Introduction

Lido V2 protocol upgrade adds support forEthereum Withdrawals and introduces additional responsibilities for Node Operators.

Lido Withdrawals are happening in four stages:

1. stETH holders request a withdrawal
2. Lido Oracles decide which Lido validators should be exited to fulfil the request and publish the list on-chain
3. Lido Node Operators exit these validators
4. stETH holders claim their ETH

This means that Node Operators need a way to react quickly to protocol requests.

The suggested method is to generate and sign exit messages ahead of time which will be sent out when needed by a special new daemon called the Validator Ejector.

To understand for which validators to generate and sign exit messages, another new app called Keys API is introduced.

Requirements

First, is running Lido tooling required?

Lido tooling is not required to use, but is recommended.

The only requirement for Node Operators is to exit their validators in time after requested by the protocol.

For details, check outValidator Exits Policy andResearch Forum Discussion .

Lido Tooling

Keys API (KAPI for short)

KAPI is a service which stores and serves up-to-date information about Lido validators.

It provides a very important function: it provides two endpoints, using which a Node Operator understands for which validators to generate and sign exit messages in advance.

Under the hood, KAPI also automatically filters out validators which are exited already or are currently exiting.

Validator Ejector (Ejector for short)

Ejector is a daemon service which monitorsValidatorsExitBusOracle events and initiates an exit when required.

In messages mode, on start, it loads exit messages in form of individual .json files from a specified folder or an external storage and validates their format, structure and signature.

It then loads events from a configurable amount of latest finalized blocks, checks if exits should be made and after that periodically fetches fresh events.

In webhook mode, it simply fetches a remote endpoint when an exit should be made, allowing to implement JIT approach by offloading exiting logic to an external service and using the Ejector as a secure exit events reader. Edit this page Previous Execution Layer Rewards Configuration Next General Information