Oracle Operator Manual

This document is intended for those who wish to participate in the Lido protocol as entity that runs - an entity who runs a daemons synchronizing state from Beacon Layer to Execution Layer of the protocol. Due to the lack of native communication between these two networks, Lido employs a network of oracles to synchronize the system at regular intervals.

TL;DR

1. Generate an Ethereum address.
2. Launch and sync anarchive
3. (archive data for at least 2 weeks) Execution Layer node with JSON-RPC endpoint enabled.
4. Launch and sync anarchive
5. Consensus Layer node with API endpoint enabled.
6. Launch and sync aKeys API Service
7. .
8. Launch the accounting and ejector modules of Oracle.
9. [Optional
10.] Add alerts to Oracle's Prometheus metrics.
11. In case of mainnet share your address and intention to join the Oracle set with public. You need to publish it on Twitter and also write a message with a twitter link under Onboarding post onthe Research forum
12. .
13. Propose your Oracle's ethereum address to Lido Team to vote on your address being added to the Oracle Members.

Intro

The Lido Oracle mechanism comprises three main components. The first component is the Oracle smart-contract suite, which receives update reports from the oracles and passes them on to the Lido contract to execute the necessary actions based on the reported changes. The second component is the off-chain oracle daemon, run by each oracle node and responsible for monitoring the protocol state and generating update reports. The third component is the network of computer nodes that run by oracle member, which collectively provide the necessary information to the Oracle smart contract to calculate the new state of the protocol.

Based on the update reports received from the oracles, the Lido smart contract performs state transitions such as updating user balances, processing withdrawal requests, and distributing rewards to node operators. Thus, the Lido Oracle mechanism acts as a synchronization device that bridges the protocol across the execution and consensus layers. It ensures that the protocol is updated in a timely and accurate manner and allows for smooth and efficient operation of the entire Lido system.

The two core contracts in the Lido Oracle suite are calledAccountingOracle andValidatorsExitBus . Together, these contracts collect information submitted by oracles about the state of validators and their balances, the amount of funds accumulated on protocol vaults, the number of withdrawal requests the protocol is able to process, and the validators are expected to be voluntary exited to finalize more withdrawal requests. This information is then used for these crucial processes:

- rebasing user balances,
- distributing node operator rewards,
- processing withdrawal requests,
- making decision which validators should initiate voluntary exit,
- distributing stake,
- putting the protocol into the bunker mode.

Oracle phases

In order to send the report data by the oracle operator to bothAccountingOracle andValidatorsExitBusOracle , it is necessary that:

- this operator participates in the oracle committee, and
- a consensus for the corresponding report must be reached

A process of sending the report data can be divided into 3 major stages:

Phase 1. Submitting a report hash and reaching consensus

At the first stage, the oracles operators collect a report for a certainrefSlot and send the hash to theHashConsensus contract.

The diagram below shows:ReportProcessor -AccountingOracle orValidatorsExitBusOracle contract.HashConsensus - a contract which manages oracle members committee and allows the members to reach consensus on the particular data hash for each reporting frame.

You can read more about HashConsensushere .

Phase 2. Submitting a report data

When the consensus is reached, one of the oracles operators submits report data and triggers the core protocol state update (including the token rebase, distribution of node operator rewards, finalization of withdrawal requests, and deciding whether to go in the bunker mode) or emitsValidatorExitRequest events to inform node operators about new voluntary exit requests needed to perform.

Phase 3. Submitting a report extra data

This step is required forAccountingOracle , involving reward distribution for staking modules on this phase.

Committee membership

The current Oracle set consists of 9 participants:

- Chorus One0x140bd8fbdc884f48da7cb1c09be8a2fadfea776e
- Staking Facilities0x404335bce530400a5814375e7ec1fb55faff3ea2
- stakefish0x946d3b081ed19173dc83cd974fc69e1e760b7d78
- P2P Validator0x007de4a5f7bc37e2f26c0cb2e8a95006ee9b89b5
- Rated0xec4bfbaf681eb505b94e4a7849877dc6c600ca3a
- bloXroute0x61c91ECd902EB56e314bB2D5c5C07785444Ea1c8
- Instadapp0x1ca0fec59b86f549e1f1184d97cb47794c8af58d
- Kyber Network0xA7410857ABbf75043d61ea54e07D57A6EB6EF186
- ChainLayer0xc79F702202E3A6B0B6310B537E786B9ACAA19BAf

The quorum is 5/9. This means that the report finalization can only occur when there are 5 identical reports from the 5 different oracle members.

SeeExpansion of Lido on Ethereum Oracle setpost for more details.

Prerequisites

Execution Client Node

To prepare the report, Oracle fetches up to 10 days old events, makes historical requests for balance data and makes simulated reports on historical blocks. This requires anarchive execution node. Oracle needs two weeks of archived data.

Client Tested Notes Geth

--gcmode=archive

--syncmode=snap

OR

--gcmode=archive

--syncmode=full Nethermind

Not tested yet Besu

Use --rpc-max-logs-range=100000

--sync-mode=FULL

--data-storage-format="FOREST"

--pruning-enabled

--pruning-blocks-retained=100000

params Erigon

Use --prune=htc

--prune.h.before=100000

--prune.t.before=100000

--prune.c.before=100000

params

Consensus Client Node

To calculate some metrics for bunker mode Oracle needsarchive consensus node.

Client Tested Notes Lighthouse

Use--reconstruct-historic-states param Lodestar

Not tested yet Nimbus

Not tested yet Prysm

Use --grpc-max-msg-size=104857600

--enable-historical-state-representation=true

--slots-per-archive-point=1024

params Teku

Use --data-storage-mode=archive

--data-storage-archive-frequency=1024

--reconstruct-historic-states=true

params

Keys API Service

This is a separate service that uses Execution Client to fetch all lido keys. It stores the latest state of lido keys in database.

Lido Keys API repository.

The oracle daemon

The Oracle daemon is a Python application that contains two modules:

- Accounting module
- Ejector module

The oracle source code is available athttps://github.com/lidofinance/lido-oracle .

Modules fetch the reportable slot, and if this slot is finalized, calculate and send the report to AccountingOracle and ExitBusOracle smart contracts.

Environment variables

The oracle daemon requires the following environment variables:

Required

- EXECUTION_CLIENT_URI
-
  - list of Execution Client uris separated with comma. The second and next uris will be used as fallback.
- CONSENSUS_CLIENT_URI
-
  - list of Consensus Client uris separated with comma. The second and next uris will be used as fallback.
- KEYS_API_URI
-
  - list of Key API client uris separated with comma. The second and next uris will be used as fallback.
- LIDO_LOCATOR_ADDRESS
-

- Lido Locator smart contract address.

Optional

One of:

- MEMBER_PRIV_KEY
-
  - Private key of the Oracle member account.
- MEMBER_PRIV_KEY_FILE
-
  - A path to the file contained the private key of the Oracle member account.

Full list could be foundhere .

Lido Locator address

Mainnet 0xC1d0b3DE6792Bf6b4b37EccdcC24e45978Cfd2Eb

Goerli 0x1eDf09b5023DC86737b59dE68a8130De878984f5

Running the daemon

Startup accounting module

docker run -d --name lido-oracle-accounting \ --env "EXECUTION_CLIENT_URI=EXECUTION_CLIENT_URI" \ --env "CONSENSUS_CLIENT_URI=CONSENSUS_CLIENT_URI" \ --env "KEYS_API_URI=KEYS_API_URI" \ --env "LIDO_LOCATOR_ADDRESS=LOCATOR_ADDRESS" \ --env "MEMBER_PRIV_KEY=MEMBER_PRIV_KEY" \ lidofinance/oracle@ accounting Startup ejector module

docker run -d --name lido-oracle-ejector \ --env "EXECUTION_CLIENT_URI=EXECUTION_CLIENT_URI" \ --env "CONSENSUS_CLIENT_URI=CONSENSUS_CLIENT_URI" \ --env "KEYS_API_URI=KEYS_API_URI" \ --env "LIDO_LOCATOR_ADDRESS=LOCATOR_ADDRESS" \ --env "MEMBER_PRIV_KEY=MEMBER_PRIV_KEY" \ lidofinance/oracle@ ejector Latest image hash https://docs.lido.fi/guides/tooling/#oracle

This will start the oracle in daemon mode. You can also run it in a one-off mode, for example if you'd prefer to trigger oracle execution as acron job. In this case, set theDAEMON environment variable to 0.

Metrics and Alerts

How to set up alerts and details about metrics could be foundhere . Edit this page Previous Tooling Setup & Configuration Next Accounting oracle