Validator Ejector

Introduction

Ejector is a daemon service which monitorsValidatorsExitBusOracle events and sends out stored exit messages when necessary. It allows Node Operators to generate and sign exit messages ahead of time, which will be sent out by the Ejector when the Protocol requests an exit to be made.

On start, it loads exit messages from a specified folder in form of individual.json files and validates their format, structure and signature. Then, it loads events from a configurable amount of latest finalized blocks, checks if exits should be made and after that periodically fetches fresh events.

Requirements

Hardware

- 2-core CPU
- 1GB RAM

Nodes

- Execution Node -Full node required
- Consensus Node

Software

Using Docker:

Docker + docker-compose.

Running directly or using for message encryption:

Node.js 16.

Exit Messages

Ejector loads and validates exit messages on start. This means that any changes to the messages folder (eg new exit messages) require a restart of the app to be picked up.

Ejector accepts messages in three formats:

Generic Format

{ "message": { "epoch": "123", "validator_index": "123" }, "signature": "0x123" }

ethdo Output Format

{ "exit": { "message": { "epoch": "123", "validator_index": "123" }, "signature": "0x123" }, "fork_version": "0x123" }

Encrypted Format

{ "version": 4, "uuid": "123abc-123abc-123abc", "path": "", "pubkey": "", "crypto": { "kdf": { "function": "pbkdf2", "params": { "dklen": 123, "c": 123, "prf": "hmac-sha256", "salt": "123abc" }, "message": "" }, "checksum": { "function": "sha256", "params": {}, "message": "123abc" }, "cipher": { "function": "aes-128-ctr", "params": { "iv": "123abc" }, "message": "123abc" } } }

Encrypting Messages

It is highly advised that after exit message are generated and signed, they should be encrypted for storage safety. Ejector will decrypt files on start by looking up the password inMESSAGES_PASSWORD environment variable.

Exit messages are encrypted and decrypted by Ejector following theEIP-2335 spec.

Ejector is bundled with a small, easy to use encryption script.

Encryption using Ejector - Source Code

1. Clone repository:

git clone https://github.com/lidofinance/validator-ejector.git cd validator-ejector 1. Create.env 2. file with encryption password or pass before the command:

MESSAGES_PASSWORD=password 1. Copy JSON exit message files toencryptor/input 2. Runyarn & yarn encrypt 3. Encrypted exit message files will be saved toencryptor/output

Encryption using Ejector - Docker

Ejector is bundled with encryptor script inside, so you can run it using the same Docker image:

docker run \ -e MESSAGES_PASSWORD=secret \ -v /full/path/to/input:/app/encryptor/input/ \ -v /full/path/to/output:/app/encryptor/output/ \ lidofinance/validator-ejector@sha256\ node /app/dist/encryptor/encrypt.js You can find a recommended version's hashhere .

For platforms with a different architecture but with emulation/transpilation support eg macOS on M processors, additionally specify:

--platform linux/amd64

Env Variables

EXECUTION_NODE

Address of the Execution Node.

CONSENSUS_NODE

Address of the Consensus Node.

LOCATOR_ADDRESS

Address of theLidoLocator contract:Goerli /Mainnet

STAKING_MODULE_ID

ID of theStakingRouter contract module.

Currently, it has only one module (NodeOperatorsRegistry), it's id is1 .

OPERATOR_ID

You can find it on the Operators Dashboard (#123 on the operator card)Goerli /Mainnet

MESSAGES_LOCATION

Location from which to load.json exit messages from.

When set, messages mode will be activated. Not needed if you are using the Ejector in webhook mode.

For example,/messages in Docker or simplymessages if running directly for local files.

External storage bucket url is also supported for AWS S3 and Google Cloud Storage:

- s3://
- for S3
- gs://
- for GCS

Authentication setup:GCS ,S3 .

VALIDATOR_EXIT_WEBHOOK

Endpoint to fetch when an exit has to be made. Allows to implement JIT approach by offloading exiting logic to an external service and using the Ejector as a secure exit events reader.

When set, webhook mode will be activated. Not needed if you are using the Ejector in messages mode.

On the endpoint, JSON will be POSTed with the following structure:

{ "validatorIndex": "123", "validatorPubkey": "0x123" } 200 response will be counted as a successful exit, non-200 as a fail.

ORACLE_ADDRESSES_ALLOWLIST

JSON array of Lido Oracle addresses, from which only report transactions will be accepted.

You can get a list from the Aragon appGoerli /Mainnet

Format:

["0x123", "0x123"]

MESSAGES_PASSWORD

Password to decrypt encrypted exit messages with on app start.

MESSAGES_PASSWORD_FILE

Alternative toMESSAGES_PASSWORD . Path to a file with password inside to decrypt exit messages with. If used, MESSAGES_PASSWORD (not MESSAGES_PASSWORD_FILE) needs to be added to LOGGER_SECRETS in order to be sanitized

BLOCKS_PRELOAD

Amount of blocks to load events from on start.

Suggested to include in your env variables, but to be left at default 50000 value (~7 days of blocks).

In case your Ejector will be down due to an emergency, this value can be tweaked to let the Ejector load a higher amount of blocks on start.

HTTP_PORT

Port for serving metrics and a health check endpoint. Default is 8989.

RUN_METRICS

Enable withtrue to serve Prometheus metricsfull list .

Will be served onHOST:HTTP_PORT/metrics .

Highly advised for monitoring and alerting.

RUN_HEALTH_CHECK

Enabled by default, disabled withfalse . Highly recommended to monitor this endpoint.

Will be served onHOST:HTTP_PORT/health .

LOGGER_LEVEL

Recommended to set toinfo (default), can be changed todebug in case of issues for easier debugging.

LOGGER_FORMAT

Format of logs,simple by default, but can be set tojson to be easily parseable byLoki , for example.

LOGGER_SECRETS

Env var names or exact values which should be replaced in logs, in JSON array of strings format.

Advised to include your MESSAGES_PASSWORD, EXECUTION_NODE, and MESSAGES_PASSWORD:

["MESSAGES_PASSWORD", "EXECUTION_NODE", "CONSENSUS_NODE"] Notice: make sure quotes are copied correctly if copying this sample.

DRY_RUN

Allows to test the app withtrue without actually sending out exit messages.

Use with caution!

Make sure to set tofalse or completely leave it out in production.

Advanced Parameters

Please don't use unless suggested by a Lido contributor.

- BLOCKS_LOOP - 900 (3 hours of blocks) - Amount of blocks Ejector looks behind on wake in polling jobs.
- JOB_INTERVAL - 384000 (1 epoch) - Time for which Ejector sleeps between jobs.
- DISABLE_SECURITY_DONT_USE_IN_PRODUCTION - false - Set totrue
- to skip security checks, for example if Exit Bus Consensus contract was changed after the Ejector was unable to exit validators eg was switched off.

Running

Source Code

1. Clone repository:

git clone https://github.com/lidofinance/validator-ejector.git cd validator-ejector 1. Create exit messages folder, for example locallymkdir messages 2. Put exit message files in the messages folder. 3. Copy env sample filecp sample.env .env 4. Fill environment variables in.env 5. file. 6. Run

yarn yarn build yarn start

Docker with docker-compose

1. Create root folder for Ejector, cd into that folder.
2. Create exit messages foldermkdir messages
3. Put exit message files in messages folder.
4. Copy env filecp sample.env .env
5. Fill environment variables in .env file.
6. Createdocker-compose.yml
7. file using the following template:

https://github.com/lidofinance/validator-ejector/blob/develop/docker-compose.yml

1. Rundocker-compose up
2. ordocker-compose up -d
3. to start in detached mode (in background).

Check Ejector is working

1. Ensure there are no errors in logs and no restarts.
2. Verify that config logged on start is correct in logs.
3. If you have put presigned messages in the messages folder, make sure Loaded Messages count is greater than0
4. .
5. Ensure you can seeJob started
6. andJob finished
7. lines in logs.

Example of correct operation logs:

info: Application started, version 1.0.0
{"EXECUTION_NODE":"","CONSENSUS_NODE":"","LOCATOR_ADDRESS":"0x123","STAKING_MODULE_ID":"1","OPERATOR_ID":"0","MESSAGES_LOCATION":"messages","ORACLE_ADDRESSE
["0x123"],"MESSAGES_PASSWORD":"","BLOCKS_PRELOAD":190000,"BLOCKS_LOOP":64,"JOB_INTERVAL":384000,"HTTP_PORT":8989,"RUN_METRICS":true,"RUN_HEALTH_CHECK":true,"DR
info: Loading messages from messages info: Loaded 123 messages info: Validating messages info: Starting, searching only for requests for operator 0 info: Loading initial events for 190000 last blocks
info: Job started {"operatorId":"0","stakingModuleId":"1","loadedMessages":123} info: Resolved Exit Bus contract address using the Locator {"exitBusAddress":"0x123"} info: Resolved Consensus
contract address {"consensusAddress":"0x123"} info: Fetched the latest block from EL {"latestBlock":12345} info: Fetching request events from the Exit Bus
{"eventsNumber":190000,"fromBlock":12345,"toBlock":12345} info: Loaded ValidatorExitRequest events {"amount":0} info: Handling ejection requests {"amount":0} info: Job finished info: Starting 384
seconds polling for 64 last blocks

What if something is wrong?

1. Make sure configuration is correct.
2. Make sure you are on the recommended Docker image SHA hash or version if running directly.
3. Check if Nodes are synced and are working correctly.
4. Restart the app.
5. Start the app with LOGGER_LEVEL=debug env variable and contact Lido devs with logs to investigate the problem.

Additional Resources

Validator Ejector GitHub Repository (Open Source)https://github.com/lidofinance/validator-ejector

Lido Withdrawals: Automating Validator Exits - parts can now be outdatedhttps://hackmd.io/@lido/BkxRxAr-o

Ejector Logic Spec - parts can now be outdatedhttps://hackmd.io/@lido/r1KZ4YNdj Edit this page Previous Tooling Overview Next Keys API