ROCKETPOOL

NEXT GENERATION ETHEREUM PROOF OF STAKE NETWORK

18.10.2018

contact@rocketpool.net

Version 2.0.8

Table of Contents

Note: This document is for informational purposes only and does not constitute an offer or solicitation to sell shares or securities in Rocket Pool or any related or associated company. Any such offer or solicitation will be made only by means of a confidential offering memorandum and in accordance with the terms of all applicable securities and other laws. All figures utilised in this document are for demonstration purposes only. Any balances, percentages, or time frames described are purely for demonstrating the concept currently under discussion, unless otherwise noted.

Overview

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

Rocket Pool's unique decentralised 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 which is automatically assigned to Node Operators for staking.

- ***Node Operators*** - maintain server infrastructure in the Rocket Pool 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 (which is staked fee-free). For providing the service, node operators 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.

Rocket Pool 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 decentralised.

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

Rocket Pool is currently in beta and is developing compatibility with Ethereum's new consensus protocol Casper, which is due in 2019.

Rocket Pool

Rocket Pool 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.

Rocket Pool 1.0

Rocket Pool 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 from a full node. Full node operators were rewarded with interest on their deposit, in return for keeping their node online 24/7 and validating the Ethereum blockchain.

A successful public beta of Rocket Pool 1.0 was run over 8 weeks at capacity with 18,385 ether deposited from several hundred users. Rocket Pool 1.0 was well received by the community and feedback was incorporated to refine the product further. All smart contracts for Rocket Pool 1.0 are publicly available to test at our Github repository.

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, Rocket Pool will align its battle-tested platform with the new version of Casper. In addition, Rocket Pool will take the opportunity to add important improvements to the platform, to make the network even more robust.

Rocket Pool 2.0

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

Rocket Pool 2.0 will continue the design goals of Rocket Pool 1.0:

1. Incentivise a robust network of node operators 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 to earn interest on their deposit.
3. Lower the barrier-to-entry for users and businesses to become node operators 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 Rocket Pool 2.0 is to be the primary staking infrastructure for Ethereum, by providing a truly decentralised, easy to use staking network for individuals and businesses.

The primary goals of Rocket Pool 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 RPL 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 Rocket Pool development roadmap through an improvement proposal process, giving network participants influence over upgrades and features.
- To ensure staking infrastructure and components are decentralised, 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

Rocket Pool 2.0 features several large redesigns, all of which are aimed at increasing network utilisation, reducing deposit risk, and allowing node operators of any size and in any location the opportunity to participate.

Components

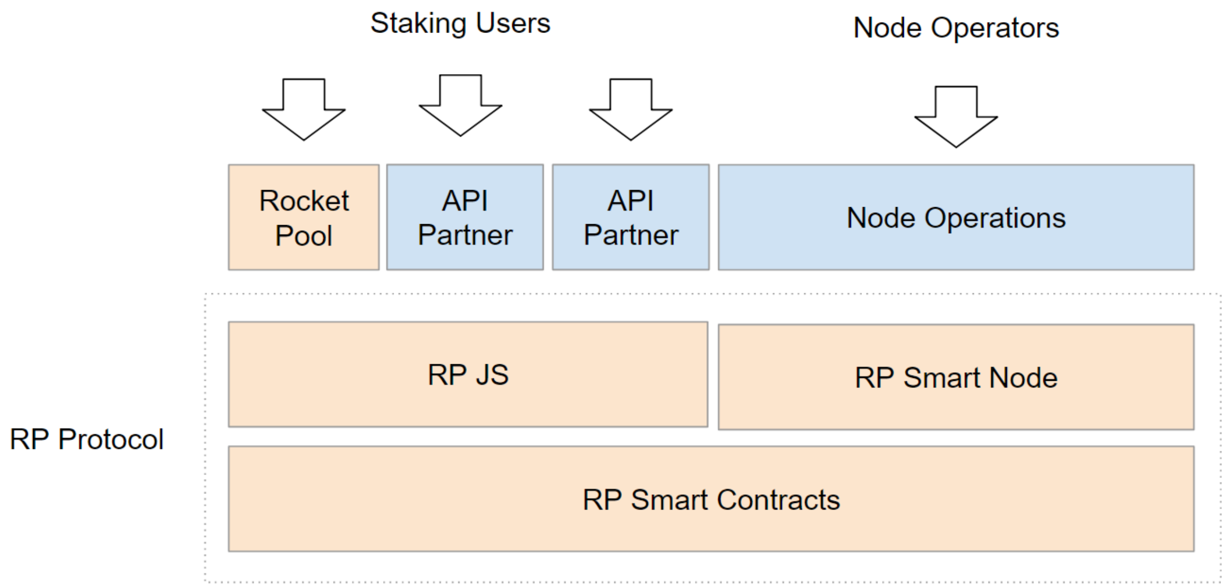
The Rocket Pool protocol defines what interactions are necessary for the Rocket Pool network to meet its design goals.

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 Rocket Pool protocol are open source or will eventually be open source.

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



Smart Contracts

Rocket Pool has a variety of smart contracts that enable users to deposit ether for staking, manage those deposits across multiple node operators, handle interactions with Casper, allow voting on improvement proposals, and much more.

Rocket Pool 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. Rocket Pool requires all its node operators to stake as much ether 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 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. View our GitHub repo for more information.

Smart Nodes

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

The smart node also automatically checks in with the Rocket Pool 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 node operators the ability to vote on RPIPs (proposals to upgrade the network).

There are two types of smart node operators in 2.0, both of which require our RPL 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 to stake (as opposed to 32 ether outside of Rocket Pool), since Rocket Pool assigns them 16 ether of user deposits.
- They earn extra ether by charging Rocket Pool users a set percentage of the interest earned on their node.
- They stake their own ether, free from any Rocket Pool fees.
- They are always in control of their own node.
- They have a say on new proposals on the Rocket Pool network.

Trusted Node Operator

Rocket Pool user deposits are always assigned to Staking Node Operators first; any surplus is assigned to Trusted Node Operators. Trusted Node Operators are a backup; they ensure the network can continue to onboard new users if there is no capacity with Staking Node Operators. Trusted Node Operators are not required to match user deposit stakes.

Minipools

A minipool is a type of smart contract that is created by Rocket Pool when a node operator makes a deposit of their own ether on their node. These contracts are used to pool ether from various stakers until they reach enough to stake with Casper. Node operators 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 Rocket Pool.

RPL Token

RPL is required if you wish to act as a node operator in the Rocket Pool network. RPL is ***not*** required as a user to stake ether on the Rocket Pool network.

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

In Rocket Pool 2.0, the RPL: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

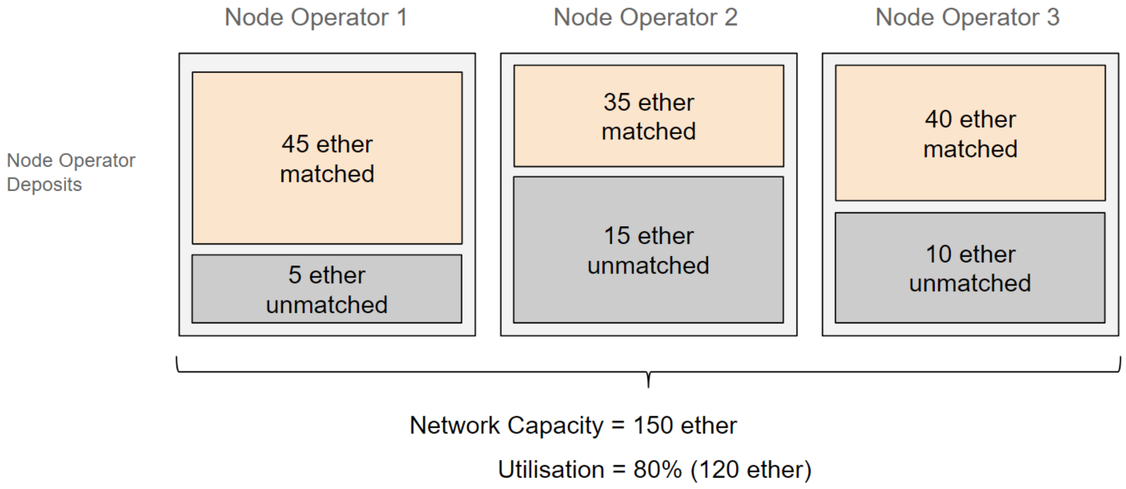
The Rocket Pool network requires enough node operators to cover spikes in demand but not so many as to suffer from underutilised nodes (see Too Many Nodes attack).

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

- Increasing capacity when needed, by incentivising node operators to join.
- Allowing for an optimal network utilisation which still has capacity for spikes in usage.

When a node operator joins the Rocket Pool network they are required to deposit ether into a smart contract. Staking user deposits are then matched with this ether and deposited into a minipool for staking. The percentage of node operator ether that has been matched can be thought of as network utilisation. As demand increases, more node operator ether 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 RPL per ether they are depositing. This RPL:ETH ratio is now dynamic and is dependent on the network utilisation, i.e:

- If the network has plenty of capacity, then node operators need more RPL to join.
- If the network is reaching capacity, then node operators need less RPL to join.

Subsequently, node operators 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 required from Casper reduced from 1,500 ether to 32 ether, a new attack vector was identified. A whale with a lot of ether and RPL could prevent current node operators from ever receiving new users by adding a large number of underutilised nodes to the network. To ensure fairness and decentralisation, node operators 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, RPL requirements make it progressively more expensive to add underutilised nodes to the network.

Incentivising Nodes: Earning extra income as a smart node in the Rocket Pool 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 Rocket Pool smart contracts. Imagine that a huge exchange decided to use Rocket Pool 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 RPL formula substantially reduces the amount of RPL required when the network nears capacity, reducing the cost to join the network and thus incentivising new node operators to join before it goes back up.

The RPL:ETH Formula

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

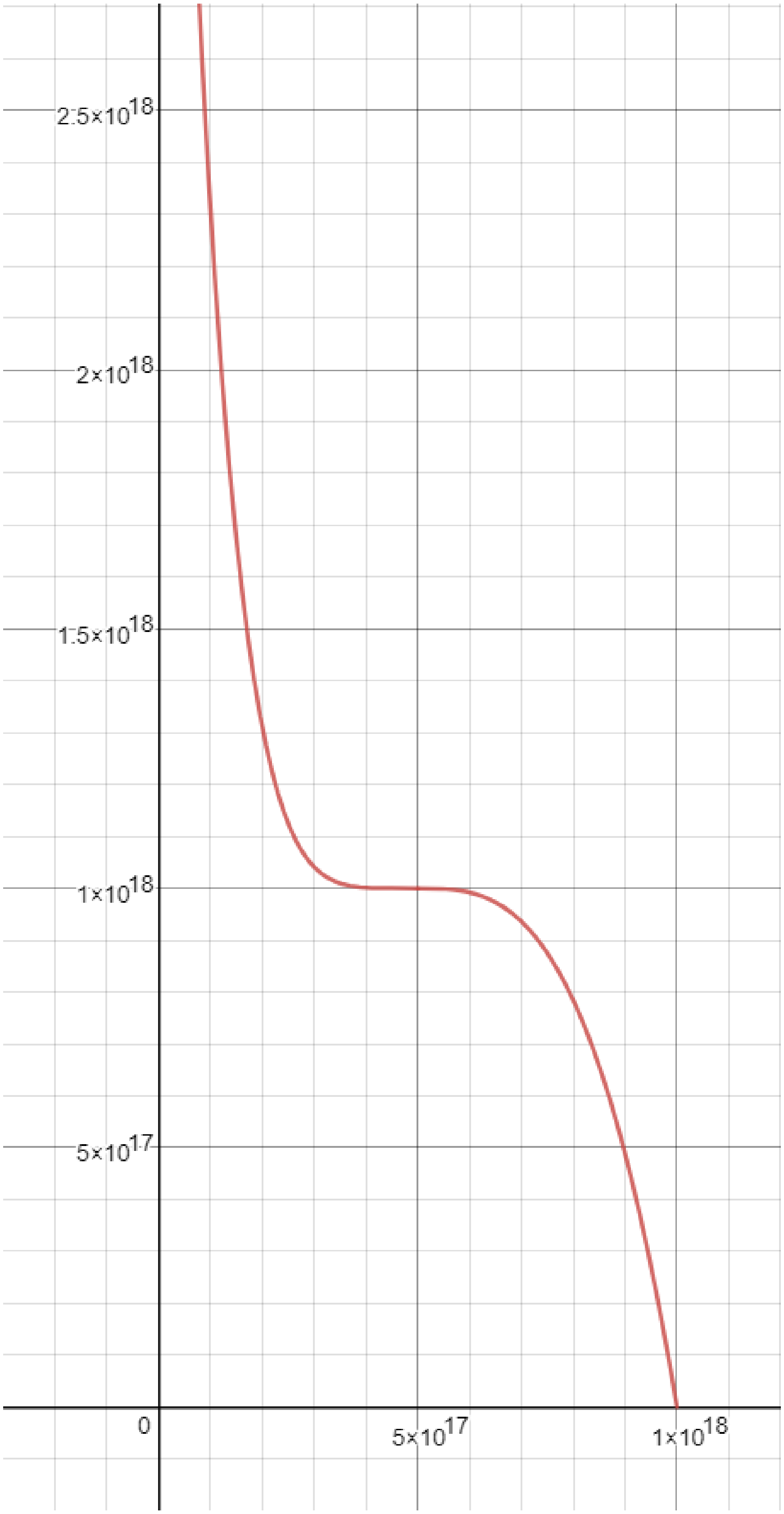


Figure 1: X axis: Network utilisation (0 - 100%); Y axis: RPL per ETH required

This determines the current RPL:ETH ratio where:

- x axis represents the current utilisation of the network (0-100%)
- y axis represents the amount of RPL required to stake 1 ether
- Ratio sweet spot is approximately 1:1 between 40%-70%.
- If the network is underutilised, RPL required increases (sharply under 15%). This prevents several attack scenarios.
- If the network is near capacity, RPL required reduces significantly to prompt new node operators to come online and increase capacity.

This ratio applies for each staking duration available. Consequently, if there are more nodes that support staking for 6 months than 12 months, it will require less RPL to stake ether for the latter due to there being less capacity for that staking duration.

The purpose of this formula is to make sure the network always operates at close to ***40-70%*** capacity, a common goal for Wide Area Networks (WAN) that allows for peak usage jumps while still making sure the network isn't wasting resources by being underutilised.

Some examples:

A new node operator wishes to deposit 100 ether in their smart node account for staking. The table below shows what the RPL cost will be for different levels of network utilisation.

Network Utilisation	RPL:ETH	RPL Cost	Comments
5%	3.5 RPL : 1 ether	350 RPL	The network is highly underutilised given its capacity.
15%	1.70 RPL : 1 ether	170 RPL	The network is underutilised given its capacity.
40% -70%	1 RPL : 1 ether	100 RPL	The network utilisation is good given its capacity.
85%	0.65 RPL : 1 ether	65 RPL	The network utilisation is high given its capacity.
95%	0.25 RPL : 1 ether	25 RPL	The network urgently needs more capacity to on-board new users.

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 Rocket Pool 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 node operators 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 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

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

Staking Users

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

- The user does not have the minimum 32 ether 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 Rocket Pool infrastructure to earn interest for their users.
- The user simply wishes to use Rocket Pool because it is convenient.

Rocket Pool allows staking users to earn interest on as little as 1 ether 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 Rocket Pool website.
- Via the API through a group (exchange, wallet provider etc).
- Or directly to Rocket Pool's smart contracts.

Easy as that!

Groups

Staking users in Rocket Pool 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 for a fixed term, without worrying about maintaining extensive staking infrastructure – just plug and play.

Groups can be registered with Rocket Pool 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 Rocket Pool network, which are collected on this contract. No one but the group owner can access these fees, including Rocket Pool.

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

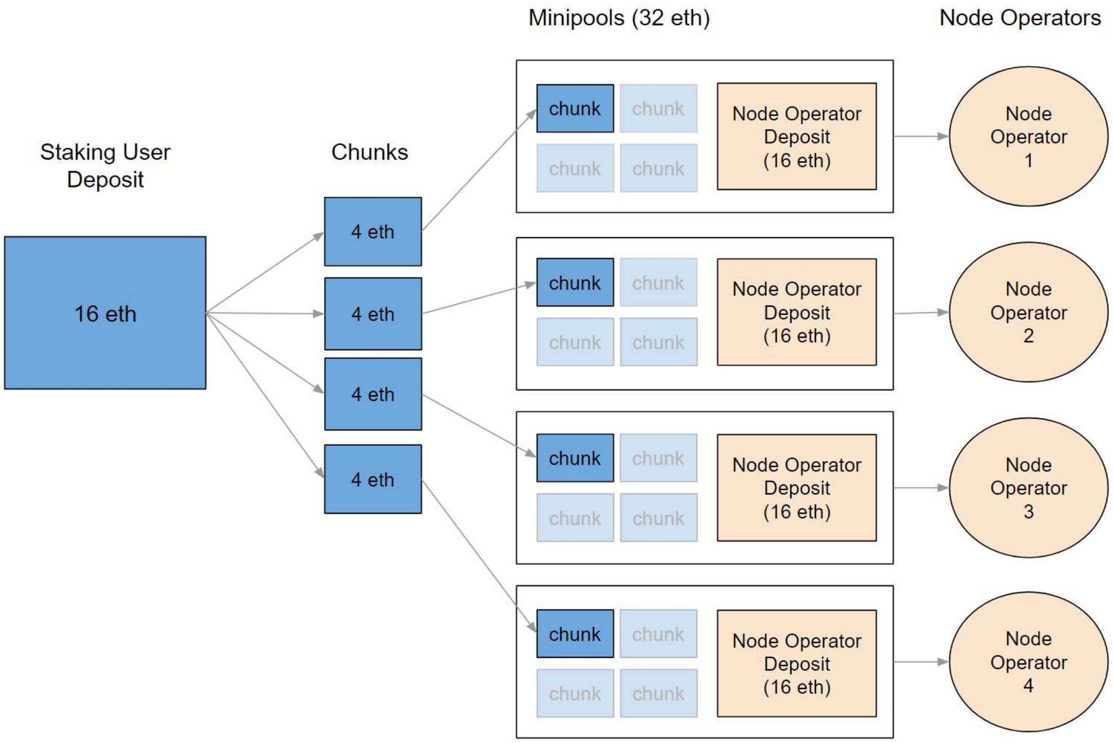
Rocket Pool's staking pool is itself a group with its own integration contracts sitting on top of the Rocket Pool API; we use the same decentralised staking infrastructure we offer every other group.

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:

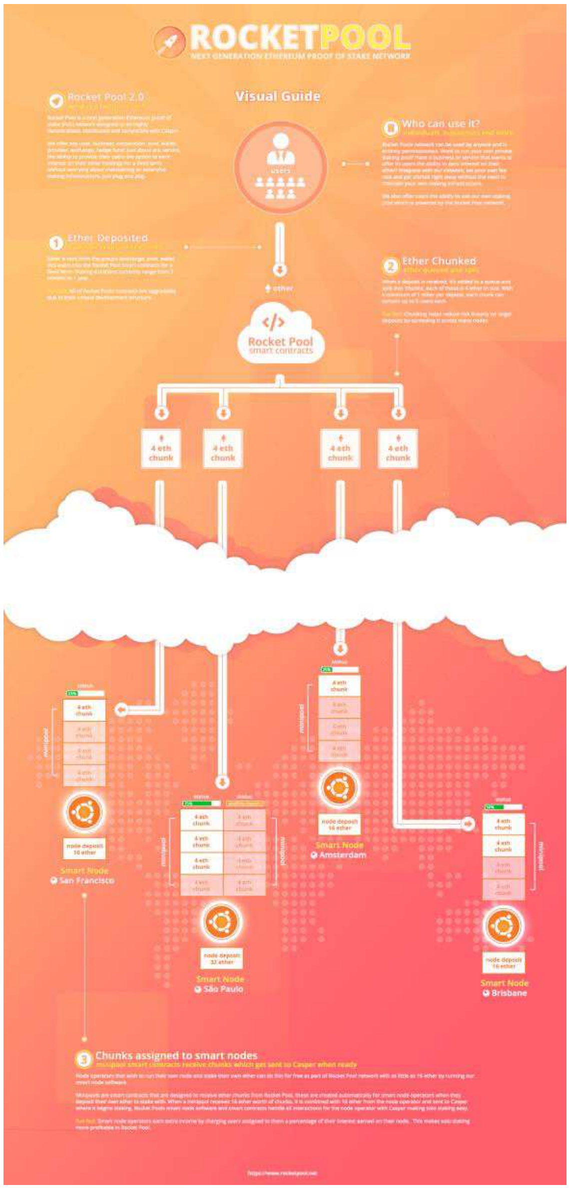
1. A staking user deposits ether into Rocket Pool (for example, 16 ether).
2. The deposit is broken up into 4 eth chunks.
3. Each chunk is allocated to a different minipool, which is assigned to a different node operator.



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

Infographic

To further explain this concept, we've created this infographic below. The full version can be seen here.



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 node operators in the network; the smaller part is a fee collected by Rocket Pool. All fees are locked in when the user begins staking; they will not change for that user over their staking duration.

Node Operators

There are major benefits to node operators who supply a node and stake their ether with the Rocket Pool network, rather than solo staking.

Benefits

Users / businesses will be able to stake using their own node in the Rocket Pool network with as little as **16 ether**. After adding their node to our network and installing the Rocket Pool smart node software, it will be able to stake ether, and will be assigned an equal amount of ether from Rocket Pool'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 in Rocket Pool than if they ran their own staking node outside of the Rocket Pool network.

Setup

This is a technical walkthrough of how the node staking process works for node operators. These users will have two options for setting up a Rocket Pool node:

Installable Package Approach

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

This will include at least:

- Ethereum mainnet client (Geth, Parity, etc)
- Ethereum beacon chain client

On installation it will register a Rocket Pool service that will:

- Automatically start on server restart
- Automatically restart on application crash
- Be globally available

This option should be preferred by node operators not hosting a node in the cloud.

DevOps Playbook

An option for node operators 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 Rocket Pool smart contracts.

Staking Process

The staking process is automatically handled by Rocket Pool's smart node software. This is a high-level walkthrough of how the deposit and staking process works for node operators on their server.

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

```
`$> rocketpool deposit $amount $duration`
```

Where `$amount` represents a numeric value of the amount of ether they wish to stake; they are not charged a fee on this ether 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 for a node registration is 16 ether.
2. It determines the amount of RPL needed to register the node's deposit based on the current network utilisation. The RPL amount is calculated using the RPL:ETH formula above.
3. If there is sufficient RPL in the node's mainnet etherbase account, the operator is prompted to transfer the RPL with the ether deposit. If not, the operator can transfer the RPL within 24hrs from any account to lock in the RPL:ETH ratio received.
4. It registers the node operator with the Rocket Pool network by creating a new smart contract for the node's deposits of ether / RPL. The current RPL: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/RPL if needed. If no RPL was deposited, the operator is informed they have 24hrs to deposit the RPL amount or the ratio will be recalculated.

Determining Node Operator Reward

Node operators can earn extra ether in addition to interest awarded by Casper by staking their ether with the Rocket Pool network and running a smart node. The reward is calculated as a percentage of the ether earned on that node by staking users assigned by Rocket Pool. This is an incentive to provide a stable and highly available smart node. For example, if 16 ether is assigned to a node operator from Rocket Pool'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 node operators charge staking users is determined by all participating node operators 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. Rocket Pool'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 node operators:

- Provides a level playing field for all node operators 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 node operators. Too low and node operators make less income; too high and staking users are less willing to participate, affecting node operators' future income.

Rocket Pool Improvement Process (RPIP)

Motivation

The Rocket Pool Improvement Process (RPIP) gives the Rocket Pool community influence over the Rocket Pool development roadmap.

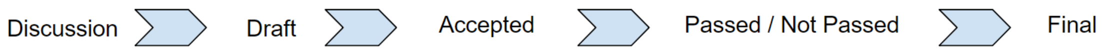
It will do this by:

- 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 Rocket Pool implementation team.

The Rocket Pool Improvement Process is not an attempt to decentralise Rocket Pool 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 Rocket Pool community. Once the proposal has proceeded through the Rocket Pool improvement process and reached the final status, it is ready to be incorporated into the Rocket Pool roadmap.



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

Discussion

Anyone in the Rocket Pool community can propose an improvement to the Rocket Pool network. To gather community feedback on the new proposal, it will be submitted to a public discussion site such as a dedicated Discourse or Rocket Pool 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 or not the improvement gets implemented. All node operators currently staking in the Rocket Pool network are eligible to vote on the proposal. Their vote weight is equal to the amount of ether they are currently staking.

The voting process is a two-phase voting scheme: commit & reveal. During the commit period, node operators 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 Rocket Pool improvement proposals, the node operator uses the rpip command:

```
$> rocketpool rpip
```

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

```
$> rocketpool rpip vote $rpipID $yesNo $comments
```

After 28 days of committing votes, the process switches into a reveal phase. In the reveal phase, node operators can no longer cast votes. The Rocket Pool 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:

```
$> rocketpool rpip 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 Rocket Pool 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 Rocket Pool network. Some of the common commands will be:

Registration

`$> rocketpool register Australia/Brisbane` – Registers the node as part of the Rocket Pool network. An optional timezone code can be supplied to help provide a visual map on the Rocket Pool 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

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

Deposit - Manual

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

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

`$> rocketpool deposit rpl $amount` – Sends RPL from the node's etherbase account to the node's registration contract that was created when the node registered itself with Rocket Pool.

`$> rocketpool deposit ether $amount` – Sends ETH from the node's etherbase account to the node's registration contract that was created when the node registered itself with Rocket Pool. Requires a reservation and sufficient RPL balance on the node's registration contract to activate the deposit reservation.

Deposit – Misc

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

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

Withdrawal

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

`$> rocketpool withdraw rpl $amount` - Wwithdraws RPL from the node's registration contract back to the node's etherbase account.

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

Fees

`$> rocketpool 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.

`$> rocketpool 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

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

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

Voting

`$> rocketpool rpip` - Displays a list of the active RPIPs from the RPIP smart contract.

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

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

`$> rocketpool` - No args passed, shows the cool Rocket Pool 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, Rocket Pool is committing to subjecting its platform to a comprehensive security audit before launching on the Ethereum mainnet.

All Rocket Pool 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 Rocket Pool’s future beta release. All audit results and applicable post mortem actions will be made publicly available.

Team

David Rugendyke | Project Founder, CTO

David has over 18 years commercial experience developing high end web applications, has been featured twice in Australia Personal Computer (APC) magazine for projects he’s built, and is currently committed to working on Rocket Pool full time.

Thriving on a challenge and with a long list of software projects behind him, David has acquired a very proficient knowledge of Cryptocurrency and DevOps as well, a unique combination of skills that lead to the creation of Rocket Pool, a project with scope spread across several of these unique fields.

David also loves to homebrew beer and when not working on Rocket Pool, can be found enjoying a beer or brewing another all grain batch over the course of a day. David’s favourite beer style is a good American Wheat Ale, followed closely by an IPA.

Jake Pospischil | Senior Developer

Jake is a full-stack developer with over 10 years' professional experience. He has a background in visual design and started his career in front-end development, but quickly progressed to building robust back-ends and provisioning servers.

Jake has had a keen interest in Cryptocurrency for several years and has been developing Ethereum applications since 2017. He is also proficient with a number of web technologies including MVC frameworks, reactive view frameworks, and various modern build pipelines.

When he's not writing code, Jake can usually be found at the gym or sitting around a table playing D&D with his friends.

Darren Langley | Senior Developer

Darren has over 16 years of software development experience. As a technical lead & architect, he has built exciting digital products for government, financial services, and professional services.

Darren has considerable Ethereum experience and has delivered several proof-of-concept projects for the public service and finance industries. With Rocket Pool, he is on the cutting-edge of cryptocurrency and blockchain development.

When Darren is not sitting at a laptop, he is usually at the beach with his family.