a bank check or a URL that provides a payment information at a 3rd party such as a bank or payment gateway.

For a stable coin, the interactions will be similar to this construct with just the replacement of fiat currency. The MSP payment model is depicted in Fig 3.

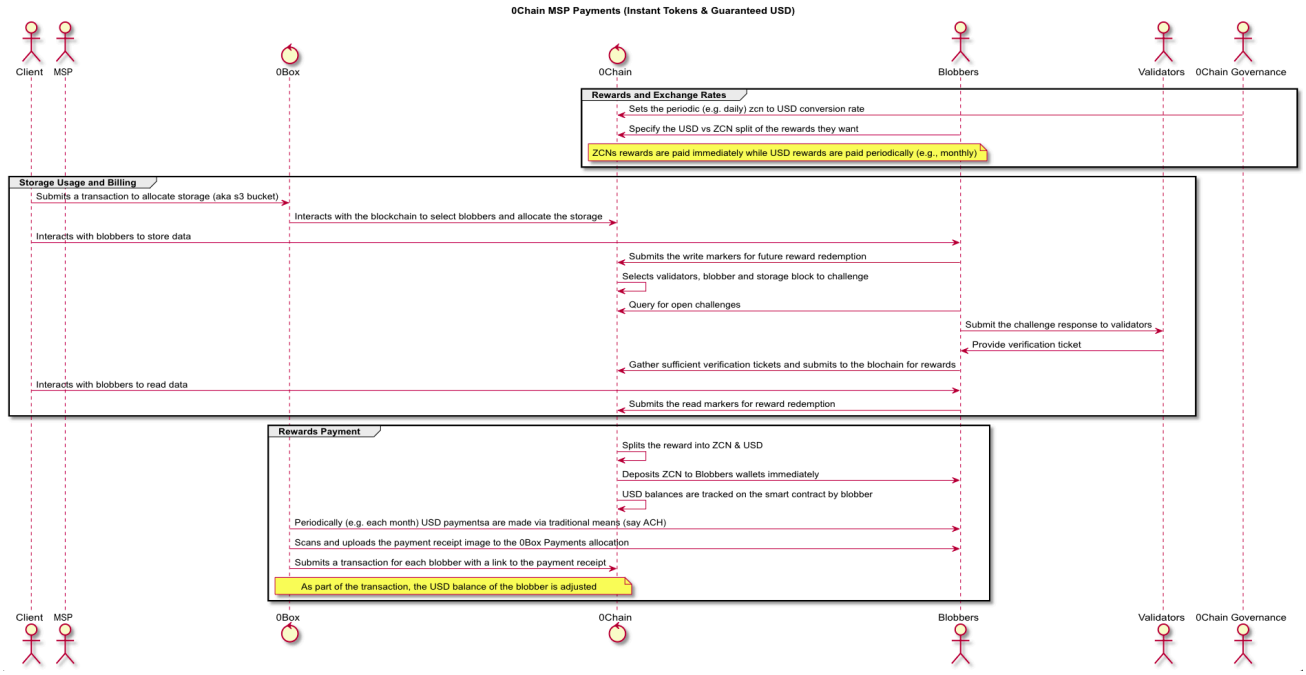

Fig 3: MSP payments in USD

2 Overview of 0Chain Blockchain[8]

In order to bring a fast, scalable, and secure dStorage platform, we needed a fast, secure, and scalable blockchain. 0Chain's blockchain architecture is described in Fig 4, where we have parallel processes and separation of roles in order to achieve high speed and scalability. To this end, we separated the role of a miner into a miner, sharder, blobber, and a validator, so they can scale independently. Transactions are submitted to miners, and they validate them and generate blocks, and send notarized blocks to sharders, who store the block and respond to queries. This way the miner is not getting bombarded by submissions and queries, and we can achieve a faster response time. The blobbers store data, and the validators challenge the blobbers. The clients interact directly with the blobbers when they upload or download data. The validators interact with blobbers as they validate their challenge response. The blobbers submit successful challenge response to the blockchain for tokens. These off-chain processes and transactions make the storage and blockchain platform scalable and fast.

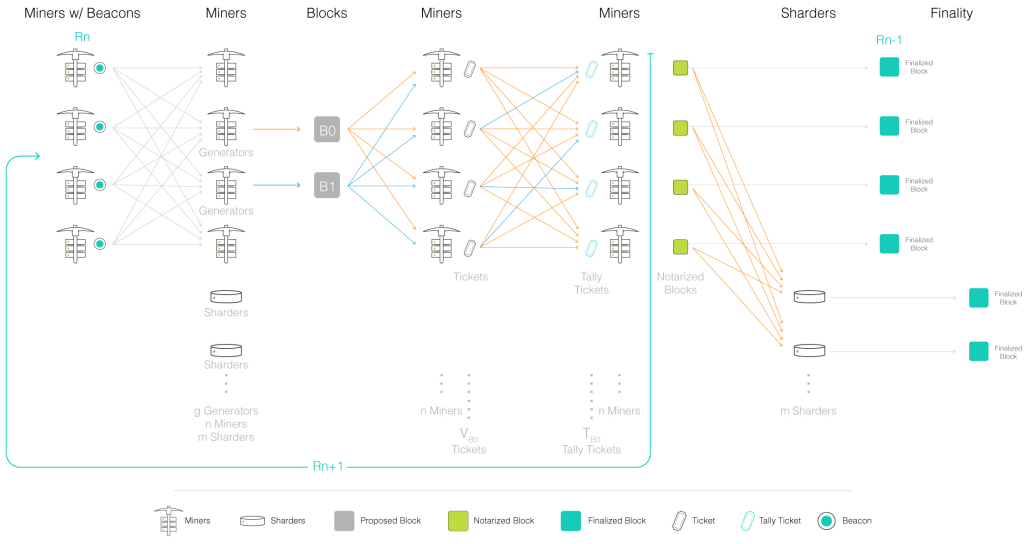

Fig 4. 0Chain Consensus protocol

[8] "The 0Chain Consensus Protocol", J. Katz, T. Austin, S. Dirisala, S. Basu, https://drive.google.com/file/d/1KcfkQ1HmtGXvXzZtalNkVHMmIcwOiA7k/view

Multiple leaders

Decentralized systems need to be both fault and Byzantine tolerant. Among the Proof-of-stake (PoS) systems, some protocols choose to do consensus by electing a leader and change the leader regularly and also when the leader is not making progress. A problem with a single leader based system is that it can be prone to external attacks such as DDoS. Hence, these protocols are typically suitable for private or permission networks.

For a public blockchain, it is desirable to have a protocol that can effectively defend against both Byzantine conditions and also external factors like network outages and DDoS attacks. Some protocols such as Dfinity propose having multiple leaders so that the blockchain will make progress even if some of them are faulty. 0Chain adopts a similar approach where more than one miner can generate a block in a given round and while the network collectively works towards notarizing the highest ranked block in a given round, the goal is to eventually notarize and make progress with any of the blocks that got generated in the round and pass the verification process (transaction signature validation, no replay of transactions and so on). In order to achieve consensus without a leader, and to mitigate the risk of network attacks, consensus is achieved by everyone sending their verification messages to every other miner on the network. We use the same logic as Dfinity regarding block weight which is 1/2^r where r is the rank of the generator starting from 0. So, the block weights go as 1, 0.5, 0.25 and so on. Verifiers give preference to the highest weighted block. In terms of view change, we differ from Dfinity. They require an individual node and a group to enter into the blockchain after an “epoch” or view change. We allow anyone to join into the upcoming view change. And, we only have a single group.

Sybil protection

0Chain uses a non-linear proof-of-stake (NL-PoS) staking approach. Note that the mainnet will be launched with squared power of stake. The idea behind this is that someone with a lot of tokens would prefer to pool them all together to have a better chance participating, rather than splitting into several parts and joining the network separately.

2.1 Deterministic Finality

If a protocol doesn’t require explicit endorsement of a block to reach consensus and just rely on other factors such as PoW, it is possible to disclose blocks much later and alter the course. Many PoS protocols require direct interaction with each other to reach consensus, using a variation of pBFT (practical Byzantine fault tolerance). Immediate direct interaction ensures consensus is discovered quickly and it will not be possible to release blocks at a later time. This makes these protocols to be almost deterministic compared to what is possible with Bitcoin for example. When multiple blocks can be proposed for a given round, it is sometimes possible to not be able to determine if a block is finalized because of network conditions. This is where protocols such as Hashgraph take a different approach. They rely on the fact that the current proposals are consistently endorsing past transactions and when a majority of the miners endorse directly or indirectly a given transaction, it gets finalized. Transactions get finalized individually but the same concept can be applied at the block level. That is, when enough miners extend directly or indirectly off of a given block, it is possible to agree on which of the notarized blocks in a round should be finalized deterministically. 0Chain executes finalization based on locally available information but also keeps track of the deterministic finality using the Hashgraph like logic applied at the block level. In our experiments we have not seen any rollbacks of finality computed based on local information even after running the blockchain for hundreds of thousands of blocks. The clients interacting with 0Chain have several choices on deciding on when to consider their transaction is finalized. They can choose to wait a certain number of blocks, they can choose to query the confirmation from multiple nodes or do a combination of both.

In 0Chain, there are three blocks that are of interest. They are:

1. The block that is being added to the current round
2. The block that is probabilistically finalized using a variant of dFinity algorithm
3. The block that is deterministically finalized using a logic explained below.

It should be noted that the deterministic finalized block lags the probabilistic finalized block which lags the current round block (If all miners generate, then it is possible to have deterministic finality as fast as probabilistic finality).

When a miner receives a block for a round and it is from a valid generator for that round, it is added to the local cache of blocks. During that time, the chain from that block is walked back till the previously identified latest deterministic finalized block. For each block in between, it is added to unique extensions map of the block if the miner of the current block hasn’t extended any block between this intermediate and current block. After sufficient progress of the chain, at some point each block that is probabilistically finalized will receive enough number of unique extensions, indicating that sufficient number of miners are working on top of that block. At that point, the block becomes deterministically finalized. The threshold used for deterministic finality will be the same as the threshold used for notarization of a block. For example, for a 3*f+1 miners with f number of Byzantine miners, there should be more than 2/3rd unique block extensions for a block to be considered finalized deterministically.

The diagram in Fig 5 indicates the above explained process. The letters in the block indicate the miner who generated the block. The letters in the green boxes below represent the unique chain extension endorsements for those blocks. Also shown are additional block proposals C and B received in a particular round.

Let's assume that receiving 5 unique extensions is considered as the threshold. As soon as block from A is received in the current round, the block by D becomes finalized deterministically as A gives an extra unique endorsement. Similarly, when block from B is received for current round, the block by E becomes deterministically finalized. So the deterministic finality moves from C to E. And so, having more generators help accelerate deterministic finality, albeit with an increase in network traffic.

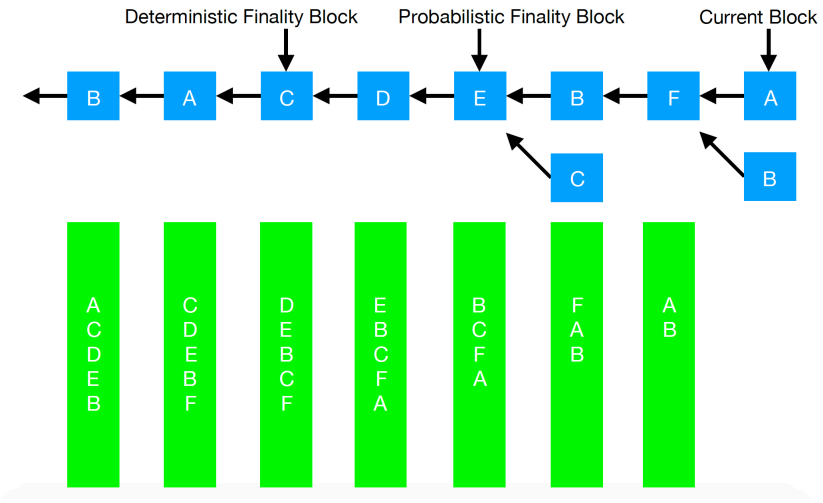

Fig 5. Deterministic Finality

2.2 Sharder Workload Optimization

As mentioned earlier, in our miner-sharder architecture, we recognize that the mining process is CPU intensive while responding to queries of finalized transactions is storage and network intensive workload. The sharding of blocks is further optimized among sharders by using a Consistent Hashing like algorithm and splitting the blocks among the sharders. That is, the 0Chain blockchains can be configured to choose the number of replicators per block and only those many (and sometimes a few more) sharders store the block. Which set of sharders are replicating a given block is completely deterministic from the hash of the block. While blocks are sharded, the mapping information regarding which transaction has made into which block is stored by all the sharders. This way, a client can initiate a query with any sharder and in the best case scenario that sharder has the block information and so will respond back with the entire transaction information and in the worst case scenario, will only be able to tell which block the transaction belongs to so the client can query the details from the right sharder.

2.3 Governance[9]

While a blockchain is decentralized and the core operations of validating transactions and producing the blocks is all automated and achieved via consensus, many blockchains have little or no overarching governance that helps go through any unforeseen issues such as malicious attacks or faulty code requiring major changes and so on. 0Chain has taken a decentralized stake based governance approach. That is, any changes required to the blockchain will be approved or vetoed via a staked voting mechanism which itself is done as a smart contract on the blockchain. In addition, to avoid last minute manipulation, 0Chain uses a novel multi-round exponentially increased limit mechanism to complete the voting process there by making it as robust, fair and transparent as possible. To learn more, check it out here.

2.4 Fast Sync Merkle Patricia Trie[10]

Since 0Chain has a fast finality the state is rapidly changing. So, it becomes critical to be able to sync the state fast and be operational, even with a partial state to contribute to the overall consensus.

---

[9] Ping-Pong Governance: Token Locking for Enabling Blockchain Self-Governance, P. Merrill, T. Austin, J. Rietz, J. Pearce. https://drive.google.com/file/d/1zi-ADhRe6KbYx_HJxMcZNuBm_C-84Pgm/view

[10] Posted on Ethresearch: https://ethresear.ch/t/fast-state-management-using-merkle-patricia-trie/5909

State management is an integral part of the fast finality claim together with the execution time of a smart contract. The production of a block involves executing smart contract logic associated with the transactions in the block that result in mutating the underlying DLT state that is securely verifiable.

So, when comparing blockchains and their finality claims, it's important to keep this full picture in mind, and not in isolation of being able to do a fast Byzantine consensus.

Sync is expensive today

In most blockchains using MPT, when a new node comes online, for it to participate in the blockchain process of generating or validating a block, it first needs to sync up the entire MPT. The MPT can be a very large data structure depending on the number of keys present running into several millions. If the blockchain is popular, such as Ethereum, it can easily go into billions over time. This makes it very time consuming to sync the entire tree.

One of the key observations of transactions on a blockchains is, like many other things, the 80-20 rule. That is, 80% of the blockchain is used by 20% of users and smart contracts (there is a good chance that these numbers are even more skewed). Hence, it doesn't make sense for someone to wait loading the entire state represented by MPT synching up data related to a long and growing tail of users and smart contracts that hardly transact and in the process impact the quality of service for the rest of the users. Ideally, if it's possible to make progress with a partial state that helps the nodes to start earning rewards as soon as possible.

But is it possible to support this given that the MPT updates are expected to be securely verifiable with each block produced on the blockchain? The answer, not surprisingly, turns out to be yes. The mathematical properties of the MPT provides a solution with proof of correctness.

Fast Sync MPT (FS-MPT)

Let's first define some terminology. An application using MPT has an API to store a value with a given key, represented as (k,v) and storing several such values can be represented as (K,V) (the upper case just defines a set while the lower case represents a single element). In order to provide security proof, MPT has to store some intermediate nodes in addition to V and hence collectively represented as (N). Further, the application keys (K) end up representing the paths in the MPT to access the values, while hashes of the nodes, H, are used as keys to the Nodes. Hence, MPT is a mapping (K,V) -> (H,N). The (H,N) pairs are persisted into a key, value store (such as rocksdb used by 0Chain). Note that the pair (H,N) is self validating because H = hash(N).

Say at a given blockchain round, x, the MPT is represented as (Hx, Nx). Subsequently the MPT got modified as the blockchain progressed. For a given block, let's say the state got updated (inserted/deleted) to a small set of keys and values, (Kb, Vb). The new MPT after applying the changes (Kb, Vb), contains a subset of (Hx ,Nx) intact and a new set of hashes and nodes (dHb, dNb) together representing the new state, (Hxb, Nxb). The key observation here is, in order to compute and verify the new state, any partial state (Hxp, Nxp) which is a subset of (Hx, Nx) is sufficient so long as it is possible to compute (dHb, dNb) using this partial state. When the new state is so computed from the partial state, the root hash will be in agreement with that computed with the entire state. This is very important and coupled with the commutative property, it is possible to keep operating on the partial states while simultaneously synching the missing state, (Hm, Nm), block by block or ad-hoc and applying it (as in just storing the new hash, node pairs) to the existing state. The order in which these delta changes are applied doesn't matter.

0Chain blockchain uses the above concept to support the ability to make progress with the blockchain (to generate a block, validate a block, respond to client queries for balances, or other state related queries). While it is good to be able to make progress, eventually the entire state needs to be synched which can happen in parallel. The state sync itself happens via two separate partial sync strategies mentioned below.

1) Block level Partial State Sync - That is, applying transactions of a block to an initial state as mentioned above results in (dHb, dNb) delta state changes. This delta change includes the root key and node and since the root of the state MPT is part of the block hash, it is possible to verify that a given delta state matches with a given block. As a result, when a node is making progress with the blockchain and temporarily unable to validate the state of a block, it can request for the block specific partial state, validate that his partial state is correct and start using it. This helps with any temporary downtime such as a short network outage and allows updating the partial state to full state without having to replay the transactions in all the intermediate blocks to compute the state. This makes the temporary state sync process very fast.

2) Missing State Sync - As mentioned, MPT is a versioned data structure by default and hence it accumulates a lot of (H, N) tuples. These need to be pruned to contain the size of the state db. 0Chain has a state pruning logic that is similar to a garbage collection of Mark and Sweep logic. During this process, it is possible to identify the set of missing state (Hm, Nm). When missing state is detected during the "marking" process, it will sync that state (randomly asking peers for Nm related to a set of hashes, Hm) and abandon the "sweeping". This can continue several times as discovering some missing nodes can lead to discovering additional missing nodes that are children of the original missing nodes. Eventually when the entire state

is synced, the older state that is no longer required is pruned. Note that the “marking” process traverses the tree while the “sweeping” process just traverses the underlying database. As a result the “marking” process can be expensive and 0Chain has a configuration to specify the frequency at which this process runs. However, when the state is missing, this configuration is ignored to ensure the state is synched as fast as possible without waiting for the next pruning cycle. It is likely that the “marking” process can be further optimized by storing the nodes of each level in a separate database or column family to reduce random I/O, something that can be explored in the future.

MPT is used for each smart contract to maintain its own state and the root hashes of each of these are stored as values in the global state that maintains state of the client wallets and smart contracts. This is similar to how Ethereum manages it’s state. Each MPT can be independently synched and it will be possible to prioritize synching some smart contract states over the other based on popularity for example.

The FS-MPT approach of state pruning and synching provides 0Chain with a robust state management. The ability to sync up partial state changes block by block provides the benefit of synching state faster for frequent users and smart contracts (and hence get ready to support them with their subsequent transactions), while eventually syncing the entire state by discovering the missing state in the background. This improves the stability and quality of service of the blockchain.

2.5 Efficient DKG using Blockchain

0Chain blockchain relies on a publicly verifiable random value calculated with each round of block generation. This is achieved by using DKG (distributed key generation) that provides the ability to recover a group signature using t-of-n threshold signature scheme. That is, in order to defend against Byzantine and other faults, only a threshold number of signatures are required to reconstruct the group signature that everyone can verify and agree up on. The group signature is used as a seed to create a random number for each round and the signature for a given round is based on the random number from the previous round. For implementation, we use Herumi’s library for DKG which is based on BN curves.

The literature has various versions of DKG and the version that 0Chain uses is called Joint-Feldman by Pedersen. This protocol starts with n parties but eventually ends up with q parties, referred to as “Qualified” parties. The qualification happens due to the nature of distributed key generation that can result in Byzantine condition. The protocol suggests using complaints and revealing of individual secret shares to narrow down on the qualified set of parties which are all verified to have shared the secrets correctly. The protocol requires broadcasting some messages to the entire network and hence the messages are of the order O(n) and some messages are sent only to individual parties and this is done by everyone resulting in an order of O(n^2). These secret shares that are O(n^2) is where the complaints and revealing happens and that can result in a lot of messages back and forth and also timing of those messages and reliability of coordinating them across all the parties.

0Chain uses a novel approach, by using the blockchain itself, to solve this problem. All the publicly available information is submitted as transactions to the blockchain and each of the n parties need to initially submit one transaction each on the blockchain with the public information of the protocol. After this, the parties will privately send the party specific information among themselves, that are of the order O(n^2). However, there is no complaint or revealing. Instead, each party as part of distributing its secret shares to the others will collect digital signature of acknowledgement that they have received and verified their secret share against the public information already available on the blockchain. At the end of distributing the secrets to all the parties, each party ends up with a) a digital signature verifying a valid distribution or b) no digital signature because the receiving party didn’t not provide one (Byzantine) or was not responsive (temporary network failure). After some amount of trying to distribute the shares, eventually each party will public another transaction on the blockchain with an array of a) digital signatures confirming the receiving of valid secret shares or b) the secret share value and the corresponding party that didn’t provide or receive the secret share. That is, the onus is on each party to submit this information to the blockchain without which the party is automatically disqualified and hence each party has the incentive to submit this transaction. Any party that is not able to gather at least t confirmation signatures will automatically gets disqualified. For anyone who only didn’t receive fewer than the threshold confirmations, the corresponding secret shares are revealed but they can’t be used to reconstruct the secret information of that party (as at least threshold number of secret shares are required to reconstruct the private information). The revealed secret share values can be used by honest parties that genuinely didn’t receive the secret shares at all or were given incorrect values as they are now publicly available on the blockchain and verifiable.

Using the above logic, the qual set can be deterministically computed by each party using the publicly verifiable information available on the blockchain. Hence, this eliminates some of the complexities of identifying the qualified set with the original DKG protocol that only suggests exchanging complaints and revealing/reconstruction messages which can be large and time consuming. In contrast, the blockchain only requires O(n) messages that are all available on the blockchain and further the onus of disclosing is with each party that want to get in.

2.6 Efficient & Secure Light Client Validation

In a decentralized blockchain, the set of active miners and sharders is constantly changing. Hence, any client using a blockchain needs to either interact with a centralized server to submit and query transactions effectively making part of the solution centralized, or need to cope with a dynamic decentralized system. 0Chain provides a mechanism to discover these active miners

using a concept similar to domain name lookup service, DNS. The change of the active set is referred as “view change”. Whenever there is a view change, the miners and sharders can broadcast this information to a set of external servers that no single entity controls. Since the view change is recorded on the blockchain, the external servers can query and verify the authenticity of the changes. Hence, clients can query and get the latest active miners and sharders from any of this static (or less frequently changing) set of decentralized servers.

Another problem for the clients dealing with a decentralized blockchain is in confirming a transaction. A transaction may get finalized but a sharder may respond otherwise knowingly or unknowingly. They can also respond that a transaction is confirmed when it actually is not. Further, since finalization lags behind, and in rare cases individual sharders may falsely finalize in their local view, even if a secure proof is presented, the transaction may not have been confirmed since the finalized block may roll back. To deal with all of these complexities, a simple approach as shown in Fig 6 is to always query a sufficient number of sharders for transaction confirmation. That wastes a lot of resources both to the client and the sharders. Alternatively, a client can try to query a single sharder for as long as possible and only fall back to multiple sharders when necessary. For example, once a transaction is expired and a confirmation is still not received, then the client can fall back to asking multiple sharders for the extra confirmation. Similarly, after establishing a confirmation from a sharder, the client can establish the validity of the subsequent block chain by querying a different sharder for each block and if necessary walking up and down the chain to establish trust. This ensures, on one hand, the client is getting extra validation of confirmation as the blockchain is being built on top of the corresponding block and on the other from different sharders for each block. This combination of traversing the chain with confirmation from a different sharder increases the trust on the outcome.

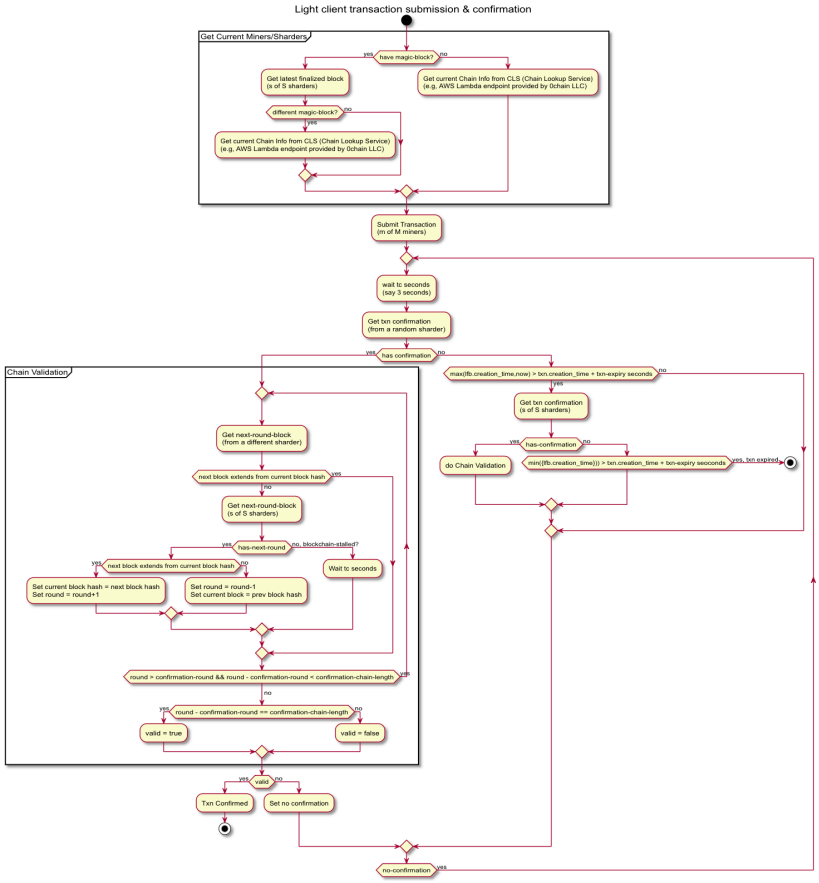

Fig 6. Efficient light client validation

2.7 Client Security[11]

0Chain worked with security researchers and came up with a novel solution based on a security protocol that has survived more than a decade of research that analyze vulnerabilities in the protocols. This protocol, called BLS, provides the ability to split a key into two and store them separately and reconstruct the signature from partial signatures. More details can be found in the

[11] "Splitting and Aggregating Signatures in Cryptocurrency Protocols", S. Selvi, A. Paul, C. Rangan, S. Dirisala, and S. Basu, https://drive.google.com/file/d/1urvaAe0w1DtO9n2JIknK4lCQzGEyE8Pl/view

previously written technical article. As the solution is completely based on software, there is no need for purchasing any expensive hardware. Users can use their existing mobile phones and computers.

0Chain enhances the overall security with various types of wallets and stake pooling mechanisms which are additional protocol layers on top of the above mentioned BLS signature scheme. These details are described below.

End user Wallets

0Chain is providing beautifully designed native mobile apps for wallet transactions and store of value. The wallet app will allow a user to setup the keys with or without splitting. To use key splitting, users will also need to download a desktop app to act as a second device. A very intuitive UI will guide the user to setup the split key between the two devices. Once setup, subsequent transaction submission will require both devices to construct the signature making it secure.
Even with split key, it will be possible to recover the primary key, should the need arise, just like the traditional keys. In addition, it will also be possible to split the keys any number of times to implement advanced security options such as key rotation where a set of keys are periodically discarded. These advanced features are optional and casual users don't have to worry about these features initially but as they become more familiar with using the blockchain and the secure wallet, and their token store of value increases, they can be assured that such advanced options are possible using the signature scheme used by 0Chain.

Service Provider Operation Wallets

All the service providers of the 0Chain blockchain, such as the miners, sharders and validators will need to digitally sign their messages for the rest of the blockchain network to be assured that the message is coming from whom it is supposed to be and is not tampered. The signature scheme used for this is same as that used for the end user wallets. Split key functionality for additional security cannot be used by the service providers for the following reasons
There is no user interaction for supporting the operations of the blockchain
With a fast finality, the number of messages exchanged per second is so large that it would be impractical to add any extra steps that delay the signing process.
As a result of the above operational constraints, 0Chain has decided to provide additional security for the service providers in a different manner. This is done by separating the keys used for staking and reward from the keys to sign the out going messages for supporting the blockchain operations. With this mechanism, the tokens are always controlled by a regular wallet that can use the split key scheme assuring that their stake is never compromised. This technique will also allow the ability to rotate operational keys, something that is common in traditional IT systems having advanced security.

Delegation Pool

The above separation of wallets is further generalized to support the concept of delegation which allows normal users to earn rewards by staking their tokens but without actually owning the blockchain operations as a service provider. A service provider starts a delegation pool by registering an operational wallet and staking tokens using the regular wallet and delegating the operations to the operational wallet. Similarly, any regular user can choose to invest into this delegation pool to increase the overall stake of the service provider which further enhances the chance of the service provider getting picked. The delegation pools are implemented as a smart contract and hence the owner of the delegation pool cannot withdraw tokens deposited by others or avoid paying the rewards as the rewards are automatically distributed to the participants by the smart contract. The reward distribution will be proportional to the stake of each participant.

Multi-Sig Wallet

There is no reason to split the key into only two parts. It can be extended to split it into n parts and even use threshold cryptography where only t-of-n signatures are required. Most existing multi-sig schemes rely on signing with unrelated keys to establish multi-sig. While this is possible, 0Chain's signature scheme allows creating an aggregate signature that can recover the signature of a regular client on the blockchain. This ensures that the balance transfers are done securely with the use of a verifiable signature of a client via multi-sig as if it is directly signed by the client. This type of multi-sig validation is suitable for server side wallets such as in digital exchanges where the transfer transactions are driven by automation.

3 Developer Rewards

In order to incentivize adoption of dStorage, the team is setting aside tokens to incentivize developers to build apps, interfaces, and use 0Chain dStorage. In addition, there will be tokens generated by the network that will go into a pool for developers to continue improvement. Developers can disrupt existing SaaS and consumer applications with dStorage for better performance and security at a lower cost. We have an ongoing Hackathon[12] for useful projects that solves real world problems, and there are prizes ranging from 10,000 to 40,000 tokens, and prizes of up to 100,000 tokens for the overall winner.

See https://0Chain.net/page-hackathon.html for more details.

[12] https://0Chain.net/page-hackathon.html

Here are the following examples of useful project categories.

1. Interface to dStorage for different blockchains such as HyperLedger and Ethereum
2. SDK versions in different languages and different platforms.
3. Interfaces for multi-party solutions, where smart contracts make use of outside data or oracles, and need to prove that the same document is shared with all the validators involved and hence is not tampered in any way.
4. Storage apps - build new dApps using dStorage for different applications such as WeTransfer, Pixieset, DocuSign, etc.
5. Social apps - build new dApps that use dStorage and compete on privacy with YouTube, SnapChat, Instagram, WhatsApp, Netflix, and Spotify.
6. Enterprise apps - build new dApps that use dStorage for healthcare, real-estate, banking, government, and e-commerce to store sensitive customer data and use them for multi-party transactions.
7. Secondary backup interfaces and plug-ins to dStorage for Veem, Commvault, Rubrik, Cohesity, and Nutanix.

Relevant links for development are:

Github: https://github.com/0Chain/gosdk
ZWallet: https://github.com/0Chain/zwalletcli
ZBox: https://github.com/0Chain/zboxcli

Community developed:

a) JS SDK: https://johansten.keybase.pub
b) Web Test for ZWallet, ZBox: http://zcn.sculptex.co.uk/zbox.php?action=info

4 Ambassador Rewards

The 0Chain Ambassador program exists to create an official role for technical and business evangelists that have expertise in the 0Chain protocols and related products such as dStorage, 0Box, 0Wallet, who are interested in helping new community members get involved in the project, and develop businesses around the 0Chain protocols. There is a pool of tokens set aside by the team (discussed in the Token Economics section) which, in future, will be generated by the network will be to fund these technical and business evangelists perpetually. If you are interested in connecting with an Ambassador in your region, find out more about the Ambassador program, reach out to Director of Ecosystem, Derick Fiebiger at operations@0chain.net.

There are several use cases that need to be promoted via Hackathon events as outlined in the Developer Rewards section. The idea is to create a reward pool for Ambassadors to organize local events, educate entrepreneurs about these new use cases, and help develop businesses.

5 Miner Rewards[13]

The miners have an incentive to mine on our network, because they have additional sources of revenue. They can still earn tokens based on traditional mining and transaction fees, but can earn tokens in other ways:

- storing blocks as a sharder,
- providing storage service as a blobber,
- “free” ZCN immediately when you stake, and
- delegation fees

When miners stake their tokens, the network generates interest tokens for them. They earn as a miner and a sharder, and as storage grows on the network, they will earn more as a storage provider. The infrastructure requirements for a miner, sharder, or blobber is a simple server that people can either buy locally or put it together themselves. The parts are listed in Fig 7 for a 24 TB storage server starter set which can mine and shard as well. If you already have spare parts that you can put together, then your cost is nil except when scaling storage as they get consumed.

Earn as a Miner and Sharder

The earnings potential for a scalable platform for a miner and sharder, if you were to stake 200,000 ZCN as an example, is shown in Fig 8. The earnings are twice as many if you were to just lock your tokens, and three times more if you are able to participate as a miner and sharder. The math is simple.

Reward tokens per Miner/Sharder = (Tokens for miners and sharders) / (number of miners and sharders)

If the number of tokens is 1.4M, and there are 130 total miners and sharders, then you will earn about 10,000 ZCN each. Since the number of tokens have been increased for the first year to 6M with team tokens, each participant will earn 46,000 ZCN each.

[13] Miner reward document. https://drive.google.com/file/d/1YMU5AQmuW8bDiTesOKKCZ7hqctlXb2g1/view

Component	SKU	Description	Unit Price	Qty	Total Cost
Case	CA432	2U - 12 DRIVE BAYS - CASE SM 2U Chassis CSE-826BE1C-R920LPB	$1,100.00	1	$1,100.00
Processor	CP495	CPU Intel Xeon Silver 4110 8C/16T 2.1G 11M 9.6GT UPI*	$550.00	1	$550.00
Motherboard	MB618	MB SM X11DPL-i Intel(R) Xeon(R) Scalable Processors., Dual Socket P (LGA 3647)	$500.00	1	$500.00
Memory	ME571	MEM DDR4 2400 16GB ECC REG	$165.00	2	$330.00
OS Harddisk	HD653	SSD SM IND SATA3 DOM ML 3IE3 V2 128GB iSLC LP, Pin8 VCC	$255.00	1	$255.00
HDD	HD634	HD Seagate ST12000NM0007 6TB 7200 RPM SATA 6Gb/s 256MB Enterprise Hard Drive	$170.00	4	$680.00
Controller	CO LSI	LSI00346 SAS 9300-4i Host Bus Adapter	$215.00	1	$215.00
			Total Cost (Excluding Tax & Shipping)		**$3,630.00**
Monthly Colocation cost	-	*https://www.colocationamerica.com/colocation/2u-colocation.htm*	*$99.00*	*1*	*$99/mo*

Fig 7: Miner rig template

Miner + Sharder Reward (assuming Bob can split his 200k ZCN and still participate as miner and sharder)

Bob's Initial ZCN	Estimated Miner Rewards per Year	Estimated Sharder Rewards per Year	Estimated Total Miner + Sharder Rewards per Year	Bob's Stake Interest per Year	Total Rewards	**Bob's Total ZCN after 1 Year (Initial ZCN + Miner + Sharder + Interest Rewards)**
200,000 ZCN	46,000 ZCN	46,000 ZCN	92,000 ZCN	20,000 ZCN	112,000 ZCN	**312,000** ZCN 56%/Year

Fig 8: Miner and Sharder rewards potential

The rewards to miners and sharders are based on the following protocol.

1) Staking results in 3 payments :
   a. interest - the rate changes based on the rate at each given view change.
   b. reward - we are striving to keep this same for miners and sharders. We will have an equation below that can be used to either keep it same or change the sharder stake a multiple of miners.
   c. transaction fee - as this is very dynamic with each transaction and block, it will be hard to predict this component. So, we will likely just indicate this extra potential without giving any value.

2) Below is the formula to compute regarding the reward per round which can then be extrapolated to show the amount for N rounds. If each view change happens after Vn rounds, then that can be used as the metric N and project the returns per view change.

3) Even if there is no net change in miners/sharders, we will do the View Change protocol as we are assuming certain payments to happen with each view change.

Equation for reward per round per party:

a. Let the number of active miners be M and sharders be S.
b. Let the reward per miner per round is x.
c. Let the reward per sharder per round is a multiple, k, of the miner's reward. So, the reward is k*x.
d. Let the available reward per round be R.

After N rounds, the total reward given out by the network = N * R.

The total reward received by all the miners and sharders = N* (x*M + k*x*S)

Equating both sides, x = R / (M + k*S).

Note that the above reward structure has no indication of number of generators or replication factor. But starting out, M = 100, S = 30, number of generators = 10%, replication factor = 6, and k = 1. We can tweak these parameters based on the ongoing cost through governance protocol.

Cost Analysis

The mining network will be effective only when the rewards exceed the costs. For the below discussion we will ignore the interest and transaction fee based rewards for simplicity without loss of generality.

*) Let the cost of generating a block be Cg and the number of generators is g.
*) Let the cost of verifying a block be Cv.
*) Let the number of blocks verified on average in a round be v.
*) Let the cost of storing a block be Cs and the number of replicators is r.
*) Let the cost of supporting queries related to transaction confirmations and block for a given round be Q (this is a cost that has no specific bound).

The cost of operating N rounds is $N * (g* Cg + (M-1)*v*Cv + r * Cs + Q)$

This reward should be greater than the cost, $N * R >= N * (g* Cg + (M-1)*v*Cv + r * Cs + Q)$

This implies that $R >= (g * Cg + (M-1)*v*Cv + r*Cs + Q)$

As the above formula indicates, the reward per block should be more than the cost that does consider the number of generators and sharders. So, while the reward per party does not directly have these parameters, the reward given indirectly has a lower bound based on these blockchain parameters.

Earn as a Blobber or MSP (Managed Service/Storage Provider)

While the earnings potential as a miner is limited because of the finite reward pool, the earnings potential of a storage provider is unlimited. In addition, the earnings can be in fiat currency as detailed in Section 1.8. And so, we encourage miners to start with an infrastructure that has the potential to provide them multiple streams of revenue, and add-on alternatives. Fig 9 shows the earnings potential of a blobber as the storage demand grows over the years with unlimited potential, as there is no barrier to entry.

Blobber Earnings (assuming Bob can stake equivalent ZCNs as a blobber)

Bob's Initial ZCN	Estimated Storage Earnings average over Year	Estimated Egress Earnings average over Year	Estimated Cost per Year	Total Invested	Bob's Total Profit first Year & ROI	
20 TB stored 40 TB egress	$1,200	$1,800	$1,200-2,400 shared	$3,600	$ 0.6-1.8k + $3k mining	< 1 year
200 TB stored 300 TB egress	$12,000	$18,000	$3,000 shared	$10,000	$ 30k	< 4 mths
2 PB stored 6 PB egress	$120,000	$180,000	$15,000 one cabinet	$100,000	$ 288k	~ 4 mths

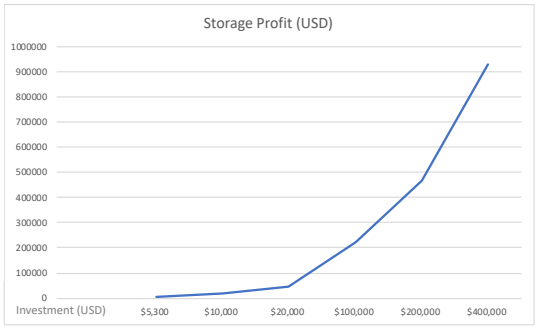

Fig 9: Blobber earnings potential

0Chain is building a MSP program and community so that the entire SMB/SME market can be served with dStorage by their community. It is a compelling cost/performance alternative to traditional cloud. Today, most SMB/SMEs use DropBox, Box, OneDrive, AWS, Azure, and Google. The MSPs can continue to make the same margin by selling dStorage and 0Box to SMB/SMEs, but can now have multiple streams of revenue as a blobber, as well as a miner and sharder. The MSPs will set in their configuration a portion of storage for their rigs, and select other MSPs from the blobber pool. As an example, if they decide on a 10/20 erasure coding, they can send data to their three rigs, the other seven to MSPs they trust and have relationships with, and the rest to anyone globally. The MSP orchestration and configuration process will make this setup simple and a clickable process, and a dashboard will show MSPs how much revenue they are earning by managing their customer's storage, and how much they are making on their equipment. Fig 9b shows how MSPs have unlimited earning potential with the growth of dStorage.

We have following payment protocol for the blobbers and clients.

Allocation

A user asks for a storage allocation of a certain capacity offering a range of min and max cost per unit storage/unit time. The smart contract identifies all the blobbers who are offering storage within the cost range provided by the user and randomly picks the required number for the chosen erasure coding. This ensures that the blobbers are randomly selected but at the same time the cost of the overall storage is within the range desired by the user. Governance can dictate absolute min and max for these ranges if desired.

Starvation Attack

When an allocation is created, it establishes write and challenge pools which is a striped by the client-blobber relationship. A minimum of 10% of the cost needs to be immediately paid by the client for successful allocation of the storage. If at the end the storage subscription time the client hasn't actually spent 10% of the storage, the initial paid amount ensures that the blobber is paid at least 10%. This prevents *starvation attack* by the clients. The initial tokens are placed into a write pool. As writes happen, these get moved into the corresponding challenge pool. To storage additional content, user needs to keep funding the write pools.

Staking Pool

When the allocation is created, the blobbers stake 100% of the storage cost for which they receive interest but the locked stake will be punished if they fail proof of storage challenges.

Amount Per Unit Storage Unit Time

Both the storage cost and the stake are converted into unit amounts by the size and time. This helps interpolate the rewards/ punishment by storage size and time left before expiration. For example, the 100% Stake from a blobber for 100 GB and 100 days becomes Stake/10^4 unit price per GB/ per day (The actual units used is subject to change based on the final implementation of the smart contract). Similarly, the reward of a stored content is Reward * Size * Duration where Reward is the unit price of storing the content for a given blobber.

Storage reward set aside for challenges

When a user writes a new file and the blobber redeems the write markers, the funds move from the Write Pool to the Challenge Pool. Similarly, when a user deletes some files, the funds may move from the challenge pool back to the write pool. However, this is done only after adjusting for some reward towards the deleted content that was stored up to that point. This is done by adjusting the challenge pool by assuming a virtual challenge that has successfully happened and rewarding appropriately.

Challenge rewards and slashing

If a blobber passes a challenge at time T1 and the next challenge at time T2 with an eventual expiration at Tstop, the amount paid will be challengePool * (T2-T1)/(Tstop-T1) which is a linear interpolation of a fraction of the challenge pool reward set aside for the entire duration from T1 to the expiration time paid in proportion to the delta time (T2-T1).

Similarly, if a blobber passes a challenge at time T1, and fails a challenge at time T2, then they would be penalized Stake * (T2-T1)/(Tstop-T1).

In these equations, the unit price by storage size does not come into picture. Any challenge failure is assumed to be a failure for the entire allocation. So, while costs are calculated based on the content size and set-aside for challenges, the reward/punishment of challenges are purely time based.

Expiration

At the time of expiration, if the blobber hasn't received 10% of the original storage cost, the write and challenge pools guarantee that the blobber is paid from the already locked tokens at the beginning of the storage allocation. Beyond that, any unused amount in the write pool goes back to the user and any amount in the challenge pool goes back to the blobber.

Extension

Users can extend their storage at any time before the expiration. When they extend the storage, the unit prices get readjusted by weighted average of the time left and the original cost with the new duration and the new prices. The cost of extension is determined by the current cost of storage and computing the cost from the original expiration time to the new expiration time. However, the rewards and punishments are based on the weighted average of the remaining values and the new values along with the time remaining to the new expiration.

6 Enterprise benefits

Enterprises benefit from dStorage in three ways:
- 50% lower cost and easier scale out their storage needs
- Higher performance (10x speed) and availability (14 nines) of their workload if they retool their client
- Data Breach protection through auditability on a public blockchain
- Supplement their IT budget by lending out their storage on the blobber pool

Likes the MSPs, we envision that enterprises will form a community that will actively participate to form a consortium of storage providers among each other to benefit from a lower cost, higher performance storage solution.

7 GTM strategy

The MSPs market size is 20,000 in the U.S. and about three times that much globally[14]. The MSPs serve the SMB/SME market and according to a World Bank Group study suggests there are between *365-445 million* micro, small and medium enterprises (MSMEs) in emerging markets: 25-30 million are formal SMEs; 55-70 million are formal micro enterprises; and 285-345 million are informal enterprises (Small and Medium Enterprises (SMEs)), and they all have a need for protecting their data.

Our GTM model is to target MSPs, VARs (Value Added Resellers), and VADs (Value Added Distributors) and start serving the SMB/SME community with 0Box and dStorage, and displace traditional alternatives such as DropBox and AWS. We will also pursue select relationships with Enterprises to build the Enterprise community model.

8 ZCN valuation[15]

The notion that ZCN, a cryptocurrency, is tied to data and locked tokens is a very novel concept.

ZCN as a data asset

For blobbers to participate on the network, they need to stake the sell price of their storage. For example, if they are selling storage at $0.1/GB for a year, and they are putting 10TB out of their 50TB rig capacity, then they need to stake $1000 equivalent number of tokens on the network. Similarly, a client that will use 10 TB of storage for a year will lock as much, and that amount will get paid to blobbers over the year. And so the average equivalent value of locked/staked tokens on the network for both parties is $1500 on average. And this constitutes the base value of the network because of such demand. Fig 10 shows the current data, cloud market, and penetration level expected for dStorage, and the corresponding value of data stored on the network. The upshot is that if we achieve a 10% penetration of today's cloud market, then there would be a $20 billion value directly based on the data on our network.

Storage Based Token Value								
Total Data (ZettaBytes, ZB)			35	48	67	92	127	175
Total Cloud Storage Market size (USD, Billions)			45	56	70	90	110	135
% penetration			0.10	1.00	3.00	6.00	9.00	11.25
ZCN equivalent staked/locked (USD, Billions)			**$0.090**	**$1.120**	**$4.200**	**$10.800**	**$19.800**	**$30.375**

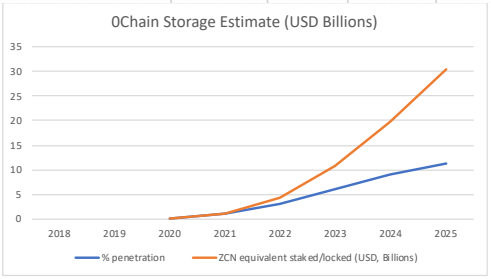

Fig 10: Dollar value of data stored on 0Chain based on estimated penetration

ZCN as a blockchain CD

We are familiar with bank CDs, where if you lock a portion of money for a specific time period, you receive interest. The banks then take your money and re-lend it at a higher interest rate to businesses and home owners, thereby making a profit. Our network is somewhat similar in the user getting the "interest" tokens, but the other mechanics are quite different. When you lock your tokens on the network, it generates these tokens, similar to Bitcoin or Ethereum, but the latter coins give all their minted tokens to the miners. In our network, the mined tokens primarily go to the token holders, who desire to "hodl", lock, and earn interest. So the inflation in our economy is primarily driven by the desire of token holders to lock tokens.

[14] https://www.channele2e.com/faq/msp-market-size-forecast/

[15] The Equilibrium Price of 0Chain Token, Andrea Buraschi and Sebastiaan Vervest, Imperial College London. https://drive.google.com/file/d/1emqKT-Z2SyZXH2MHmCrr8suJfoVhffbc/view

The value of the ZCN network is directly related to the number of people who will use ZCN as a currency and receive interest tokens. The more people we have on our network to lock tokens for long term, the higher the value of ZCN based on the closed form supply/demand equilibrium equations on the paper by Professor Buraschi of Imperial College, London.

Receive Immediate Interest

Unlike banks, when you lock tokens, you receive interest tokens immediately. This helps hodlers to use interest tokens for services such as storage, transactions, and future services on the 0Chain platform.

ZCN valuation = interest + data

So the valuation of ZCN is a combination of data on the network and hodlers' desire to earn interest through locked tokens. The locked and staked tokens can be visualized transparently on the network to estimate the base value of the network at any given point of time. Unlike other cryptocurrency, the value of ZCN can be deterministic and less speculative, and unlike most stable coins, it will be a transparent currency.

9 Token Economics

The distribution of tokens (by the 0Chain Foundation) is outlined in Fig 11. There is a total of 200M tokens pre-mined. The distribution is as follows: 85M for the team, 15M of "reward" tokens set aside by the team for miners, sharders, community (developers, ambassadors, MSPs) to build the dStorage ecosystem for the first 4 years after main net is launched, 20M for seed, 40M for private pre-sale investors, 40M reserved for team with equal vesting in Jan 2020 and Jan 2022 if the market value is greater than $10. Most of the team tokens will be locked for several years to preserve the integrity of the network as shown on the table in Fig 11. The network will generate the reward tokens after 4 years to continue funding the participants on the network. The reward and interest tokens are expected to constitute an average inflation rate of 3% for the network. The table in Fig 11 elucidates potential tokens in circulation, inflation, and interest rate, and also shows the reserve token schedule. There are a couple of interesting observations.

All numbers in millions of tokens								
	2018	**2019**	**2020**	**2021**	**2022**	**2023**	**2024**	**2025**
Vested Team - Expect to be Locked for mainnet operation*	18.75	18.75	18.75	18.75				
Vested Team - Expect to be Unlocked and in circulation	2.5	2.5	2.5	2.5				
Vested Seed	5	5	5	5				
PreSale Holders	40							
Reserve**			20		20			
Team Give Away Rewards***								
Miner			3	1.5	0.75	0.75		
Sharder			3	1.5	0.75	0.75		
Community****			1.5	0.75	0.375	0.375		
Interest rate			10%	7.50%	5%	5%	5%	5%
Network generated Rewards							1.875	1.875
Expect Network generated Interest tokens *****			7.31	6.88	5.31	5.49	5.62	5.81
Interest tokens/Reward tokens ratio			0.98	1.84	2.83	2.93	3.00	3.10
Total Circulation	**47.50**	**55.00**	**77.31**	**95.45**	**102.63**	**109.99**	**117.49**	**125.18**
Total Tokens (Locked + Circulation)	66.25	92.50	153.56	190.45	217.63	224.99	232.49	240.18
Expected Inflation rate (percentage)		39.6	66.0	24.0	14.3	3.4	3.3	3.3
Expected Inflation rate (percentage) based on tokens in circulation		**15.8**	**40.6**	**23.5**	**7.5**	**7.2**	**6.8**	**6.5**
* Locked tokens to preserve network integrity								
** Reserved for team. Unlocked over 2 years, if token price > $10								
*** Assume live mainnet in 2020 for simplicity								
**** Developers, Ambassadors, MSPs								
***** Assume 50% of tokens locked								

Fig 11: Token distribution and inflation economics

ZCN Inflation driven Interest demand from Token Holders

Unlike Bitcoin and Ethereum, where all the minted network tokens go toward the miners, the inflation of ZCN is driven by how many of the token holders desire to lock their tokens. And this is independent of storage demand. This concept is depicted in Fig 12.

Storage demand counters selling pressure and provides liquidity

Because the ZCN needs to be staked on the network, we expect constant liquidity as new storage gets signed up on the network. We also expect the storage demand to stabilize selling pressure as there is constant buying and locking on the network.

The value of ZCN is based on the number of users lock tokens for interest, and amount of data stored on the network.

12a: Bitcoin (old coin) vs ZCN Inflation economics

Fig 12b: Expected ZCN Inflation

10 Products

0Box, a private cloud

0Box is a private cloud. No logins and no emails are needed, especially if the user pays the premium version with ZCN tokens. The first version of 0Box, launched with mainnet, will have the fiat feature so that the regular user will have a user experience as DropBox or Box, and can purchase the privacy cloud service with their credit card. In this scenario, 0Box will handle all protocols and payments on behalf of the user, but the user has absolute control and access to their data. They can see challenges to their data regularly, giving them the satisfaction that the data is stored safely and not breached by anyone, and they can see which blobbers are storing the pieces of their data and their url as shown in Fig 13. In addition, the consumer will have a faster user experience than other competitors such as DropBox. Currently, 0Box is in beta; check it out at: https://testflight.apple.com/join/akv3fqGz

0Wallet, a serverless 2FA wallet for ZCN

0Wallet protects ZCN asset with innovative serverless 2FA using mobile and laptop. If either device gets compromised, your assets are still safe. The setup is simple as shown in Fig 14. The user downloads an authenticator app on the laptop and uses it to accept a transaction initiated by the phone. It is a point and click operation to have the mobile device talk to the computer as long as they are on the same network. If one of the devices is compromised your asset is still safe, and you can regenerate a new set of keys with your original passphrase (private key). Check out the iOS app on the App Store and Android app on the Play store. Download the Windows and Mac authenticator from our website https://0chain.net/page-wallet.html.

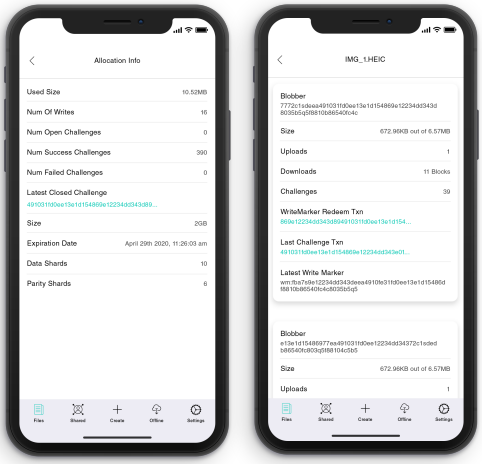

Fig 13: 0Box, a private cloud - private, anonymous, transparent

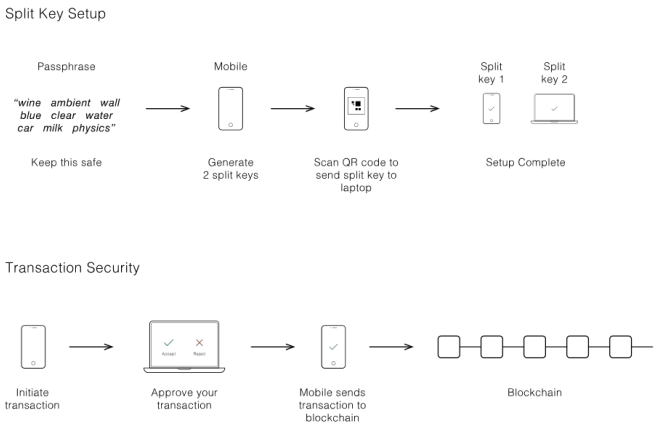

Fig 14: The setup process for 0Wallet, a serverless 2FA wallet for ZCN

11 Roadmap

The current development has been ongoing since July 2017 on the design of several protocols and products. The table below shows the development and product release roadmap for 2019 leading up to the launch of main net, following which we expect to work on other aspects of the dStorage.

Q1	Q2	Q3	Q4 and beyond
Alpha Network Release	SDK Release	BetaNet & MainNet	0Box with ZCN iOS & Android
0ChainNet Consensus	Go SDK and JS SDK	Hackathon	0Box Android & Windows Sync
0ChainNet Storage protocol	CLI for Storage	Bugathon	dStorage™ for MSPs, SMEs
0ChainNet Explorer	CLI for Wallet	Launch 0Box iOS & Mac Sync	S3 Interface
		Launch 0Wallet iOS & Android with Mac/Windows Authenticator	MSP Orchestration