Deployment Best Practices

The following are a selection of best practices for deploying Distributed Validator Clusters at scale on mainnet.

Hardware Specifications

The following specifications are recommended for bare metal machines for clusters intending to run a significant number of mainnet validators:

Minimum Specs

- A CPU with 4+ cores, favouring high clock speed over more cores. (>3.0GHz and higher or a cpubenchmarksingle thread
- score of >2,500)
- 16GB of RAM
- 2TB+ free SSD disk space (for mainnet)
- 10mb/s internet bandwidth

Recommended Specs for extremely large clusters

- A CPU with 8+ physical cores, with clock speeds >3.5Ghz
- 32GB+ RAM (depending on the EL+CL clients)
- 4TB+ NVMe storage
- 25mb/s internet bandwidth

An NVMe storage device ishighly recommended for optimal performance , offering nearly 10x more random read/writes per second than a standard SSD.

Inadequate hardware (low-performance virtualized servers and/or slow HDD storage) has been observed to hinder performance, indicating the necessity of provisioning adequate resources.CPU clock speed and Disk throughput+latency are the most important factors for running a performant validator.

Note that the Charon client itself takes less than 1GB of RAM and minimal CPU load. In order to optimize both performance and cost-effectiveness, it is recommended to prioritize physical over virtualized setups. Such configurations typically offer greater performance and minimize overhead associated with virtualization, contributing to improved efficiency and reliability.

When constructing a DV cluster, it is important to be conscious of whether a cluster runs across cloud providers or stays within a single provider's private networking. This likely can impact the bandwidth and latency of the connections between nodes, as well as the egress costs of the cluster (Charon has a relatively low communication with its peers, averaging 10s of kb/s in large mainnet clusters). Ideally, bare metal machines in different locations within the same continent and with at least two providers, balances redundancy and performance.

Intra-cluster Latency

It is recommended tokeep peer ping latency below 235 milliseconds for all peers in a cluster . Charon should report a consensus duration averaging under 1 second through its prometheus metriccore_consensus_duration_seconds_bucket and associated grafana panel titled "Consensus Duration".

In cases where latencies exceed these thresholds, efforts should be made to reduce the physical distance between nodes or optimize Internet Service Provider (ISP) settings accordingly. Ensure all nodes are connecting to one another directly rather than through a relay.

For high-scale, performance deployments; inter-peer latency of < 25ms is optimal, along with an average consensus duration under 100ms.

Node Locations

For optimal performance and high availability, it is recommended to provision machines or virtual machines (VMs) within the same continent. This practice helps minimize potential latency issues ensuring efficient communication and responsiveness. Consider maps ofundersea internet cables when selecting locations across oceans with low latency.

Peer Connections

Charon clients can establish connections with one another in two ways: either through a third publicly accessible server known asa relay or directly with one another if they can establish a connection. The former is known as a relay connection and the latter is known as a direct connection.

It is important that all nodes in a cluster be directly connected to one another - this can halve the latency between them and reduces bandwidth constraints significantly. Opening Charon's p2p port (the default is3610) to the Internet, or configuring your routers NAT gateway to permit connections to your Charon client, are what are required to facilitate a direct connection between clients.

Instance Independence

Each node in the cluster should have its own independent beacon node (EL+CL) and validator client as well as Charon client. Sharing beacon nodes between the different nodes would potentially impact the fault tolerance of the cluster and as a result should be avoided.

Placement of Charon clients

If you wish to divide a Distributed Validator node across multiple physical or virtual machines; locate the Charon client on the EL/CL machine instead of the VC machine. This setup reduces latency from Charon to the consensus layer, as well as keeping the public-internet connected clients separate from the clients that hold the validator private keys. Be sure to use encrypted communication between your VC and the Charon client, potentially through a cloud-provided network, a self-managed network tunnel, a VPN, a KubernetesCNI , or other manner.

Node Configuration

Cluster sizes that allow for Byzantine Fault Tolerance are recommended as they are safer than clusters with simply Crash Fault Tolerance (See this guide for reference -Cluster Size and Resilience).

MEV-Boost Relays

MEV relays are configured at the Consensus Layer or MEV-boost client level. Refer to ourguide to ensure all necessary configuration has been applied to your clients. As with all validators, low latency during proposal opportunities is extremely important. By default, MEV-Boost waits for all configured relays to return a bid, or will timeout if any have not returned a bid within 950ms. This default timeout is generally too slow for a distributed cluster (think of this time as additive to the time it takes the cluster to come to consensus, both of which need to happen within a 2 second window for optimal proposal broadcasting). It is likely better to only list relays that are located geographically near your node, so that once all relays respond (e.g. in < 50ms) your cluster will move forward with the proposal.

Use Charon'stest mev command to test a number of your preferred relays, and select the two or three relays with the lowest latency to your node(s), you do not need to have the same relays on each node in a cluster.

Client Diversity

The clusters should consist of a combination of your preferred consensus, execution, and validator clients. It is recommended to include a combination of multiple clients in order to have a healthy client diversity within the cluster, ideally, if any single client type fails, it should be less than the fault tolerance of the cluster, and the validators should stay online/not do anything slashable.

Remote signers can be included as well, such as Web3signer or Dirk. A diversity of private key infrastructure setups further reduces the risk of total key compromise.

Tested client combinations can be found in therelease notes for each Charon version.

Metrics Monitoring

As requested by Obol Labs, node operators can pushstandard monitoring (Prometheus) and logging (Loki) data to Obol Labs' core team's cloud infrastructure for in-depth analysis of performance data and to assist during potential issues that may arise. Our recommendation for operators is to independently store information on their node health over the course of the validator lifecycle as well as any information on validator performance that they collect during the normal life cycle of a validator.

Obol Splits

LeveragingObol Splits smart contracts allows for non-custodial fund handling and allows for net customer payouts in an ongoing manner. Obol Splits ensure no commingling of funds across customers, and maintain full non-custodial integrity. Read more about Obol Splitshere .

Deposit Process

Deposit processes can be done via an automated script. This can be used for DV clusters until they reach the desired number of validators.

It is important to allow time for the validators to be activated (usually < 24 hours).

Consider using batching smart contracts to reduce the gas cost of a script, but take caution in their integration not to make an invalid deposit. Edit this page Previous Self-Host a Relay Next Advanced Docker Configs