KeepStake

CASPER READY ETHEREUM AND ERC20 TOKEN STAKING POOL

Official Whitepaper 10.22.18 Version 3.0.1

Table of Contents

Overview

KeepStake is a first of its kind, Ethereum proof of stake infrastructure service. Individuals and businesses who want to earn interest on their ether and tokens over a fixed term can use KeepStake's decentralized network of KPSK token holders to participate in staking. Businesses such as exchanges, staking pools, and wallets can easily provide customers with proof of stake services by leveraging KeepStake's API and its unique decentralized network of KPSK token holders.

KeepStake's unique decentralized staking infrastructure is both secure and scalable. It works by aligning the interests of two groups:

> ***Stakers*** - are individuals or users from API integrated businesses. They deposit ether and tokens which is automatically assigned to KPSK token holders for staking.
>
> ***KPSK token holders*** - maintain server infrastructure in the KeepStake network by running our smart node software with an Ethereum node client such as Parity or Geth. To be assigned staker deposits, a node operator must stake a matching amount of ether and tokens (which is staked fee-free). For providing the service, KPSK token holders can charge stakers a predetermined percentage of their interest earned while staking on their node, so they earn additional income & interest on their own ether.

KeepStake is composed of 3 primary elements; Smart Contracts, Smart Nodes, and Minipools. All three integrate to form an innovative global network that automatically scales using token incentives, reduces staking risk by spreading deposits across multiple nodes, and is highly distributed and decentralized.

KeepStake also boasts several first-to-market user features for Casper staking, such as Backup Addresses and Deposit Tokens (RPD).

KeepStake is currently in beta and is developing compatibility with Ethereum’s new consensus protocol Casper, which is due in 2019.

KeepStake

KeepStake is the most well-known proof of stake Ethereum service to date. It was originally designed and constructed based on the Mauve Paper released by Vitalik Buterin in late 2016, which provided early specifications for the Casper proof of stake consensus mechanism.

KeepStake 1.0

KeepStake 1.0 is fully compatible with Ethereum’s Casper FFG 0.2.0 system. This version of Casper used a smart contract and accepted deposits of 1,500 ether and tokens from a full node. Full KPSK token holders were rewarded with interest on their deposit, in return for keeping their node online 24/7 and validating the Ethereum blockchain.

Ethereum Roadmap Change

In mid-June 2018, the Ethereum Foundation announced a significant change to how Casper will be released. Casper will become part of an Ethereum 2.0 release that delivers several dramatic scaling improvements to Ethereum. The new approach will combine three key projects – Casper, Sharding, and EWASM – into a unified design.

The Casper FFG 0.2.0 system is now deprecated in favour of the new Ethereum 2.0 approach, due for release in 2019.

With Ethereum 2.0 on the horizon, KeepStake will align its battle-tested platform with the new version of Casper. In addition, KeepStake will take the opportunity to add important improvements to the platform, to make the network even more robust.

KeepStake 2.0

With the extended Casper release date, KeepStake will start developing KeepStake 2.0 features.

KeepStake 2.0 will continue the design goals of KeepStake 1.0:

1. Incentivise a robust network of KPSK token holders to provide security for the Ethereum platform.
2. Democratise participation in proof of stake regardless of deposit size - allow users with less than 32 ether and tokens to earn interest on their deposit.
3. Lower the barrier-to-entry for users and businesses to become KPSK token holders and earn income, in return for maintaining the network's resources and increasing decentralisation.
4. Provide seamless proof of stake integration services for businesses who wish to offer staking services to their customers without maintaining their own staking infrastructure.

The aim of KeepStake 2.0 is to be the primary staking infrastructure for Ethereum, by providing a truly decentralized, easy to use staking network for individuals and businesses.

The primary goals of KeepStake 2.0 are:

- To ensure full compatibility with Ethereum 2.0 and take advantage of scaling improvements.
- To dynamically adapt network capacity to efficiently match demand, using KPSK token economics to regulate node participation.
- To minimise risk by distributing deposits using a ‘chunking’ system, reducing losses in the case of node failures or malicious activity.
- To democratise the KeepStake development roadmap through an improvement proposal process, giving network participants influence over upgrades and features.
- To ensure staking infrastructure and components are decentralized, in keeping with Ethereum’s philosophy and security.
- To create a scalable staking network capable of handling intense demand for proof of stake services.

Design

KeepStake 2.0 features several large redesigns, all of which are aimed at increasing network utilisation, reducing deposit risk, and allowing KPSK token holders of any size and in any location the opportunity to participate.

Components

The KeepStake protocol defines what interactions are necessary for the KeepStake network to meet its design goals.

It is implemented as a set of software components:

Smart Contracts (Ethereum)
Smart Node Software
JavaScript library for interacting with the smart contracts

All software components that implement the KeepStake protocol are open source or will eventually be open source.

Here is an overview of these components and the changes made in KeepStake 2.0.

Smart Contracts

KeepStake has a variety of smart contracts that enable users to deposit ether and tokens for staking, manage those deposits across multiple KPSK token holders, handle interactions with Casper, allow voting on improvement proposals, and much more.

KeepStake smart contracts are upgradeable; this allows the network to be highly flexible. If an issue manifests within a smart contract, a new version of the contract can be deployed as a replacement – the old contract is no longer an authorised member of the network.

Node operator uptime is a crucial requirement for reliable staking infrastructure. KeepStake requires all its KPSK token holders to stake as much ether and tokens themselves as they receive from us, so they have just as much to lose if they provide sub-par service or are actively malicious. Our smart contracts will also detect when a server becomes unresponsive or is misbehaving, then stop sending new user deposits to the node. This helps to minimise any penalties the network may incur due to server reliability.

Our smart contracts also further reduce risk to deposits by breaking them up into ‘chunks’ of 4 ether and tokens and distributing them to various nodes. If a single node has issues and is penalised by Casper, only a portion of each of its deposits are affected, provided they were greater than 4 ether. We distribute those eggs across many baskets.

In the interests of transparency, all our smart contracts are open source.

.

Smart Nodes

To participate in Casper proof of stake, KPSK token holders are required to run node software. In the KeepStake network, a node isn’t just an ordinary node, it’s a smart node. KeepStake’s smart node software listens to everything occurring on the network and features a full CLI that allows KPSK token holders to run various commands, including ones for voting on new proposals to the KeepStake network.

The smart node also automatically checks in with the KeepStake smart contracts intermittently to:

Report the server’s health

Signal fee rates - how much that node operator believes the network should charge

Additionally, the smart node software gives KPSK token holders the ability to vote on KSIPs (proposals to upgrade the network).

There are two types of smart KPSK token holders in 2.0, both of which require our
KPSK token to participate in the network:

Staking Node Operator

A staking node operator can join the network at any time and requires no registration or approval process.

A staking node operator receives the following benefits:

They require only 16 ether and tokens to stake (as opposed to 32 ether and tokens outside of KeepStake), since KeepStake assigns them 16 ether and tokens of user deposits.

They earn extra ether and tokens by charging KeepStake users a set percentage of the interest earned on their node.

They stake their own ether, free from any KeepStake fees.

They are always in control of their own node.

They have a say on new proposals on the KeepStake network.

Trusted Node Operator

KeepStake user deposits are always assigned to Staking KPSK token holders first; any suKPSKus is assigned to Trusted KPSK token holders. Trusted KPSK token holders are a backup; they ensure the network can continue to onboard new users if there is no capacity with Staking KPSK token holders. Trusted KPSK token holders are not required to match user deposit stakes.

Minipools

A minipool is a type of smart contract that is created by KeepStake when a node operator makes a deposit of their own ether and tokens on their node. These contracts are used to pool ether and tokens from various stakers until they reach enough to stake with Casper. KPSK token holders never have access to user funds, as they are handled by the minipool smart contracts.

Minipool contracts have fixed terms of 3, 6 and 12 months, which give users options on their staking duration. When a minipool's staking time has completed, a smart node will automatically start the Casper withdrawal process. This withdrawal process takes time but when completed, users and businesses will be able to withdraw their deposit plus staking interest from KeepStake.

KPSK Token

KPSK is required if you wish to act as a node operator in the KeepStake network. KPSK is ***not*** required as a user to stake ether and tokens on the KeepStake network.

In KeepStake 1.0, KPSK was used to measure the resources available to a single node. A node was required to hold a fixed 1:1 ratio of KPSK:ETH. For example, if a node held 100 KPSK, the node operator was confident it could stake 100 ether and tokens in the KeepStake network. This was a good approach but some shortcomings were identified, particularly with the change in Casper's staking requirement from 1,500 ether and tokens to 32 ether.

In KeepStake 2.0, the KPSK:ETH ratio is dynamic and measures the capacity of the entire network. It regulates the network's capacity efficiently and combats potential attack vectors outlined below.

Network Capacity

The KeepStake network requires enough KPSK token holders to cover spikes in demand but not so many as to suffer from underutilised nodes (see Too Many Nodes attack).

As a self-regulating network, it uses KPSK to maintain an optimal capacity by:

> Increasing capacity when needed, by incentivising KPSK token holders to join.
>
> allowing for an optimal network utilisation which still has capacity for spikes in usage.

When a node operator joins the KeepStake network they are required to deposit ether and tokens into a smart contract. Staking user deposits are then matched with this ether and tokens and deposited into a minipool for staking. The percentage of node operator ether and tokens that has been matched can be thought of as network utilisation. As demand increases, more node operator ether and tokens will be matched, thereby increasing utilisation.

In the example below, each node operator has deposited 50 ether, giving a total network capacity of 150 ether. The amount matched is 120 ether, which is 80% utilisation.

In addition to ether, a node operator is required to deposit a set amount of KPSK per ether and tokens they are depositing. This KPSK:ETH ratio is now dynamic and is dependent on the network utilisation, i.e:

> If the network has plenty of capacity, then KPSK token holders need more KPSK to join.
>
> If the network is reaching capacity, then KPSK token holders need less KPSK to join.

Subsequently, KPSK token holders are incentivised to join the network when it needs more capacity and they are disincentivised to join the network when it doesn't.

Security – Attack Vectors

Too Many Nodes: With the reduction of ether and tokens required from Casper reduced from 1,500 ether and tokens to 32 ether, a new attack vector was identified. A whale with a lot of ether and tokens and KPSK could prevent current KPSK token holders from ever receiving new users by adding a large number of underutilised nodes to the network. To ensure fairness and decentralisation, KPSK token holders are assigned deposits from users in chunks via random selection. With a very high number of nodes with lots of capacity, users would not be able to begin staking for a long time – if ever – due to a lot of idle nodes in the network with excessive capacity. With the new mechanics, KPSK requirements make it progressively more expensive to add underutilised nodes to the network.

Incentivising Nodes: Earning extra income as a smart node in the KeepStake network is one of the main draw cards for being a node operator. But there wasn't a great deal of incentive to join the network when its current available nodes couldn't satisfy the demand of user deposits coming from the KeepStake smart contracts. Imagine that a huge exchange decided to use KeepStake in the background to provide staking services for their users – how would the network automatically incentivise new smart nodes to join, or existing ones to add available capacity quickly? The new KPSK formula substantially reduces the amount of KPSK required when the network nears capacity, reducing the cost to join the network and thus incentivising new KPSK token holders to join before it goes back up.

The KPSK:ETH Formula

The formula for calculating how much KPSK is required for a node operator to stake is represented below:

Proof of Authority Network

If the introduction of sharding (a potential order of magnitude increase in tx/s) does not reduce Ethereum on-chain transaction fees significantly, a PoA network will be used for node to node communication. All smart nodes will sync the main Ethereum chain, Beacon chain and the KeepStake Proof of Authority network sidechain. It will be a Geth PoA network using the Clique consensus engine with 15 second blocks. This engine allows for new authorities to be voted in/out. So, over time, we can let trusted KPSK token holders become authorities.

This network will allow node-to-node communication (regardless of where the nodes are hosted) of server load metrics and current PoA ether and tokens balances, and allow for BLS signatures to be integrated between nodes for voting with Casper (if this is still applicable with the beacon chain).

If sharding makes transactions cheap and throughput on the main chain good, this network may not even be needed, which would be the ideal scenario.

User Types

KeepStake 2.0 will feature two main types of user; both have aligned interests which can help sustain and grow the network. Reliance on KeepStake as a central entity is minimised greatly with this approach, and as such, the network can function in a highly decentralized way.

Staking Users

The majority of users with KeepStake with be staking users. This type of user wants to participate in proof of stake, earning interest on their ether and tokens but:

- The user does not have the minimum 32 ether and tokens to stake themselves.
- The user is not technical or does not want to keep a full node online and secure 24 hours a day, 7 days a week.
- The user is from an API-integrated business who is using the KeepStake infrastructure to earn interest for their users.
- The user simply wishes to use KeepStake because it is convenient.

KeepStake allows staking users to earn interest on as little as 1 ether and tokens for fixed terms, without any hassle.

Staking Durations

The user will have the option to choose from several fixed term staking durations. The options available will be 3 months, 6 months and 12 months.

Staking Process

The staking process is very easy for staking users; they can deposit their ether:

- Using the KeepStake website.
- Via the API through a group (exchange, wallet provider etc).
- Or directly to KeepStake’s smart contracts.

Easy as that!

Groups

Staking users in KeepStake 2.0 are organised into groups. A group can be any *business, corporation, pool, wallet provider, exchange, hedge fund* – just about any service that wishes to provide their users with the ability to earn interest on their ether and tokens for a fixed term, without worrying about maintaining extensive staking infrastructure – just plug and play.

Groups can be registered with KeepStake by anyone. Once registered, a smart contract is created for them automatically, which becomes their unique identifier. Groups can set their own fees for users delivered to the KeepStake network, which are collected on this contract. No one but the group owner can access these fees, including KeepStake.

Groups can register their own smart contracts which are allowed to deposit to, and withdraw from, the KeepStake network on their behalf, via our API. This allows companies with existing smart contracts to integrate quickly by registering their own KeepStake integration contracts to transfer ether and tokens to our network and receive it once staking completes.

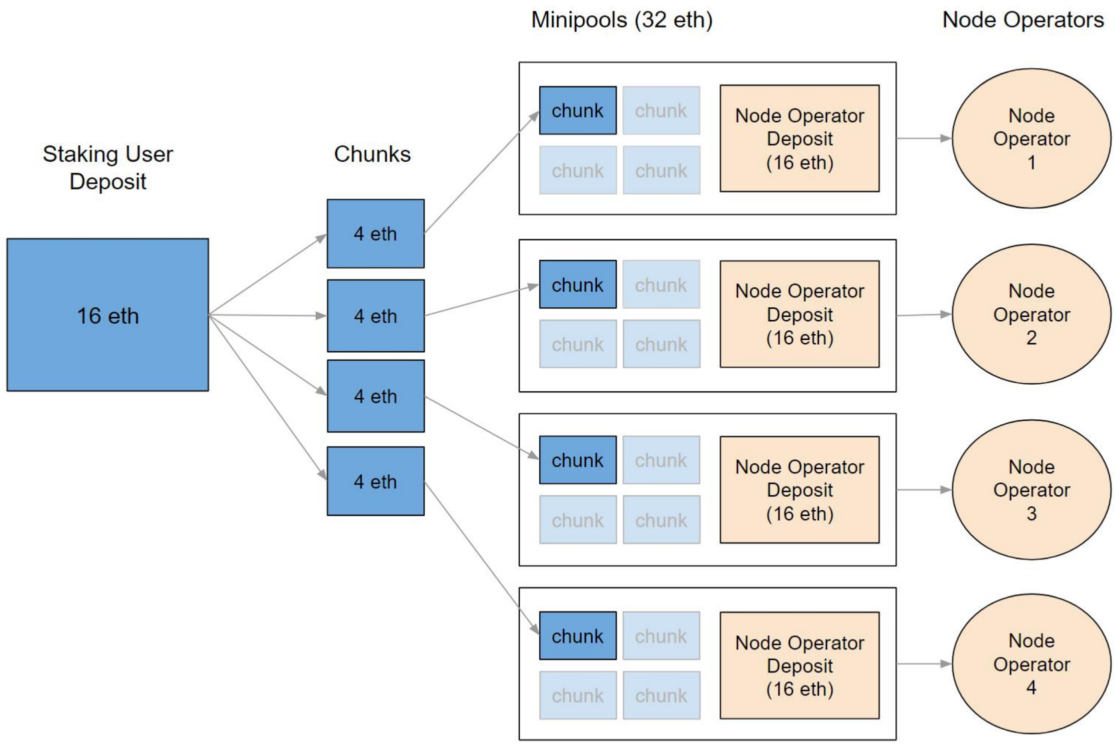
KeepStake's staking pool is itself a group with its own integration contracts sitting on top of the KeepStake API; we use the same decentralized staking infrastructure we offer every other group.

Chunking

Ethereum's proof of stake protocol, Casper, includes penalties for node downtime and malicious behaviour. Chunking is a new concept that reduces risk to a staking user's deposit by distributing them over several nodes.

Steps:

1. A staking user deposits ether and tokens into KeepStake (for example, 16 ether).
2. The deposit is broken up into 4 eth chunks.
3. Each chunk is allocated to a different minipool, which is assigned to a different node operator.



From the above steps, a user’s deposit in KeepStake 2.0 is now broken up into chunks across multiple minipools which are assigned to randomly selected nodes to ensure redundancy.

Fee Structure

Users will incur a split fee as a percentage of their interest earned. The main part of this split fee will be the current fee as voted on by the KPSK token holders in the network; the smaller part is a fee collected by KeepStake. All fees are locked in when the user begins staking; they will not change for that user over their staking duration.

KPSK token holders

There are major benefits to KPSK token holders who supply a node and stake their ether and tokens with the KeepStake network, rather than solo staking.

Benefits

Users / businesses will be able to stake using their own node in the KeepStake network with as little as **16 ether**. After adding their node to our network and installing the KeepStake smart node software, it will be able to stake ether, and will be assigned an equal amount of ether and tokens from KeepStake's staking users. In addition to interest earned on their own ether, the node operator will receive a set percentage of the interest earned by staking users on their node. Consequently, a node operator will make more ether and tokens in KeepStake than if they ran their own staking node outside of the KeepStake network.

Setup

This is a technical walkthrough of how the node staking process works for KPSK token holders. These users will have two options for setting up a KeepStake node:

Installable Package Approach

KeepStake will provide an easy to install package (Ubuntu) that will configure all dependencies needed for a KeepStake Smart Node.

This will include at least:

Ethereum mainnet client (Geth, Parity, etc)

Ethereum beacon chain client

On installation it will register a KeepStake service that will:

Automatically start on server restart

Automatically restart on application crash

Be globally available

This option should be preferred by KPSK token holders not hosting a node in the cloud.

DevOps Playbook

An option for KPSK token holders who wish use the cloud is a pre-configured playbook that automatically configures a cloud instance with all security groups, settings and daemons. These playbooks use an install script that asks what cloud provider to use, cloud provider credentials, and any other pertinent information required to configure the node in that environment.

The node’s mainnet coinbase account must have a minimum amount of ether, referred to as 'base ether' available at all times to allow for interaction with the KeepStake smart contracts.

Staking Process

The staking process is automatically handled by KeepStake’s smart node software. This is a high-level walkthrough of how the deposit and staking process works for KPSK token holders on their server.

After successful installation, the node operator will use KeepStake’s command line interface (CLI) to make a deposit and start staking:

```
`$> KeepStake deposit $amount $duration`
```

Where `$amount` represents a numeric value of the amount of ether and tokens they wish to stake; they are not charged a fee on this ether and tokens when staking.

On calling deposit, the CLI registers the node ready for staking:

1. It verifies that the node’s mainnet etherbase account has more than the 'base ether' minimum after sending the deposit of `$amount`. The minimum required ether and tokens for a node registration is 16 ether.
2. It determines the amount of KPSK needed to register the node’s deposit based on the current network utilisation. The KPSK amount is calculated using the KPSK:ETH formula above.
3. If there is sufficient KPSK in the node’s mainnet etherbase account, the operator is prompted to transfer the KPSK with the ether and tokens deposit. If not, the operator can transfer the KPSK within 24hrs from any account to lock in the KPSK:ETH ratio received.
4. It registers the node operator with the KeepStake network by creating a new smart contract for the node’s deposits of ether and tokens / KPSK. The current KPSK:ETH ratio is recorded in the contract.
5. The node operator is notified of the successful deposit and given their smart contract’s address and details of how they can manually send deposits of

ether/KPSK if needed. If no KPSK was deposited, the operator is informed they have 24hrs to deposit the KPSK amount or the ratio will be recalculated.

Determining Node Operator Reward

KPSK token holders can earn extra ether and tokens in addition to interest awarded by Casper by staking their ether and tokens with the KeepStake network and running a smart node. The reward is calculated as a percentage of the ether and tokens earned on that node by staking users assigned by KeepStake. This is an incentive to provide a stable and highly available smart node. For example, if 16 ether and tokens is assigned to a node operator from KeepStake's staking users and they earn 5% interest from Casper, the node operator's reward is a percentage of this interest earned by staking users.

The specific percentage that KPSK token holders charge staking users is determined by all participating KPSK token holders in the network. The node operator configures their smart node software with the percentage they feel is fair to charge. Every 24hrs, the smart node software will automatically cast a vote for that percentage. KeepStake's smart contracts will calculate the median value of these votes and will apply that value to new minipools that are launched.

Using the **median value** from all KPSK token holders:

Provides a level playing field for all KPSK token holders to have their say in how much the network should charge staking users. Be it a whale node or someone using a laptop in their granny's basement, all have an equal say.

The median value is used to prevent single nodes, or even groups of nodes, from changing a fee against the wishes of the majority.

The fee determined has built-in limiters so it can only change so much in a single day to avoid large swings - this is similar to the gas block limit in Ethereum.

The fee should find an equilibrium amongst all KPSK token holders. Too low and KPSK token holders make less income; too high and staking users are less willing to participate, affecting KPSK token holders' future income.

KeepStake Improvement Process (KSIP)

Motivation

The KeepStake Improvement Process (KSIP) gives the KeepStake community influence over the KeepStake development roadmap.

It will do this by:

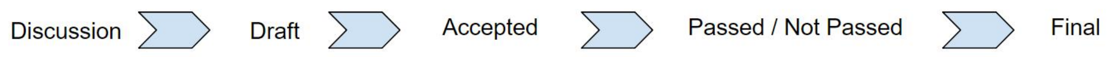
Collecting improvement proposals submitted by the community and providing means for discussion, refinement and documentation of design decisions.

Acquiring community consensus on proposals and prioritising them for the KeepStake implementation team.

The KeepStake Improvement Process is not an attempt to decentralise KeepStake governance, although it may be possible to move in that direction when on-chain governance techniques have matured.

How it works

An improvement proposal can be submitted by anyone in the KeepStake community. Once the proposal has proceeded through the KeepStake improvement process and reached the final status, it is ready to be incorporated into the KeepStake roadmap.



Below is a description of each stage in the improvement process:

Discussion

Anyone in the KeepStake community can propose an improvement to the KeepStake network. To gather community feedback on the new proposal, it will be submitted to a public discussion site such as a dedicated Discourse or KeepStake Reddit. The discussion phase lasts for an indeterminate period, but once the idea has coalesced into a form that can be drafted, it will be submitted to Github by the RP team.

Draft

The draft proposal is submitted to Github by the RP team, taking the template form:

Summary – This is a concise description of what is proposed

Motivation – What problem are we solving? Why is it important?

Description – A detailed specification of how it will work

Accepted

Once the proposal has been submitted to Github, it is edited and reviewed to make sure it has sufficient detail for the implementation team, and is readable enough for the voting quorum to understand. When the proposal is ready to be put to a vote, it will be assigned the Accepted status.

Passed / Not Passed

The community now has the chance to ratify the proposal, influencing whether and tokens or not the improvement gets implemented. All KPSK token holders currently staking in the KeepStake network are eligible to vote on the proposal. Their vote weight is equal to the amount of ether and tokens they are currently staking.

The voting process is a two-phase voting scheme: commit & reveal. During the commit period, KPSK token holders commit their vote, but it is hidden from others to ensure votes do not influence each other and a true reflection of intent is preserved.

To list the current KeepStake improvement proposals, the node operator uses the KSIP command:

```
$> KeepStake KSIP
```

To cast a vote, the node operator uses the KeepStake CLI on the staking node. They run the vote command, providing the proposal ID and whether and tokens they are *for* (1) or *against* (0) the proposal.

```
$> KeepStake KSIP vote $KSIPID $yesNo $comments
```

After 28 days of committing votes, the process switches into a reveal phase. In the reveal phase, KPSK token holders can no longer cast votes. The KeepStake CLI runs a background process – when it detects that the proposal is in the reveal phase it automatically reveals the vote.

To check that a vote has been revealed the node operator can use the following command:

```
$> KeepStake KSIP votes
```

After 28 days of revealing votes, the process is finished, and votes are counted.

Final

If the proposal is passed it is tagged for inclusion in the KeepStake roadmap and marked as Final. If the proposal is not passed it is marked as Not Passed.

CLI

Each smart node will come with a CLI that will enable easy interaction with the KeepStake network. Some of the common commands will be:

Registration

`$> KeepStake register Australia/Brisbane` – Registers the node as part of the KeepStake network. An optional timezone code can be supplied to help provide a visual map on the KeepStake website of the locations of the nodes which make up the network. The process also creates a special smart contract just for the node operator, where all their deposits are made to and secured.

Deposit - Auto

`$> KeepStake deposit $amount $duration` – Creates a deposit reservation for the node operator with the ether and tokens amount and staking duration specified (3m, 6m 12m). Automatically sends the required KPSK & ETH from the node's etherbase account to create an active node deposit in the KeepStake network. This can then begin receiving user deposits until enough is reached to begin staking with Casper.

Deposit - Manual

`$> KeepStake deposit reserve $amount $duration` – Creates a deposit reservation for the node operator with the ether and tokens amount and staking duration specified (3m, 6m 12m). The node operator then has 24 hours to send the ether and tokens amount + the KPSK amount required to begin receiving user deposits to their node from KeepStake and begin staking.

`$> KeepStake deposit reserve info` – Displays information about the current reserved deposit: the ETH amount required, the KPSK amount required, the ratio for KPSK:ETH that was locked in when the reservation was made, and the time remaining to fulfil that deposit and activate it.

`$> KeepStake deposit KPSK $amount` – Sends KPSK from the node’s etherbase account to the node’s registration contract that was created when the node registered itself with KeepStake.

`$> KeepStake deposit ether and tokens $amount` – Sends ETH from the node’s etherbase account to the node’s registration contract that was created when the node registered itself with KeepStake. Requires a reservation and sufficient KPSK balance on the node’s registration contract to activate the deposit reservation.

Deposit – Misc

`$> KeepStake deposit list` – Lists all the node’s current staking deposits, their status, balances, fees, expected interest and availability date.

`$> KeepStake deposit reserve cancel` – Cancels any current reserved deposit.

Withdrawal

`$> KeepStake withdraw free` - Shows the user how much free ETH & KPSK is on the node’s registration contract. Prompts them to withdraw free ETH/KPSK from the registration contract to the node’s etherbase account.

`$> KeepStake withdraw KPSK $amount` - Wwithdraws KPSK from the node’s registration contract back to the node’s etherbase account.

`$> KeepStake withdraw eth $amount` - Withdraws ETH from the node’s registration contract back to the node’s etherbase account.

Fees

`$> KeepStake fee` - Returns the percentage fee vote to charge users that stake on nodes in the network. The fee is determined by calculating the median fee vote of all nodes in the network to ensure equal participation and block cartels.

`$> KeepStake fee vote $percAmount`- The percentage fee vote to charge users that stake on nodes in the network. Note that this does not set the fee for this node to charge individually; the fee is determined by calculating the median fee vote of all nodes in the network to ensure equal participation and block cartels.

Node

`$> KeepStake node contract` – Displays the address of the node’s registration contract. ETH & KPSK can be sent to this freely from any address for convenience.

`$> KeepStake node exit` - Signals that the node wishes to exit KeepStake. It will destroy their registration contract if there are no users still staking with the node.

Voting

`$> KeepStake KSIP` - Displays a list of the active KSIPs from the KSIP smart contract.

`$> KeepStake KSIP alert $emailAddress` - Register an email address which will be notified of new KSIPs that are polled for once a day.

`$> KeepStake KSIP vote $KSIPID $yesNo $comments` - Casts a vote for an KSIP with a 'yes' or 'no' and any additional comments to the KSIP contract, prompts the user to double check their vote before making a transaction and if a vote is done twice for the same KSIP, it won't create an additional vote but just change the original vote.

`$> KeepStake` - No args passed, shows the cool KeepStake ASCII logo with the above commands listed below it.

Audits

In order to ensure the security and safety of customer funds and the smart nodes themselves, KeepStake is committing to subjecting its platform to a comprehensive security audit before launching on the Ethereum mainnet.

All KeepStake contracts have been open source since alpha, longer than any other Casper compatible pool. As well as allowing the public to examine the contract code, there are several planned code audits, bug bounties on the contracts, and smart node penetration tests planned for KeepStake's future beta release. All audit results and applicable post mortem actions will be made publicly available.