EtherLite

Whitepaper 2.0.1 | https://etherlite.org/

Index

1. Introduction

Bitcoin, which was initially launched in 2009, gave rise to blockchain technology and the notion of cryptocurrencies. It is still the most well-known blockchain project for storing the most significant monetary value. However, it is evident that it's only the beginning, and it is possible to grow this space.

As the blockchain technology and its demand grew, problems also grew in the system such as expensive and glacially slow transactions. It has become increasingly obvious that the Proof of Work consensus models are not ecologically viable as a long term solution. Bitcoin’s model currently uses a minimum of 2.55 gigawatts of electricity, and will potentially consume over 7.67 gigawatts in the near future, which is almost as much as the total energy consumption of Austria which uses 8.2 gigawatts.

2. The Rise of Ethereum

Ethereum developed the blockchain concept, moving from the UTXO to the ac- count architecture and adding a Turing-complete virtual machine that facilitates the formation of a domain-specific language called Solidity (and Serpent), as well as the entire ecosystem that surrounds it.

2.1. Success

Ethereum is by far the largest and one of the most well-established, open-ended decentralized software platform. Therefore it allows users to not only deploy smart contracts, build various decentralized applications (DApps) but also run without any downtime, fraud, control or interference from any related or unrelated third party. The simplicity and minimum effort in developing smart- contracts is the main factors behind Ethereum's success.

2.2. Advantages

2.2.1. Decentralization

Ethereum is only second to Bitcoin in being most accepted cryptocurrency in the world. Now even with influencers play a massive role in how the cryptocurrency behaves in the market, ultimately there is no central authority with ultimate control over the project which allows the trust within the community to persist in the platform.

2.2.2 Modularity

Ethereum have modular architecture. This allows customization of blockchain the way every digital asset demands. The modularity makes ethereum exceptionally flexible. That’s why multiple DApps (Decentralized Apps) are built and continue to being built in ethereum ecosystem. The mass adoption and spread of community across the world is the reason behind increasing gas prices.

2.2.3 Governance

On-chain governance is nothing but a system for controlling and executing modifications to the cryptocurrency's blockchains. In these types of governance, rules for managing/executing modifications are built-in the blockchain's protocol. Developers propose these modifications through the code updates and each node then vote on whether the modification should be accepted or rejected.

DApps built on Ethereum ecosystem have their own governance mechanism and governance coin. 'Users may vote on the growth of decentralised protocols using Governance Tokens, and they provide DeFi creators new methods to attract assets into their platforms. (One of such example is UNI token of Uniswap platform which have helped platform grow manifold and have given worldwide recognition since launch and airdrop)

Interoperability

Money Logos is a protocol on the Ethereum Community which explain that when you build an app on the Ethereum network, you instantly connect it to hundreds of other protocols that already exist in the ecosystem.

3. Key Terms

The most important problem is the consensus algorithm, which can be thought as a method of defining the valid history.

To understand the problems, we need to grasp some technical terms of Blockchain.

3.1. Blockchain

Blockchain is a public digital ledger that keeps track of all prior transactions in chronological order. The blockchain is called so because it is a chain of blocks. A blockchain is a hash-linked data structure. Every blockchain technology is composed of 3 primary parts:

- peer-to-peer networking,
- the consensus algorithm (software rules to agree on the state of shared data)
- and the blockchain.

3.1.1 Consensus algorithm

Blockchain can be seen as a ledger of blocks subdivided in atomic transactions. The way new block are constructed, which transaction to include there and which not to include is called the consensus algorithm. There are several different consensus algorithms yielding different properties of the network.

Let's have a look at them.

3.1.2 Proof of work

As discussed earlier, POW is the oldest algorithm of consensus introduced by bitcoin. The new block is constructed by solving a mathematical puzzle which asserts that some amount of computational work has been done. The one who solves the puzzle first is the creator of new block. This makes the owner of the most computational power to be the one who decides which transactions are considered as valid and which or not. This algorithm of consensus is the most simple and well-studied, however it has a lot of drawbacks, for example its slowness and power consumption.

Over the last few years, blockchain has become more complex, new point of views have emerged.

3.1.3 Proof of authority

To describe the next consensus algorithm, we need to draw distinction between public and private blockchains. The private are Proof of authority consensus algorithm means there are certain set of actors, who have been given the ability to construct new blocks. Thus only they can do this and no one else.

3.1.4 Proof of stake

Proof of Stake (PoS) concept states that a person can mine or validate a block of transactions according to how many coins he or she holds. This means that the more native coins owned by a miner, the more mining power he or she has. Proof of Stake (POS) is considered to be less hazardous in terms of the possibility for miners to attack the network since it arranges rewards in such a manner that an attack is less profitable to the miner.

Proof of stake (PoS) attempts to overcome this problem by assigning mining power to a miner's percentage of native currencies. Instead than using energy to solve PoW problems, a PoS miner is restricted to mining a fraction of transactions equal to their ownership share.

Ethereum 2.0 is using Proof of Stake as a consensus protocol. Although it is claiming to reduce gas prices, that is long term process and may take several years

PBFT

Byzantine Fault Tolerance (BFT) is the feature of a distributed network to reach consensus even when some of the nodes in the network fail to respond or respond with incorrect information. The goal of a BFT mechanism is to protect against malfunctions by utilising joint decision (both right and faulty nodes) to reduce the effect of faulty nodes. Byzantine Generals' Problem is the source of BFT.

Byzantine fault tolerance could be accomplished if the network's correctly functioning nodes concur on their values. There can be a default vote value given to missing messages i.e., we can assume that the message from a particular node is ‘faulty’ if the message is not received within a certain time limit. If the majority of nodes answer with the correct value, we may also provide a default response.

When a transaction is made in Hyperledger, the transaction details are sent to the

nodes in the network. There might some nodes that will approve the transactions and some nodes that won't. The majority (or a minimum specific number) of nodes have to approve the transaction for the transaction to go through.

1.3. Nodes

Nodes are the network's distributed computers, each of which has a copy of the complete blockchain. Copies of the blockchain and access to it are disseminated as new users join the blockchain network. The data is replicated, synched and shared across all the nodes across multiple networks. The data is not controlled by an individual node or network.

3.2. Smart Contract

Once signed, a smart contract is a digital agreement stored on the blockchain that cannot change. It specifies specific logic processes that must be completed in order to complete activities like depositing money or data.

For example: Conditions of releasing money to a third party delivery team: A sender wants to send goods to a certain receiver using third party services, but wants to pay money for delivery only after the delivery is successful. Then the smart contract structure could be as follows; “The sender pays the shipment money on the day of loading the goods. The smart contract will hold payment in escrow till such a time that the recipient confirms the receipt of goods to the sender.” Only then will the smart contract release the payment and automatically transfer the money to the delivery team.

4. What Caused the Problem

The most important problem is the consensus algorithm, that can be thought as a method of defining a valid history. Blockchain was introduced to the world with proof of work. There are two primary disadvantages to Proof of Work systems. The first is that they waste massive amounts of energy and therefore, it is not scalable. Ethereum 1.0 is facing a similar issue. Undoubtedly, It is user friendly and open, hence lot of DApps are being built on it and that’s why gas prices are unfathomably high.

3.3. Price

At the time of writing the typical transaction fee for a token transfer requires 40,000 - 200,000 gas units, with gas price of 150 Gwei and Ether price of $1500 yields price of the transfer as high as about $10, let alone more complex transactions, fees for which are would cost even hundreds of dollars. This effectively makes Ethereum network unusable.

There were several attempts to create alternatives, the Binance smart chain (bnsc) is one such example.

3.4. Scalability

The primary problem with Ethereum too, is scalability. Transactions are still very slow. Since Ethereum's public blockchain can only handle about 15-20 transactions per second (TPS), compared to 45,000 for Visa.

Enterprises want a high transaction throughput, which Ethereum's public blockchain currently cannot provide.

5. Introduction to Etherlite

EtherLite is a blockchain platform designed to increase productivity by allowing users to create and execute various Business Logics such as Smart Contracts and DApps quickly, safely, and cost-effectively. It works on the PoS consensus method and is fully compatible with Ethereum's tools and Web3 technology stacks.

While Ethereum 2.0 which is also known as Serenity, will aim to bring such a Proof of Stake consensus that will control Sybil attacks and many such innovations to the entire ecosystem. However, the completion date of the project is unknown and only speculations are keeping the heat alive.

The Etherlite POS protocol provides a readily scalable solution for Ethereum 1.0, creating the perfect mechanism for delegated staking, higher transaction speeds, and very efficient transactional costs. While on an Etherlite enabled sidechain, users can stake tokens (>minimum required for staking) in order to become eligible to be shortlisted as validators on the network. Delegators can back these users by staking their tokens, thus providing a voting like scenario and contributing to the user's pool.

Etherlite is currently supported by OpenEthereum and POS Contracts. Etherlite runs on a fully compatible EVM-based chain, allowing mainnet interoperability while also providing greater efficiency, lower fees, configurability, and other benefits relative to the current EVM consensus implementations. Etherlite will connect with ethereum mainnet and also compatible with other EVM based blockchains. This will enable interoperability between all the EVM based blockchains. Etherlite is using POSDAO as consensus protocol.

5.1 Features of EtherLite

- **Web3 compatibility**

EtherLite is fully compatible with Ethereum's web3.js interface API. It simply means that the website or service interacts with the Ethereum network. Web 2.0 focuses on users to generate value from which the owner or host may profit. Unfortunately, users have little to no control over their data in most situations. Furthermore, consumers have no way of knowing whether or not the material they appreciate will be available in the future. When material is no longer needed or poses a financial risk, the website or service's provider has the authority to remove it from their platform, leaving you out in the cold.

In today's world, most service providers, such as social media and OTT platforms, control their users' data under their terms of service. Web3 aims to empower people while also recapturing the value they generate. DApps, or decentralised apps, are developed on decentralised peer-to-peer networks like Ethereum and IPFS. Instead of being run by some company, these networks are built, operated, and maintained by their users. They're self- organizing and lack a central point of failure.

- **High throughput**

As explained earlier POW requires lots of resources and energy and Proof of authority is for private blockchains. Therefore, EtherLite's Proof Of Stake protocol makes it a

highly scaleable Solution. Etherlite also provides really fast block times ~3s, thanks to its POS system.

- **Fast transaction finality using a PoS consensus engine**
- **Smart Contract and Tooling cost much less.**

Etherlite has the ability to use existing Ethereum smart contract and tooling. Developers can port their existing Ethereum-based DApps in a matter of minutes, substantially upgrading the performance and lowering their costs.

- **Compatibility**

Etherlite is also compatible with existing tools like Clients, Metamask, Remix, and Truffle, etc. which are used in Ethereum Blockchain.

- **EVM Compatible**

EVM stands for Ethereum Virtual Machine. Virtual machines will essentially create a level of abstraction between the executing code and the executing machine. This layer will improve the portability of software, and will also ensure that applications are separated from each other, and separated from their host.

5.2 Advantages of Etherlite

- **Fast** - At EtherLite, transactions become instantly irreversible, there is no need to wait minutes to hours for confirmations. EtherLite runs on a POS consensus and is extremely scalable, it provides fast block times ~3s and 10,000+ TPS.
- **Compatible -** EtherLite is fully compatible with EVM. Meaning that any existing Ethereum smart contract and tooling can be used at EtherLite. Developers can port their existing Ethereum-based DApps in a matter of minutes, substantially upgrading the performance and lowering the costs. Any developer familiar with Ethereum can use EtherLite without (almost any) additional skills required.
- **Modular -** EtherLite's modular internal architecture is flexible, easily customizable and less error prone. Further development, additional feature implementation and bug fixing becomes much easier making the framework developer friendly.
- **Security -** EtherLite runs on a trust-less Proof-of-stake network where validators are themselves stakers.
- **Scalable -** EtherLite is having capability to process thousands of transactions per seconds and scale to thousands of nodes.
- **Interoperability -** Using Etherlite you can transfer or trade assets or coins from one chain to another.
- **Bridge -** Native support for cross-chain communication among the two blockchains. The communication protocol should be bi-directional,

decentralized, and trustless. It will concentrate on moving digital assets between EtherLite to another EVM supported blockchain

5.3 OpenEthereum

OpenEthereum is the fastest, lightest, and most secure Ethereum client and is coded on Rust. It is licensed under the GPLv3 and therefore can be used across the Ethereum network. OpenEthereum provides the core infrastructure essential for speedy and reliable services.

- Clean, modular codebase for easy customization
- Advanced CLI-based client
- Minimal memory and storage footprint
- Synchronize in hours, not days with Warp Sync
- Modular for light integration into your service or product

6. What is ETL?

If you consider Etherlite as a country then ETL is it's native currency. Everything that executes in Etherlite, ETL is used as a currency for that. It is used to sustain the network, run the governance mechanism and to pay the network fees. By holding ETL, participants are able to access EtherLite's core functionalities.

6.1 Securing the network

EtherLite runs on a Proof-of-Stake mechanism which requires ETL to sustain the network. Validator nodes are required to stake a minimum of 100,000 ETL and in return validators will receive rewards and fees for their service. If validator does a malicious activity, then his staked ETL are basically confiscated

6.2 Governance

ETL is also the 'Governance Tokens' that shall enable users to not only vote on the future of decentralized protocols, but also present fresh ways to entice assets onto protocol. Thus making EtherLite a truly decentralized protocol.

6.3 Payments

The ETL token is ideal for sending and receiving payments thanks to the EtherLite network's high-throughput, fast finality, and low fees. On EtherLite, money transfers take place within seconds and it cost much less.

6.4 Network fees

ETL is used to pay for general network fees such as transaction fees to access and support network operations.

6.5 **Reward Distribution**

Along with network fees, rewards in the EtherLite ecosystem are distributed using

ETLs. Rewards like block rewards are explained later in this Whitepaper.

7. Introduction to POSDAO

Proof Of Stake Decentralized Autonomous Organization or POSDAO, is a consensus algorithm designed in such a way, so as to provide a fair, and energy efficient solution for public chains. The algorithm works as a cohesive intelligent contract written in Solidity. It is implemented with a general purpose BFT consensus protocol. A customizable incentive system incentivizes validators to act in the best interests of a network. The algorithm provides Sybil attack control mechanism by managing a set of validators, distributing rewards, and reporting and penalising malicious validators.

How it works?

Participants in POS protocols stake EtherLite Coins (ETLs), to protect the network and achieve agreement regarding blockchain transactions. ETL is used as DPOS staking coins, coin holders can participate as validators or delegators in the consensus process on the EtherLite network. They can earn rewards (either block rewards or transactional rewards) based on their participation. This provides an opportunity for coin holders to convert any number of current holdings into staking coins, which in turn earn reward-based dividends.

7.1. POSDAO consensus model

POSDAO creates a layered POS architecture connected by smart contracts (see image below). Sybil Attack control and incentives exist in smart contracts working within the EVM and the execution state is stored on a public chain implementing the POSDAO consensus. On the network protocol level, the underlying BFT consensus exists. This model requires modifications to the BFT consensus algorithm implementations in the Ethereum client to facilitate information exchange between smart contracts and the consensus layer. This includes communication relays regarding consensus faults and validator set management.

7.2. Reward Distribution

Reward distribution is the primary incentivising mechanism of the algorithm.The reward distribution rules between validators and delegators inside pools are explained in this section depending on their contribution quantities.. Although EtherLite network provides the option to implement a dual token environment, Etherlite works on single token environment. That token is ETL.

ETL is used for –

1. Staking for EtherLite
2. Block Rewards
3. Transaction Fees (paid to validators only)

The block reward is paid via a 18% annual inflationary measure applied to the native staking coin (ETL)

7.3 Reward Structure

7.3.1 **Transaction Fees**

The validator that validates the block collects the transaction fees associated with that block. Only that validator will get the transaction fees. The transaction fee is basically the gas fee on each on-chain transaction. It indicates the consumption toward computational expenses on the EtherLite network. Transaction fee is charged on execution that is done on the blockchain such as implementing a smart contract or even performing transaction on the blockchain.

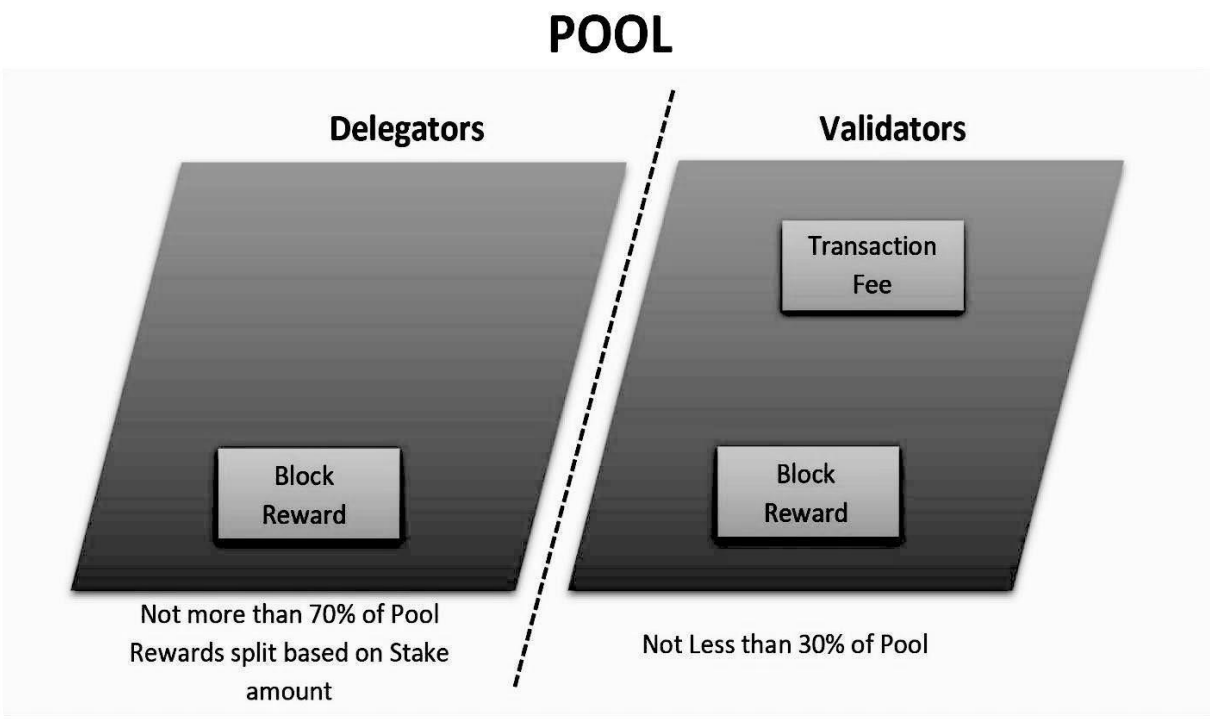
7.3.2 Bridge Fees

In case of inter-blockchain transactions, a bridge is required to connect the blockchains. Entry and Exit fee is charged as transaction moves between EVM based blockchains. The fee is distributed to validators and delegators based on their staking ratios.

7.3.3 Fixed Block Rewards

EtherLite's reward distribution mechanism mints reward tokens for all active validators and their delegators. If a validator is removed due to misbehaviour, its pool is not included in the reward distribution.

The Figure below explains the reward structure:



7.4 Etherlite Reward Distribution Rules

In order to maintain fairness and incentivize elected validators, reward distribution is calculated according to the following rules:

1. After the staking epoch, each pool within the validator set receives an equal portion of the reward (assuming all validators always generate blocks and do not skip them). If a validator skips a block then accordingly reward reduces on pro rata basis.

1. As long as the entire delegators' stake percentage is less than 70%, pool rewards are split between validators and staking delegators.
2. Even if the total delegators' stake surpasses 70%, the validator is guaranteed to get at least 30% of the pool reward; the delegators' payouts are modified, and the validator still receives 30%.

7.5 Validator Set Formation

Any address with the minimum required candidate stake (100,000 ETL) can become a validator. When an address calls the “addPool” contract function and meets the minimum required candidate stake, it becomes a candidate and forms a new pool.

7.5.1 Network participants

The maximum number of EtherLite network members is limited by the MAX

CANDIDATES and MAX VALIDATORS settings. (The maximum number of candidates is 151254, and the maximum number of validators is 50418)

To become a delegator, any random address having at least DELEGATOR MIN STAKE native coins (50,000 ETL) can stake their tokens.

7.5.2 Staking epochs

The network's operation is divided into staking epochs (STAKING_EPOCH_PERIOD – 1 week with 3 sec of block time). A new staking epoch begins immediately following the termination of the previous epoch.

There is a different validator set in each Staking Epoch to avoid attacks. At the beginning of each staking epoch, the algorithm selects a new validator set from the current list of candidates and creates a snapshot of the current state of the validators' pools. If there are fewer than MAX_VALIDATORS+1 candidates, every candidate becomes a validator. The snapshot is used to calculate the reward amount for validators and delegators when they claim the reward.

7.5.3 Becoming a candidate

The maximum number of EtherLite network members is limited by the MAX CANDIDATES and MAX VALIDATORS settings. (The maximum number of candidates is 151254, and the maximum number of validators is 50418)

To become a delegator, any random address having at least DELEGATOR MIN STAKE native coins (50,000 ETL) can stake their tokens.

A new active pool is created for address X, and this account becomes a candidate account.

- Address X is the staking address used to collect rewards and place stakes into their own pool.
- Address Y (the mining address) is used by the validator's node to sign blocks and participate

in the randomness beacon (see figure 3), and report on malicious validators. This address is defined in the engine_signer config option of validator’s OpenEthereum node.

7.5.4 Candidate pools

The EtherLite algorithm chooses active candidate pools to participate as validators in the next validator set at the start of each staking epoch. Inactive pools are not taken into account.

If a candidate withdraws all of their tokens/coins from their pool, the pool becomes inactive and does not take part in the next validator selection process.

The candidate can either fully withdraw their coins and remove themselves as a pool, or partially remove their coins (provided that they leave CANDIDATE_MIN_STAKE) and participate in later staking epochs.

7.5.5 Staking and withdrawal to/from a pool

An EtherLite network participants can stake or withdraw their coins to or from pools during the majority of a staking epoch. The exception is a defined period at the end of each epoch

(STAKE_WITHDRAW_DISALLOW_PERIOD is 12 hours for 3-second blocks). This measure prevents stake manipulation based on the random seed value generated towards the very end.

- On their pool, a candidate's or validator's total stake must be less than CANDIDATE MIN STAKE (100,000 ETL)
- A delegator's total stake on any pool must be less than DELEGATOR MIN STAKE (50,000 ETL)

The minimum investment is very high, which encourages institutional investors to participate as candidates and validators. This large stake creates additional incentives for candidates to protect their nodes and prevent DoS attacks. However, DoS attacks on individual validator nodes are still possible. A validator's role includes defending against such assaults by utilising ISPs that offer DoS protection.

Additional token staking or withdrawal to/from a pool during the current staking epoch (and thereby changing the size of the pool) does not impact the current pool reward. The status of the pool decides the payout at the start of the staking era. However, these changes will impact validator selection probability for the following staking epoch.

A participant cannot withdraw their coins from an active validator's pool unless the amount was staked during the current staking epoch (this amount hasn’t been allotted as a stake yet, so it can be withdrawn). Coins can be withdrawn from a candidate's pool at any time (because the candidate is not a validator).

A participant (delegator or validator) can arrange a withdrawal from an active validator's pool if they desire to quit the collection or lower their staked amount. The selected amount can be claimed after the current staking epoch is complete.

Suppose a validator wants to terminate their validator status on the next staking epoch. In that case, they can schedule a withdrawal of their staked amount or call the contract’s *removeMyPool* function (In this scenario, the pool becomes inactive, and the algorithm will not pick it at the start of the following staking epoch).

7.5.6 Moving stakes

An EtherLite network participant (delegator or candidate) can move their full or partial stake amount from one pool to another without withdrawing the amount from the contract. However, the withdrawal rules stated above apply to such a change.

7.5.7 New staking epoch, validator selection, and finalizing changes

During each staking epoch, participants can call these functions:

- StakingAuRa.stake: to place stakes;
- StakingAuRa.addPool: to create a new pool (for candidates only);
- StakingAuRa.removeMyPool: to remove an existing pool, make it inactive (for candidates and validators);
- StakingAuRa.withdraw, StakingAuRa.orderWithdraw,
- StakingAuRa.claimOrderedWithdraw: to withdraw stakes;
- StakingAuRa.moveStake: to move stakes.
- StakingAuRa.claimReward: to withdraw a reward from the specified pool for the specified staking epochs.

On the last block of the staking epoch, the BlockRewardAuRa.reward function calls the

ValidatorSetAuRa.newValidatorSet function. This function:

- selects new validators;
- writes the list of new validators (validator set) into the pending list;
- increments the staking epoch number which is returned by StakingAuRa.stakingEpoch getter;
- resets the number of the validator set apply block which is returned by the

ValidatorSetAuRa.validatorSetApplyBlock getter.

If the total number of candidates and validators is less than or equal to MAX_VALIDATORS, they will all become validators on the next staking epoch. Otherwise, the validators are selected based on a random value and weighted by their pool sizes (the larger the pool, the more likely the candidate becomes a validator). The random seed is taken from the RandomAuRa contract described below.

The pending list of new validators (their mining addresses) can be read by the ValidatorSetAuRa.getPendingValidators getter.

The validator set cannot be changed immediately because the InitiateChange event must be emitted (see the OpenEthereum Wiki) prior to this change, so the pending validator set is queued by the ValidatorSetAuRa.newValidatorSet function to be handled later by the **ValidatorSetAuRa.emitInitiateChange** and **ValidatorSetAuRa.finalizeChange** functions.

The **ValidatorSetAuRa.finalizeChange** function is only called for one InitiateChange event. If there are several such events, finalizeChange is called only once for the first emitted event (and the additional InitiateChange events

are ignored). So, we can't emit the next InitiateChange unless the previous event is not finalized.

For that reason, there is a pending validator set used by the ValidatorSetAuRa.emitInitiateChange function. When the **ValidatorSetAuRa.emitInitiateChangeCallable** getter returns true, a validator's node calls ValidatorSetAuRa.emitInitiateChange which reads the pending validator set and emits the InitiateChange event to let the validators' nodes know that the set of validators should be changed. When the InitiateChange event is emitted, the ValidatorSetAuRa.emitInitiateChange function sets the initiateChangeAllowed boolean flag to false so that the ValidatorSetAuRa.emitInitiateChangeCallable getter returns false until ValidatorSetAuRa.finalizeChange is called by the engine.

When applying the new validator set, the engine calls the ValidatorSetAuRa.finalizeChange function to notify the ValidatorSetAuRa contract that the pending validator set can be written to the current validator set, which is returned by the ValidatorSetAuRa.getValidators getter.

At this point, the ValidatorSetAuRa.finalizeChange function:

- writes the pending validator set to the current set;
- sets validatorSetApplyBlock to the current block number;
- sets the initiateChangeAllowed boolean flag to true.

The ValidatorSetAuRa.validatorSetApplyBlock getter is used to determine the block number where the current validator set was applied by the engine. If this getter returns 0, it means the new staking epoch is started (ValidatorSetAuRa.newValidatorSet is called) but the new validator set is not yet applied by the engine.

7.5.8 Validator set changes and pending validator set

The validator set is always evaluated at the very end of a staking epoch, however, it can also be changed during a staking epoch if a validator needs to be removed due to misbehavior.

In such cases the malicious validator is removed from the pending validator set (see the internal ValidatorSetAuRa._removeMaliciousValidator function) and the new pending validator set is marked to be handled later by the

ValidatorSetAuRa.emitInitiateChange and

ValidatorSetAuRa.finalizeChange functions.

As with staking epoch validator set changes, the ValidatorSetAuRa.finalizeChange function is used to finalize the validator set change when the malicious validator is removed.

The pending validator set checkmark (see ValidatorSetAuRa._pendingValidatorsChanged internal boolean flag) is used to finalize different validator sets one after another. This may be required when a malicious validator is removed and the new staking epoch begins immediately after the removal, or when one malicious validator is removed at block number N, but another malicious validator is removed on the next block N+1. Without the boolean checkmark, such cases would be handled incorrectly because some of the corresponding InitiateChange events would be ignored by the OpenEthereum engine (as mentioned above)

7.5.9 Random seed accumulation

When the number of pools (validators + candidates) is more than MAX_VALIDATORS, a random seed is used to select MAX_VALIDATORS validators for a new staking epoch (see the ValidatorSetAuRa.newValidatorSet function).

The random seed is stored in the RandomAuRa contract (see the RandomAuRa.currentSeed getter) and generated in the RANDAO manner. There are several collection rounds per staking epoch, each of which is split into two equal phases - a commit phase and reveal phase:

1. Commit phase: Each validator node generates its secret number on each phase and calls the RandomAuRa.commitHash function to commit the hash of the secret (one time per each commit phase).

2. Reveal phase: Each node passes its secret to the RandomAuRa.revealNumber function (once per each reveal phase). The revealNumber function XORs the revealed secret with the current seed stored in the contract, increasing entropy in ever collection round.

Committing/revealing the secret is mandatory for each validator. These functions are called automatically by each validator’s node from the validator's *mining address*.

The length of a collection round is defined at network startup in the InitializerAuRa contract constructor when starting from genesis (or through the RandomAuRa.initialize function when starting on an existing network).

The RandomAuRa.onFinishCollectRound function is called on each block by the BlockRewardAuRa.reward function. At the end of each collection round

RandomAuRa.onFinishCollectRound checks whether the validator skipped revealing the secret during the collection round and increments a skip counter if they did.

On the final collection round of each staking epoch the RandomAuRa.onFinishCollectRound function checks each validator to see

1. if they skipped revealing too often during the current staking epoch or,

2. if they skipped revealing on the last collection round.

If either of these are true, the validator is treated as malicious and removed with the ValidatorSetAuRa.removeMaliciousValidators function.

The maximum number of reveal skips is defined in the **RandomAuRa.onFinishCollectRound** function and depends on the disallow period duration (see the **StakingAuRa.stakeWithdrawDisallowPeriod** getter) and on the length of collection round (see the RandomAuRa.collectRoundLength getter).

The final collection round is especially important to check. During the final round, the last validator in the validator set can decide not to reveal their secret to try to influence the outcome in the ValidatorSetAuRa.newValidatorSet function. For this reason, any validator that doesn't reveal their secret during the last collection round is treated as malicious and removed from the validator set.

To prevent this from happening accidentally due to disconnection, it is recommended that each validator run two separate nodes simultaneously with different internet connections. The first node must have the engine_signer option in a configuration toml file, the second node should not have that option but should have the *watchguard* script which detects if the first node goes offline and sets the engine_signer option for the second node.

For debugging purposes, it is possible to temporarily turn off the punishment for skipping random number reveals in the RandomAuRa contract (see its setPunishForUnreveal function). This boolean flag may be used for testing when a new version of OpenEthereum is unstable and block skipping is evident due to possible bugs in the engine.

Since a new validator set can be applied by the engine at any time (e.g. in the middle of some commits/reveals phase), there may be a case when new validators can't fully commit/reveal their secrets at the very beginning of the staking epoch. Because of this, there is a short grace period after the new validator set is applied during which the skip counter is not incremented if a reveal is skipped (see the code of RandomAuRa.onFinishCollectRound function).

RandomAuRa.commitHash and RandomAuRa.revealNumber are called with a zero gas price. This is enabled because the validators' *mining addresses* can have zero balances.

To protect the network against possible spamming by a malicious validator calling the RandomAuRa functions with zero gas price too often, there is a protection mechanism implemented in the TxPermission.allowedTxTypes function: it doesn't allow creating the RandomAuRa transactions unless they are permitted on the current block (see the RandomAuRa.commitHashCallable and RandomAuRa.revealNumberCallable getters).

The TxPermission.allowedTxTypes getter is called by the OpenEthereum node every time a new transaction is about to be added to a block. This getter checks if the transaction can be included into the block or not.

Removing malicious validators

There are two cases when a validator can be removed due to misbehavior:

1. by the RandomAuRa contract due to not revealing random numbers (see above);

2. by the ValidatorSetAuRa.reportMalicious function (see the OpenEthereum Wiki).

The ValidatorSetAuRa.reportMalicious function can be called by validators to report a specified validator's misbehavior on a specified block number. If more than 2/3 of validators report about the same validator for the same block, the validator is removed from the validator set (see the code of the ValidatorSetAuRa.reportMalicious function).

The validators' nodes call ValidatorSetAuRa.reportMalicious with a zero gas price.

This is enabled because the validators' mining addresses can have zero balances.

To protect the network against possible spamming by a malicious validator calling the ValidatorSetAuRa.reportMalicious function with a zero gas price too often, there is a protection mechanism implemented in the TxPermission.allowedTxTypes function: it doesn't allow creation of the ValidatorSetAuRa.reportMalicious transaction unless it is permitted on the current block (see the ValidatorSetAuRa.reportMaliciousCallable getter).

Moreover, if some validator calls ValidatorSetAuRa.reportMalicious too often (much more often than others), such a validator is also treated as malicious (see the code of ValidatorSetAuRa.reportMaliciousCallable and ValidatorSetAuRa.reportMalicious).

The validator removal process is implemented in the ValidatorSetAuRa._removeMaliciousValidator internal function. It is called by ValidatorSetAuRa._removeMaliciousValidators. When the validator is removed, their mining address is banned for the BAN_PERIOD. This means that the banned validator and their delegators cannot withdraw their stakes until the BAN_PERIOD expires (see the StakingAuRaBase._isWithdrawAllowed internal function). Also, the banned mining

address cannot be added into the validator set during that period.

When the validator is removed from the current validator set, the new pending validator set waits to be passed to the InitiateChange event.

Helpful public getters:

- ValidatorSetAuRa.isValidatorBanned: check if a specified mining address is banned;
- ValidatorSetAuRa.banCounter: read the ban counter, which is incremented when a validator is banned;
- ValidatorSetAuRa.bannedUntil: view the block number when a mining address is released from the ban.

In a test network, it is possible to set one validator that is not removable (see 9.2.11). Such a validator cannot be removed from the validator set even due to misbehavior (see ValidatorSetAuRa._removeMaliciousValidator and the section about the unremovable validator below). This feature prevents test network shutdown if there is a problem with the validator set (for example if all validators leave the network simultaneously).

7.5.10 Block reward distribution

The block reward is distributed among validators and their delegators by the BlockRewardAuRa.reward function. This function is called by the OpenEthereum engine when closing each block.

The pool reward distribution logic is implemented in the BlockRewardAuRaBase._distributeRewards internal function. Changes in stake amounts during the staking epoch do not affect the pool reward during that epoch, but they do affect the future rewards collected on the next staking epoch.

7.5.12 Randomness when selecting a validator

It has to be made sure that there is different set of validators in each Epoch to ensure network safety. The protocol implements a random number generator similar to RANDAO, which is used to randomly select a set of validators from the group of candidates at the start of each staking epoch. For each staking epoch, candidates with a bigger pool have a higher chance of being chosen to a validator set (candidates with higher stakes are probabilistically selected as validators for more staking epochs). If the number of candidates is less than Minimum validators+1, all validators are chosen automatically, and a random number generator is not required.

8. EtherLite network initialization

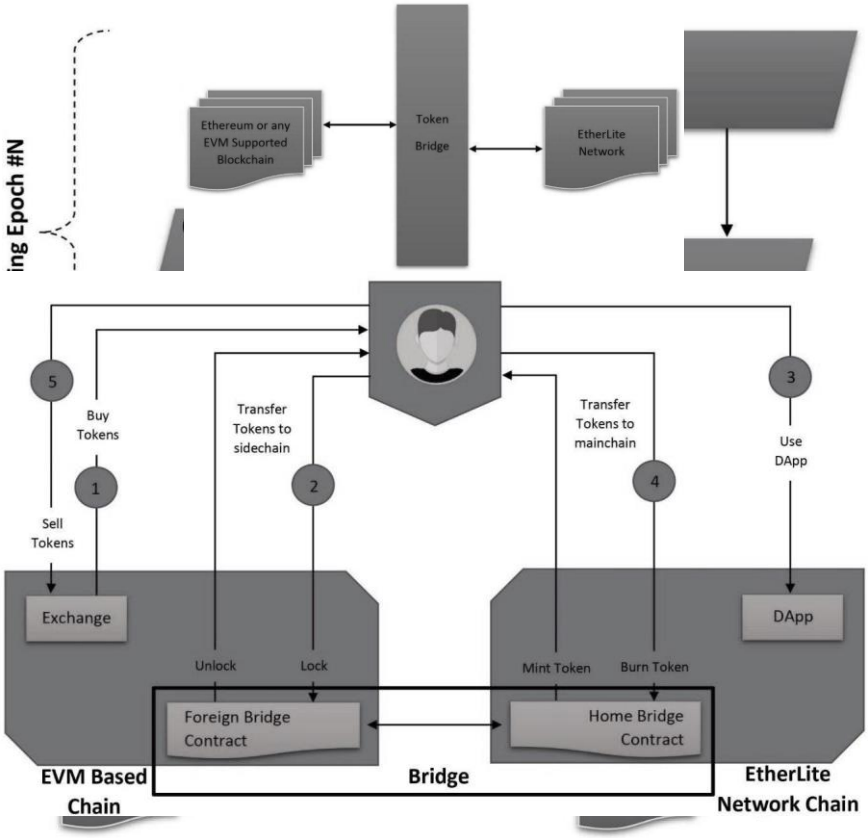
Smart contracts are initialized in the genesis block or in an arbitrary block on an already existing network. The contracts' pre-configured parameters are included in the chain specification bytecode for genesis initialization, and the list of initial validators is also defined in these parameters.

If the network is initiated from the genesis block, all of the addresses in the network (including initial validators) have zero balances. There are no pre- initialized stakes for initial validators, so their pools are also empty.

POSDAO may be configured to run as a standalone blockchain. It can also run using a bridge or bridges which connect to one or more other networks. This bridging situation is detailed below and is utilised in the reference implementation.

8.1. Bridged network scenario

The EtherLite TokenBridge connects all EVM-based chains to the Ethereum mainnet, allowing users to move assets across them. In the reference implementation, two POA



TokenBridge instances connect the POSDAO sidechain network to the Ethereum mainnet.

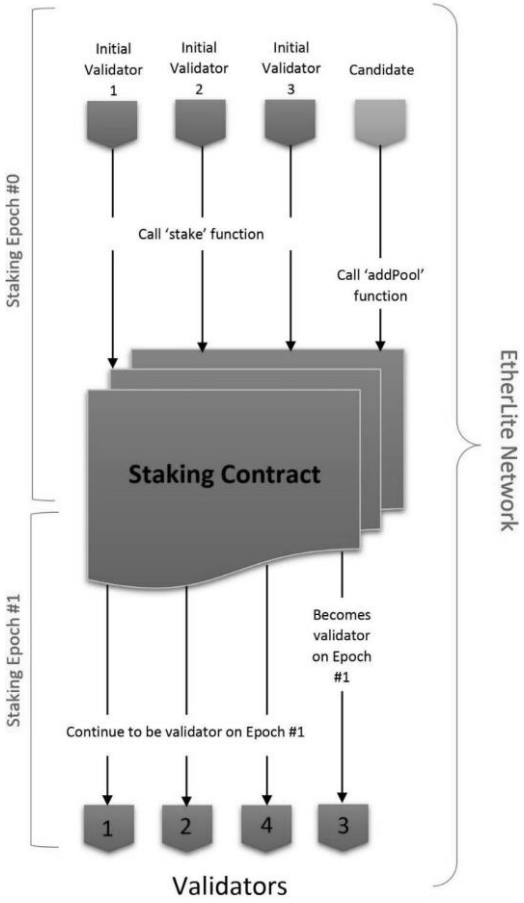
Both bridges have validator sets that are not linked to the POSDAO network's consensus validator set. Bridge validators are in charge of securing token transfers across chains and are not compensated for their efforts.

1.1. Initial validators

Because the validators do not have ETL when the network starts from genesis, they can make service transactions to the Etherlite contracts using zero gas. The validators can make unlimited service transactions but only within the scope of the consensus contracts. The *TxPermission* smart contract protects against possible spam sent from a validator. Figure below shows an example network initialization.

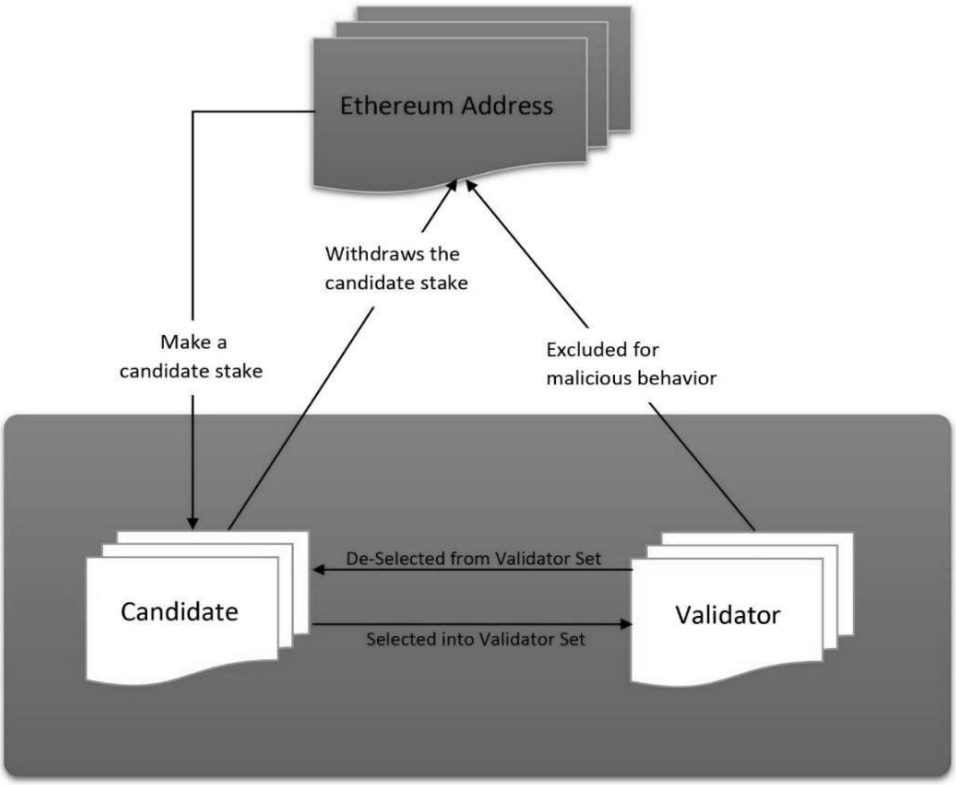
If an initial staking epoch ends and there are no candidates (none of the initial validators made a stake into their pool), the initial validator set is retained for the following staking epoch. However, if at least one candidate appears (the address which added a stake to its pool), any initial validators with empty pools are removed from the set, and the candidate becomes a validator on the new staking epoch. Thus, if an initial validator wants to keep their seat (and still has no staked tokens) after the initial staking epoch, they must place a stake into their own pool.

After the bridge is connected, individuals can bridge their EtherLite tokens and become candidates in the EtherLite network. Figure below illustrates the various possible interactions from a candidate’s address.


Initial Validator 1
Initial Validator 2
Initial Validator 3
Candidate
Staking Epoch #0
Call 'stake' function
Call 'addPool' function
Staking Contract
EtherLite Network
Staking Epoch #1
Becomes validator on Epoch #1
Continue to be validator on Epoch #1
1
2
4
3
Validators

The validators are picked randomly (using the randomization beacon) from the list of candidates if the number of candidates is more than MAX VALIDATORS at the start of a new staking epoch: the larger the candidate's pool, the higher the candidate's chance of becoming a validator.

In addition to the initial validators, an owner who is the foundation team of EtherLite network deploys the genesis contracts and has the ability to upgrade the consensus contracts when required (for example if bugs are found or the code needs to be modified). The owner is a MultiSig smart contract which has a trusted setup. Future implementations may include a voting mechanism to allow validators the ability to vote on upgrades. This would alleviate the need for the owner contract.



9. Reward Structure

9.1. Minimum candidate stake – (100,000 ETL)

Such high minimum candidate stake discourages the potential centralization of candidate seats, where individuals may attempt to register many candidate nodes and thus control a large percentage of validator sets. A high minimum candidate stake also deters a malicious set of validators from attempting a coordinated validator set attack.

1.2. Equal share of the block reward

Within a validator set, each validator pool gets an equal portion of the block reward*. Thus, while a larger stake affects the likelihood of a candidate pool becoming a validator pool, the payout is the same for all validator pools. This ensures that all validators participating in each staking period are on the same page.

*Note: Hard working validators are rewarded more than others. shares will only be equal if every validator produces blocks continuously. If a validator skips blocks, their pool will receive proportionally less reward than other (continuously working) validators. For example, if there are two validators in the validator set and one of them produced 20 blocks, but another only 10 blocks, the first validator's pool will receive 20/(20+10)=66% of the total reward, and the second pool will receive the remaining 34%.

9.2. Proportional reward distribution of 70/30%

The 70/30 distribution ratio is a common revenue sharing heuristic. When set at the initial value, delegators receive block rewards within their validator pool(s) up to 70% of the total pool value, incentivizing delegators to quickly fund candidate pools they believe should be validators.

Once the 70% mark is reached, additional stake returns a proportionally smaller amount. Delegators can now opt to finance other candidate pools, expanding the quantity and diversity of possible candidates, or stake more tokens into the current collection, improving the likelihood of a candidate pool becoming a validator pool in the next staking epoch.

When the ratio is set to 70/30, a validator never receives less than 30% of the pool reward. Validators are responsible for running a node and a reward baseline prevents a situation where delegators can claim an overwhelming percentage of the reward. Once a pool reaches the 70/30 threshold, a validator may choose to increase their stake to attract additional delegator funds or to increase their position on the leaderboard. Successful validator sets (those with a high stake, high transactional throughput, and continuous node uptime) will continue to draw a stake from delegators since reputation is a valuable commodity.

10. Prospective use of EtherLite –

1. Peer-to-Peer Transactions

EtherLite allows for quick and efficient payments. Money may be sent quickly, cheaply, and efficiently with EtherLite. Peer-to-peer (P2P) payments on the blockchain can take

the role of today's costly, sluggish, and bank-driven online payment systems.

2.DeFi (Decentralized Finance)

As said in the introduction, Ethereum's gas prices are at an all-time high. Etherlite can make advantage of Ethereum's current smart contract and tools. In a matter of minutes, developers can migrate their current Ethereum-based DApps, significantly improving performance and decreasing expenses. On Ethereum, high gas prices make many apps more challenging to utilise. This includes DeFi programmes, such as Uniswap and others, as well as DEXs. In addition, fees can sometimes outweigh actual trading amounts, limiting a trader's ability to profit from possibilities. However, users can conduct a lot of swaps or open many positions on EtherLite since the quantity of trades isn't restricted by fee pricing. Faster blocks also enable rapid trading options.

Transferring assets to EtherLite comes with a price (and bridging them back to Ethereum). Trades can be completed as frequently as a trader desires once they have been moved.

3. Community Currencies

CICs (Community Inclusion Currencies) are local currencies that may use to pay for products and services. CICs aren't supposed to replace federal money; instead, they're meant to supplement local trade. CICs allow individuals to preserve federal money (which can be volatile or scarce) for dealings with more prominent companies and government organisations outside of their immediate community while also letting them spend and trade regularly.

By establishing a decentralised, local banking infrastructure, CICs help and empower communities to create jobs, implement social programmes, and support commerce. CCs are being distributed to refugee camps and other disaster-prone areas through efforts sponsored by Grassroots Economics and the Red Cross.

CICs benefit from blockchain technology because it creates a transparent web-based platform for local currency exchange. All users require is a mobile phone and a bespoke wallet application to exchange local currencies based on bonding curves.

The EtherLite Chain provides the infrastructure needed for local digital currencies to grow, thanks to its speed, stability (known low transaction costs that may be subsidised), and dependability.

When markets or supply chains degrade during a crisis, CICs can be utilised to provide targeted assistance. Specific health or food-related businesses may be targeted using blockchain transaction data to assist people in need immediately.

4. Prediction Market

Daily transactions on the EtherLite chain keep gas prices low due to reduced costs and

faster finality, and data is frequently synchronised with the Ethereum mainnet.

5. **Blockchain Games**

The application of blockchain technology isn't limited to cryptocurrencies. According to experts, gaming is expected to be the first genuine use case for blockchain, reshaping the industry and making games more immersive than ever before. Moreover, the way gaming overcomes the remaining roadblocks will serve as a model for other sectors considering widespread blockchain use.

6. **NFT Mint and Transfer**

In March 2021, the NFT art Market ‘SuperRare' secured $9 million in financing. This is the way things are going to be in the future. 'Everything Digital' is the tagline.

Non-fungible tokens (NFTs) are one-of-a-kind, non-transferable assets created on the blockchain. NFTs are being used in various applications, including digital art, collectables, tickets, gaming, digital ownership, and more. In 2020, the digital art business alone is expected to be worth $315 million.

Each NFT has its own set of distinguishing characteristics that are both trackable and unchangeable. NFT artists may sell their work directly to collectors, and anybody can check the validity and number of pieces minted at any moment. In future resale events, the settings can also enable for royalty collection.

NFT game avatars, user settings, and in-game objects may all be moved from one game to the next. Ownership records, domain names, and other assets benefit from easy-to-verify proof-of-ownership. Like other fungible assets (cryptocurrency), Token owners have total control over and management of their assets without relying on a third party.

On Ethereum, high gas prices can make minting and trading NFTs on the mainnet prohibitively expensive. "Based on current gas costs (08/28/2020)," says creator Austin Griffith.

On Ethereum, a single piece of NFT artwork costs between $15.00 and $50.00 to mint, and $3.00 or more to transmit to another account."

With minting, trading, and storing NFTs, EtherLite addresses this problem. Together with all related information, unique assets may be moved to Ethereum with the TokenBridge once the value is created and/or access on Ethereum is necessary.

This technology enables the creation and management of NFTs across the blockchain ecosystem quickly and cost-effectively.

7. Digital Voting

Blockchain technology will empower individuals, which promises to provide safe,

transparent, and non-censorable platforms. Governance and on-chain voting are two applications of this technology that allow for free and open democracy.

Users know their vote is submitted and counts towards the outcome when each vote can be validated and is tamper-proof. This is important for involvement in tiny groups, petitions, local government (including DAOs!), and much bigger communities (such as national elections).

Voting must be simple with minimal limitations, allow for anonymity, be scalable for users, be highly affordable (so no one is excluded), and preferably operate from a smartphone to be effective. In addition, votes must be tracked in real-time and must not be censored by any organisation.

The EtherLite is ideally suited to enable fair and transparent voting processes for various circumstances due to its rapid and affordable processing. As a result, the demand for this technology is more significant than ever, and we're looking forward to seeing more EtherLite ideas connected to digital and blockchain voting.

8. DAO Governance —

Voting and administration for Decentralized Autonomous Organizations (DAOs) on the blockchain. EtherLite is a kind of DAO that uses POSDAO consensus. The validators are a dispersed group of self-sufficient persons that offer STAKE in exchange for rewards.

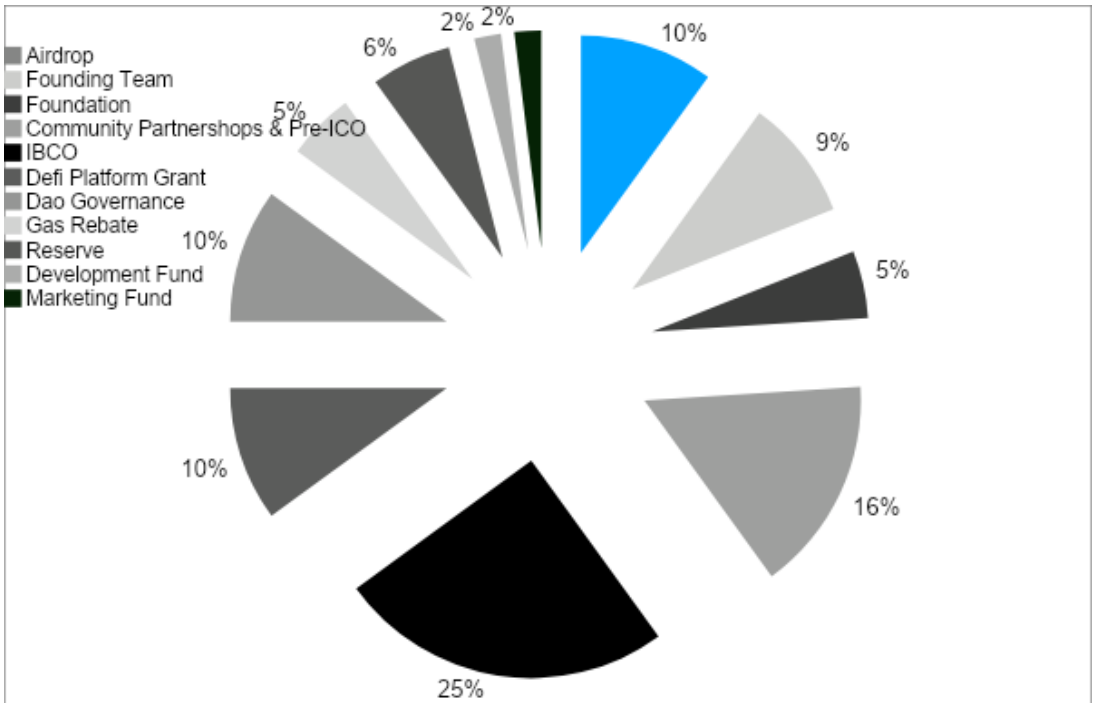
EtherLite may operate DAO governance processes such as proposal and vote systems, community money collecting, and more in DeFi projects.

11. Coin Allocation

- IBCO - Initial Bonding Curve Offering
- Platform Grant - Those Who Want Shift on EtherLite Network
- DAO - Governance (Vote) System to Run the EtherLite
- GAS Rebate - Claim Ethereum Gas Fees in EtherLite Network
- Initial Supply: 21,000,000,000

Community Distribution Chart.

1. Team

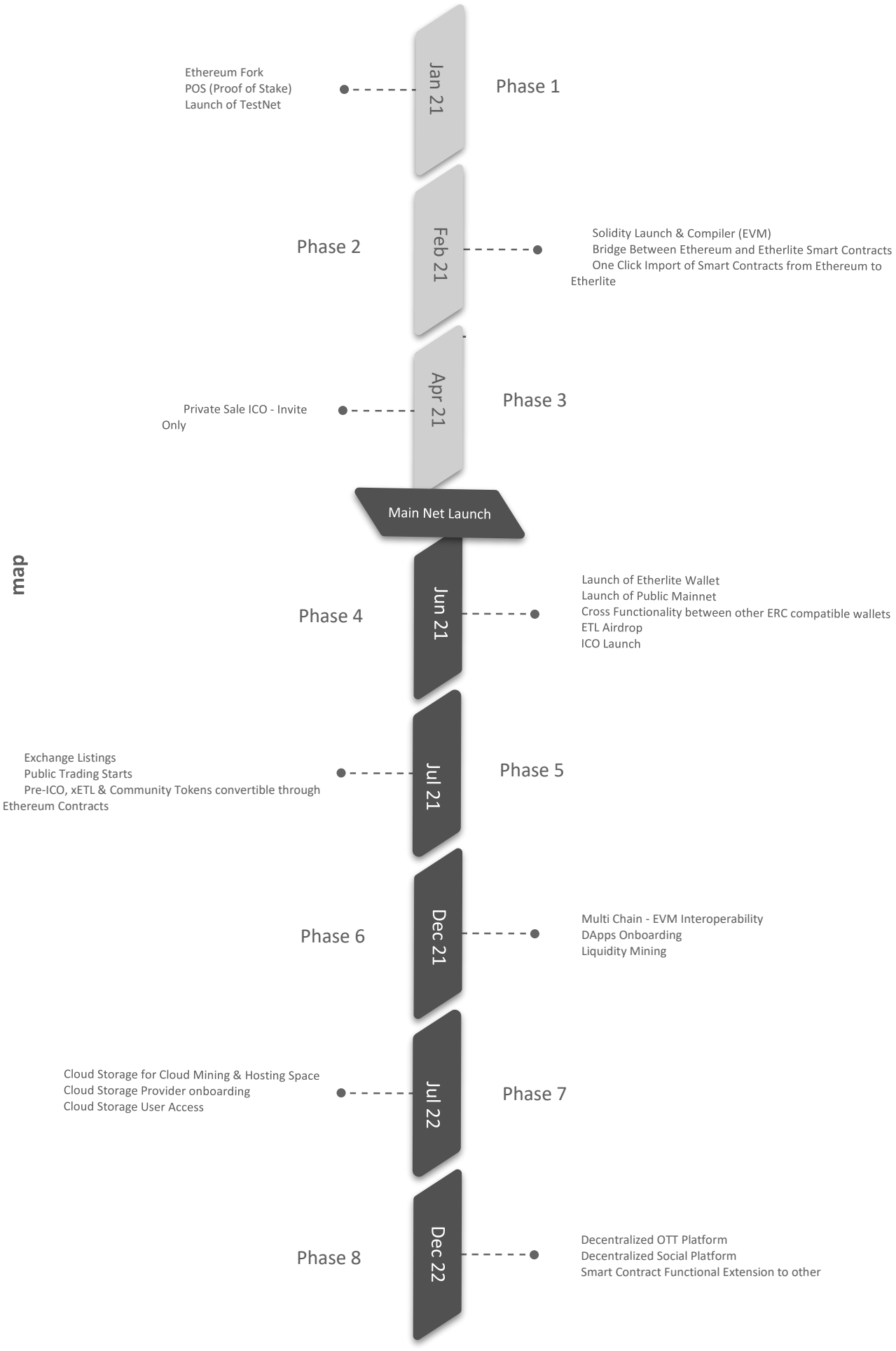


EtherLite Foundation is dedicated to building the infrastructure for a more democratic and efficient future. Our team is made up of engineers, scientists, researchers, designers, and entrepreneurs who share the same vision.

Together, we're working to improve everyone's lives by making advanced technologies more accessible and seamlessly integrated. True to the maxim of decentralization, our team is distributed across the world. Join our community to make this vision a reality.

To know more about EtherLite team click here

13. Road map


Jan 21
Phase 1
Ethereum Fork
POS (Proof of Stake)
Launch of TestNet
Feb 21
Phase 2
Solidity Launch & Compiler (EVM)
Bridge Between Ethereum and Etherlite Smart Contracts
One Click Import of Smart Contracts from Ethereum to Etherlite
Apr 21
Phase 3
Private Sale ICO - Invite Only
Main Net Launch
Jun 21
Phase 4
Launch of Etherlite Wallet
Launch of Public Mainnet
Cross Functionality between other ERC compatible wallets
ETL Airdrop
ICO Launch
Jul 21
Phase 5
Exchange Listings
Public Trading Starts
Pre-ICO, xETL & Community Tokens convertible through Ethereum Contracts
Dec 21
Phase 6
Multi Chain - EVM Interoperability
DApps Onboarding
Liquidity Mining
Jul 22
Phase 7
Cloud Storage for Cloud Mining & Hosting Space
Cloud Storage Provider onboarding
Cloud Storage User Access
Dec 22
Phase 8
Decentralized OTT Platform
Decentralized Social Platform
Smart Contract Functional Extension to other

14. References:

1. https://bitcoin.org/bitcoin.pdf
2. https://ethereum.org/en/whitepaper/
3. https://www.xdaichain.com/for-validators/posdao-whitepaper
4. https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/
5. https://github.com/openethereum/openethereum
6. https://www.xdaichain.com/about-xdai/use-cases
7. https://github.com/ethereum/wiki/wiki/Ethereum-introduction
8. https://www.geeksforgeeks.org/practical-byzantine-fault-tolerancepbft/
9. https://www.xdaichain.com/for-validators/posdao-whitepaper
10. https://openethereum.github.io/Network-Configuration