General Information

What Are Exit Messages?

To initiate a validator exit, anexit message needs to be generated, signed and submitted to a Consensus Node.

It looks like this:

{ "message": { "epoch": "123", "validator_index": "123" }, "signature": "0x123" } After it's generated, it is signed using the validator BLS key which needs to be exited and asigned exit message is formed.

Pre-sign or Not to Pre-sign

In short, it's a well balanced solution allowing to achieve meaningful automation without sacrificing security or decentralisation.

You can find the reasoning behind the suggested approach in theLido Withdrawals: Automating Validator Exits RFC.

However, if you have existing tooling in place to create and send out exit messages, you can use webhook mode of the Ejector which will call an endpoint in order to initiate a validator exit.

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.

If it's not enough, you'll have to fork the Ejector or monitorValidatorExitRequest events of theValidatorsExitBusOracle in your tooling.

Example in the Ejector

How Many Keys to Pre-sign

Node Operators should pre-sign an amount or a percentage of validators which they are comfortable with managing. Smaller amounts means more frequent refills and vice versa.

For Withdrawals preparations, for example, exits for 10% of the validators pre-signed is suggested. After that, it will depend on the Withdrawals demand and Node Operator preferences.

How to Understand Which Keys to Pre-sign

Using KAPI (recommended)

First endpoint returns a list of validators for a specific Node Operator, for which to generate and sign exit messages next:

/v1/modules/{module_id}/validators/validator-exits-to-prepare/{operator_id}

Returns data:

[{ "validatorIndex": 123; "key": "0x123"; }] Additionally, there is also a second endpoint which calculates the same data, but returns ready to sign exit messages:

/v1/modules/{module_id}/validators/generate-unsigned-exit-messages/{operator_id}

danger Make sure to visually inspect the returned data as a precaution as you will be signing it. You can find the expected format in theWhat Are Exit Messages section. Returns data:

[{ "validator_index": "123"; "epoch": "123"; }] Furthermore, both endpoints allow for additional configuration via query parameters:

- percent
  - Percent of validators to return data for. Default value is 10.
- max_amount
  - Number of validators to return data for. If validator number is less than the specified amount, all validators are returned.

info Note: Only one parameter is active at a time. If both are provided,percent has a higher priority. KAPI will automatically

filter out validators which are exited already or are currently exiting, so one call to the KAPI is all that's needed.

Manually

If your validator signing keys are stored in generation order, you can simply start exit generation and signing from the oldest keys since the exit algorithm is deterministic and will choose oldest keys first for each Node Operator.

However, for each batch you'll need to either track the last key exit message was generated for or query validator statuses on the Consensus Node to understand where to start next.

Storing Signed Exit Messages

Storing signed exit messages as-is is discouraged. If an attacker gains access to them, they will simply submit them, which will exit every validator for which you had exit messages.

It's recommended to encrypt the messages, for example using the encryption script provided with the Ejector.

How to Use the Ejector Encryptor

You can also check out thesource code and integrate it in your own tooling if needed.

Ejector automatically decryptsEIP-2335 -encrypted files on app start.

How to Initiate a Validator Exit

Signed messages are sent to a Consensus Node via the/eth/v1/beacon/pool/voluntary_exits endpoint in order to initiate an exit.

The Ejector will do this automatically when necessary.

If you don't run the Ejector, you'll have to do this manually or develop your own tooling. If your exit messages are encrypted, you'll need to decrypt them first. Edit this page Previous Introduction Next Exit Message Generation & Signing