Enable MEV

caution Charon is in a beta state and should be used with caution according to its Terms of Use . This quickstart guide focuses on configuring the builder API for Charon and supported validator and consensus clients.

Getting started with Charon & the Builder API

Running a distributed validator cluster with the builder API enabled will give the validators in the cluster access to the builder network. This builder network is a network of "Block Builders" who work with MEV searchers to produce the most valuable blocks a validator can propose.

MEV-Boost is one such product from flashbots that enables you to ask multiple block relays (who communicate with the "Block Builders") for blocks to propose. The block that pays the largest reward to the validator will be signed and returned to the relay for broadcasting to the wider network. The end result for the validator is generally an increased APR as they receive some share of the MEV.

info Before completing this guide, please check your cluster version, which can be found inside thecluster-lock.json file. If you are using cluster-lock version1.7.0 or higher release verions, Obol seamlessly accommodates all validator client implementations within a mev-enabled distributed validator cluster.

For clusters with a cluster-lock version1.6.0 and below, charon is compatible only with Teku . Use the version history feature of this documentation to see the instructions for configuring a cluster in that manner (v0.16.0).

Client configuration

note You need to add CLI flags to your consensus client, charon client, and validator client, to enable the builder API.

You need all operators in the cluster to have their nodes properly configured to use the builder API, or you risk missing a proposal.

Charon

Charon supports builder API with the--builder-api flag. To use builder API, one simply needs to add this flag to thecharon run command:

charon run --builder-api

Consensus Clients

The following flags need to be configured on your chosen consensus client. A flashbots relay URL is provided for example purposes, you should choose a relay that suits your preferences fromthis list .

- Teku
- Lighthouse
- Prysm
- Nimbus
- Lodestar

Teku can communicate with a single relay directly:--builder-endpoint="https://0xac6e77dfe25ecd6110b8e780608cce0dab71fdd5ebea22a16c0205200f2f8e2e3ad3b71d3499c54ad14d6c21b41a37ae@boost-relay.flashbots.net" Or you can configure it to communicate with a localMEV-boost sidecar to configure multiple relays:--builder-endpoint=http://mev-boost:18550 Lighthouse can communicate with a single relay directly:lighthouse bn --builder "https://0xac6e77dfe25ecd6110b8e780608cce0dab71fdd5ebea22a16c0205200f2f8e2e3ad3b71d3499c54ad14d6c21b41a37ae@boost-relay.flashbots.net" Or you can configure it to communicate with a localMEV-boost sidecar to configure multiple relays:lighthouse bn --builder "http://mev-boost:18550" prysm beacon-chain --http-mev-relay "https://0xac6e77dfe25ecd6110b8e780608cce0dab71fdd5ebea22a16c0205200f2f8e2e3ad3b71d3499c54ad14d6c21b41a37ae@boost-relay.flashbots.net" --payload-builder=true --payload-builder-url="https://0xac6e77dfe25ecd6110b8e780608cce0dab71fdd5ebea22a16c0205200f2f8e2e3ad3b71d3499c54ad14d6c21b41a37ae@boost-relay.flashbots.net" You should also consider adding--local-block-value-boost 3 as a flag, to favour locally built blocks if they are withing 3% in value of the relay block, to improve the chances of a successful proposal. --builder --builder.urls "https://0xac6e77dfe25ecd6110b8e780608cce0dab71fdd5ebea22a16c0205200f2f8e2e3ad3b71d3499c54ad14d6c21b41a37ae@boost-relay.flashbots.net"

Validator Clients

The following flags need to be configured on your chosen validator client

- Teku
- Lighthouse
- Prysm
- Nimbus
- Lodestar

teku validator-client --validators-builder-registration-default-enabled=true lighthouse vc --builder-proposals prysm validator --enable-builder --payload-builder=true --builder="true" --builder.selection="builderonly"

Verify your cluster is correctly configured

It can be difficult to confirm everything is configured correctly with your cluster until a proposal opportunity arrives, but here are some things you can check.

When your cluster is running, you should see if charon is logging something like this each epoch:

13:10:47.094 INFO bcast Successfully submitted validator registration to beacon node {"delay": "24913h10m12.094667699s", "pubkey": "84b_713", "duty": "1/builder_registration"} This indicates that your charon node is successfully registering with the relay for a blinded block when the time comes.

If you are using theultrasound relay , you can enter your cluster's distributed validator public key(s) into their website, to confirm they also see the validator as correctly registered.

You should check that your validator client's logs look healthy, and ensure that you haven't added afee-recipient address that conflicts with what has been selected by your cluster in your cluster-lock file, as that may prevent your validator from producing a signature for the block when the opportunity arises. You should also confirm the same for all of the other peers in your cluster.

Once a proposal has been made, you should look at theBlock Extra Data field underExecution Payload for the block onBeaconcha.in , and confirm there is text present, this generally suggests the block came from a builder, and was not a locally constructed block. Edit this page
Previous Create a DV using the SDKNext Split existing validator private keys