Conductor

This page will teach you what theop-conductor service is and how it works on a high level. It will also get you started on setting it up in your own environment.

Enhancing sequencer reliability and availability

Theop-conductor(opens in a new tab) is an auxiliary service designed to enhance the reliability and availability of a sequencer within high-availability setups. By minimizing the risks associated with a single point of failure, the op-conductor ensures that the sequencer remains operational and responsive.

Assumptions

It is important to note that theop-conductor does not incorporate Byzantine fault tolerance (BFT). This means the system operates under the assumption that all participating nodes are honest and act correctly.

Summary of guarantees

The design of theop-conductor provides the following guarantees:

- No Unsafe Reorgs
- No Unsafe Head Stall During Network Partition
- 100% Uptime with No More Than 1 Node Failure

Design

On a high level,op-conductor serves the following functions:

Raft consensus layer participation

- Leader determination:
- Participates in the Raft consensus algorithm to
- determine the leader among sequencers.
- State management:
- Stores the latest unsafe block ensuring consistency
- across the system.

RPC request handling

- Admin RPC:
- Provides administrative RPCs for manual recovery scenarios,
- including, but not limited to: stopping the leadership vote and removing itself
- from the cluster.
- Health RPC:
- Offers health RPCs for theop-node
- to determine whether it
- should allow the publishing of transactions and unsafe blocks.

Sequencer health monitoring

- Continuously monitors the health of the sequencer (op-node) to ensure
- optimal performance and reliability.

Control loop management

- Implements a control loop to manage the status of the sequencer (op-node),
- including starting and stopping operations based on different scenarios and
- health checks.

Conductor state transition

The following is a state machine diagram of how the op-conductor manages the sequencers Raft consensus.

Helpful tips: To better understand the graph, focus on one node at a time, understand what can be transitioned to this current state and how it can transition to other states. This way you could understand how we handle the state transitions.

Setup

At OP Labs, op-conductor is deployed as a kubernetes statefulset because it requires a persistent volume to store the raft log. This guide describes setting up conductor on an existing network without incurring downtime.

You can utilize theop-conductor-ops(opens in a new tab) tool to confirm the conductor status between the steps.

Assumptions

This setup guide has the following assumptions:

- 3 deployed sequencers (sequencer-0, sequencer-1, sequencer-2) that are all
- in sync and in the same vpc network
- sequencer-0 is currently the active sequencer
- You can execute a blue/green style sequencer deployment workflow that
- involves no downtime (described below)
- conductor and sequencers are running in k8s or some other container
- orchestrator (vm-based deployment may be slightly different and not covered
- here)

Spin up op-conductor

Deploy conductor

Deploy a conductor instance per sequencer with sequencer-1 as the raft cluster bootstrap node:

- suggested conductor configs:
- OP_CONDUCTOR_CONSENSUS_ADDR
- :
- ''
- OP_CONDUCTOR_CONSENSUS_PORT
- :
- '50050'
- OP_CONDUCTOR_EXECUTION_RPC
- :
- ':8545'
- OP_CONDUCTOR_HEALTHCHECK_INTERVAL
- :
- '1'
- OP_CONDUCTOR_HEALTHCHECK_MIN_PEER_COUNT
- :
- '2'
- # set based on your internal p2p network peer count
- OP_CONDUCTOR_HEALTHCHECK_UNSAFE_INTERVAL
- :
- '5'
- # recommend a 2-3x multiple of your network block time to account for temporary performance issues
- OP_CONDUCTOR_LOG_FORMAT
- :
- logfmt
- OP_CONDUCTOR_LOG_LEVEL
- :
- info
- OP_CONDUCTOR_METRICS_ADDR
- :
- 0.0.0.0
- OP_CONDUCTOR_METRICS_ENABLED
- :
- 'true'
- OP_CONDUCTOR_METRICS_PORT

- :
- '7300'
- OP_CONDUCTOR_NETWORK
- :
- ''
- OP_CONDUCTOR_NODE_RPC
- :
- ':8545'
- OP_CONDUCTOR_RAFT_SERVER_ID
- :
- 'unique raft server id'
- OP_CONDUCTOR_RAFT_STORAGE_DIR
- :
- /conductor/raft
- OP_CONDUCTOR_RPC_ADDR
- :
- 0.0.0.0
- OP_CONDUCTOR_RPC_ENABLE_ADMIN
- :
- 'true'
- OP_CONDUCTOR_RPC_ENABLE_PROXY
- :
- 'true'
- OP_CONDUCTOR_RPC_PORT
- :
- '8547'
- sequencer-1 op-conductor extra config:
- OP_CONDUCTOR_PAUSED
- :
- "true"
- OP_CONDUCTOR_RAFT_BOOTSTRAP
- :
- "true"

Pause two conductors

Pausesequencer-0 &sequencer-2 conductors withconductor_pause RPC request.

Update op-node configuration and switch the active sequencer

Deploy anop-node config update to all sequencers that enables conductor. Use a blue/green style deployment workflow that switches the active sequencer tosequencer-1 :

- all sequencer op-node configs:
- OP_NODE_CONDUCTOR_ENABLED
- :
- "true"

- # this is what commits unsafe blocks to the raft logs

- OP_NODE_RPC_ADMIN_STATE
- :
- ""

- # this flag can't be used with conductor

Confirm sequencer switch was successful

Confirmsequencer-1 is active and successfully producing unsafe blocks. Becausesequencer-1 was the raft cluster bootstrap node, it is now committing unsafe payloads to the raft log.

Add voting nodes

Add voting nodes to cluster usingconductor_AddServerAsVoter RPC request to the leader conductor (sequencer-1)

Confirm state

Confirm cluster membership and sequencer state:

- sequencer-0
- andsequencer-2
- :
- 1. raft cluster follower
- 1. sequencer is stopped
- 1. conductor is paused
- 1. conductor enabled in op-node config
- sequencer-1
- 1. raft cluster leader
- 1. sequencer is active
- 1. conductor is paused
- 1. conductor enabled in op-node config

Resume conductors

Resume all conductors withconductor_resume RPC request to each conductor instance.

Confirm state

Confirm all conductors successfully resumed withconductor_paused

Transfer leadership

Trigger leadership transfer tosequencer-0 usingconductor_transferLeaderToServer

Confirm state

- sequencer-1
- andsequencer-2
- :
- 1. raft cluster follower
- 1. sequencer is stopped
- 1. conductor is active
- 1. conductor enabled in op-node config
- sequencer-0
- 1. raft cluster leader
- 1. sequencer is active
- 1. conductor is active
- 1. conductor enabled in op-node config

Update configuration

Deploy a config change tosequencer-1 conductor to remove theOP_CONDUCTOR_PAUSED: true flag andOP_CONDUCTOR_RAFT_BOOTSTRAP flag.

Blue/green deployment

In order to ensure there is no downtime when setting up conductor, you need to have a deployment script that can update sequencers without network downtime.

An example of this workflow might look like:

1. Query current state of the network and determine which sequencer is
2. currently active (referred to as "original" sequencer below).
3. From the other available sequencers, choose a candidate sequencer.
4. Deploy the change to the candidate sequencer and then wait for it to sync
5. up to the original sequencer's unsafe head. You may want to check peer counts
6. and other important health metrics.
7. Stop the original sequencer usingadmin_stopSequencer
8. which returns the
9. last inserted unsafe block hash. Wait for candidate sequencer to sync with
10. this returned hash in case there is a delta.
11. Start the candidate sequencer at the original's last inserted unsafe block
12. hash.1. Here you can also execute additional check for unsafe head progression
13.
    1. and decide to roll back the change (stop the candidate sequencer, start the
14.
    1. original, rollback deployment of candidate, etc.)
15. Deploy the change to the original sequencer, wait for it to sync to the
16. chain head. Execute health checks.

Post-conductor launch deployments

After conductor is live, a similar canary style workflow is used to ensure minimal downtime in case there is an issue with deployment:

1. Choose a candidate sequencer from the raft-cluster followers
2. Deploy to the candidate sequencer. Run health checks on the candidate.
3. Transfer leadership to the candidate sequencer usingconductor_transferLeaderToServer
4. . Run health checks on the candidate.
5. Test if candidate is still the leader usingconductor_leader
6. after some
7. grace period (ex: 30 seconds)1. If not, then there is likely an issue with the deployment. Roll back.
8. Upgrade the remaining sequencers, run healthchecks.

Configuration options

It is configured via itsflags / environment variables(opens in a new tab)

--consensus.addr (CONSENSUS_ADDR

)

- Usage:
- Address to listen for consensus connections
- Default Value:
- 127.0.0.1
- Required:
- yes

--consensus.port (CONSENSUS_PORT

)

- Usage:
- Port to listen for consensus connections
- Default Value:
- 50050
- Required:
- yes

--raft.bootstrap (RAFT_BOOTSTRAP

)

For bootstrapping a new cluster. This should only be used on the sequencer that is currently active and can only be started once with this flag, otherwise the flag has to be removed or the raft log must be deleted before re-bootstrapping the cluster. * Usage: * If this node should bootstrap a new raft cluster * Default Value: * false * Required: * no

--raft.server.id (RAFT_SERVER_ID

)

- Usage:
- Unique ID for this server used by raft consensus
- Default Value:
- None specified
- Required:
- yes

--raft.storage.dir (RAFT_STORAGE_DIR

)

- Usage:
- Directory to store raft data
- Default Value:
- None specified
- Required:
- yes

--node.rpc (NODE_RPC

)

- Usage:
- HTTP provider URL for op-node
- Default Value:
- None specified
- Required:
- yes

--execution.rpc (EXECUTION_RPC

)

- Usage:
- HTTP provider URL for execution layer
- Default Value:
- None specified
- Required:
- yes

--healthcheck.interval (HEALTHCHECK_INTERVAL

)

- Usage:
- Interval between health checks
- Default Value:
- None specified
- Required:
- yes

--healthcheck.unsafe-interval (HEALTHCHECK_UNSAFE_INTERVAL

)

- Usage:
- Interval allowed between unsafe head and now measured in seconds
- Default Value:
- None specified
- Required:
- yes

--healthcheck.safe-enabled (HEALTHCHECK_SAFE_ENABLED

)

- Usage:
- Whether to enable safe head progression checks
- Default Value:
- false
- Required:
- no

--healthcheck.safe-interval (HEALTHCHECK_SAFE_INTERVAL

)

- Usage:
- Interval between safe head progression measured in seconds
- Default Value:
- 1200
- Required:
- no

--healthcheck.min-peer-count (HEALTHCHECK_MIN_PEER_COUNT

)

- Usage:
- Minimum number of peers required to be considered healthy
- Default Value:
- None specified
- Required:
- yes

--paused (PAUSED

)

There is no configuration state, so if you unpause via RPC and then restart, it will start paused again. * Usage: * Whether the conductor is paused * Default Value: * false * Required: * no

--rpc.enable-proxy (RPC_ENABLE_PROXY

)

- Usage:
- Enable the RPC proxy to underlying sequencer services
- Default Value:
- true
- Required:
- no

RPCs

Conductor exposesadmin RPCs(opens in a new tab) on theconductor namespace.

conductor_overrideLeader

OverrideLeader is used to override the leader status, this is only used to return true forLeader() &LeaderWithID() calls. It does not impact the actual raft consensus leadership status. It is supposed to be used when the cluster is unhealthy and the node is the only one up, to allow batcher to be able to connect to the node so that it could download blocks from the manually started sequencer.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_overrideLeader","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_pause

Pause pauses op-conductor.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_pause","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_resume

Resume resumes op-conductor.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_resume","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_paused

Paused returns true if the op-conductor is paused.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_paused","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_stopped

Stopped returns true if the op-conductor is stopped.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_stopped","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_sequencerHealthy

SequencerHealthy returns true if the sequencer is healthy.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_sequencerHealthy","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_leader

API related to consensus. Leader returns true if the server is the leader.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_leader","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_leaderWithID

API related to consensus. LeaderWithID returns the current leader's server info.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_leaderWithID","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_addServerAsVoter

API related to consensus. The address parameter is the raft consensus address, not the RPC address. AddServerAsVoter adds a server as a voter to the cluster.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_addServerAsVoter","params":[, ,],"id":1}' \ http://127.0.0.1:50050

conductor_addServerAsNonvoter

API related to consensus. AddServerAsNonvoter adds a server as a non-voter to the cluster. non-voter The non-voter will not participate in the leader election.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_addServerAsNonvoter","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_removeServer

API related to consensus. RemoveServer removes a server from the cluster.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_removeServer","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_transferLeader

API related to consensus. TransferLeader transfers leadership to another server.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_transferLeader","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_transferLeaderToServer

API related to consensus. TransferLeaderToServer transfers leadership to a specific server.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_transferLeaderToServer","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_clusterMembership

ClusterMembership returns the current cluster membership configuration.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_clusterMembership","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_active

API called byop-node . Active returns true if the op-conductor is active (not paused or stopped).

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_active","params":[],"id":1}' \ http://127.0.0.1:50050

conductor_commitUnsafePayload

API called byop-node . CommitUnsafePayload commits an unsafe payload (latest head) to the consensus layer.

curl cast curl

-X

POST

-H

"Content-Type: application/json"

--data \ '{"jsonrpc":"2.0","method":"conductor_commitUnsafePayload","params":[],"id":1}' \ http://127.0.0.1:50050

Next steps

- Checkoutop-conductor-mon(opens in a new tab)
- :
- which monitors multiple op-conductor instances and provides a unified interface
- for reporting metrics.
- Get familiar withop-conductor-ops(opens in a new tab)
- to interact with op-conductor.

op-conductor op-txproxy