OP Stack Genesis Creation

The following guide shows you how to generate the L2 allocs. This is a JSON file that represents the L2 genesis state. You will provide this file to the execution client (op-geth) to initialize your network.

Solidity Script

At the time of this writing, the preferred method for genesis generation is to use the foundry script located in the monorepo. It can be found atpackages/contracts-bedrock/scripts/L2Genesis.s.sol(opens in a new tab) .

Configuration

Create or modify a file.json inside thedeploy-config folder in the monorepo. The script will read the latest active fork from the deploy config and the L2 genesis allocs generated will be compatible with this fork. The automatically detected fork can be overwritten by setting the environment variableFORK either to the lower-case fork name (currentlydelta ,ecotone , orfjord) or tolatest , which will select the latest fork available (currentlyfjord).

By default, the script will dump the L2 genesis allocs of the detected or selected fork only, to the file atSTATE_DUMP_PATH . The optional environment variableOUTPUT_MODE allows you to modify this behavior by setting it to one of the following values:

- latest
- (default) - only dump the selected fork's allocs.
- all
-
    - also dump all intermediary fork's allocs. This only works ifSTATE_DUMP_PATH
- is not set. In this case, all allocs will be written to files/state-dump-.json
- . Another path cannot currently be specified for this
- use case.
- none
-
    - won't dump any allocs. Only makes sense for internal test usage.

Creation

- CONTRACT_ADDRESSES_PATH
- represents the deployment artifact that was
- generated during a contract deployment.
- DEPLOY_CONFIG_PATH
- represents a path on the filesystem that points to a
- deployment config. The same deploy config JSON file should be used for L1 contracts
- deployment as when generating the L2 genesis allocs.
- STATE_DUMP_PATH
- represents the filepath at which the allocs will be
- written to on disk.

CONTRACT_ADDRESSES_PATH

deployments/artifact.json \ DEPLOY_CONFIG_PATH =< PATH_TO_MY_DEPLOY_CONFI G

    \ STATE_DUMP_PATH =< PATH_TO_WRITE_L2_ALLOC S

    \ forge

script

scripts/L2Genesis.s.sol:L2Genesis \ --sig

'runWithStateDump()'

Subcommand (op-node genesis l2)

Historically, the genesis file creation was handled by thegenesis l2 subcommand, provided by theop-node . The following is an example of its usage fromv1.7.6(opens in a new tab) .

go

run

cmd/main.go

genesis

l2 \ --deploy-config= < Path

to

deploy

config

fil e

        \ --l1-deployments

value= < Path

to

L1

deployments

JSON

file

as

in

superchain-registr y

        \ --outfile.l2= < Path

to

L2

genesis

output

file:

i.e.

./genesis.jso n

        \ --outfile.rollup= < Path

to

rollup

output

file:

i.e.

./rollup.jso n

        \ --l1-rpc= < RPC

URL

for

an

Ethereum

L1

node.

Cannot

be

used

with

--l1-starting-bloc

Alternatively, you can omit--l1-rpc and use--l1-starting-block . This option lets you provide a path to a JSON file containing the L1 starting block. This overrides the need for using an L1 RPC to fetch the block. It cannot be used with--l1-rpc .

Next Steps

- Learn how toinitialize
- op-geth
- with your genesis file
- Learn more about the off chainarchitecture
- .

Contract Deployment Creating Your Own L2 Rollup Testnet