OP Stack Deployment Overview

When deploying an OP Stack chain, you'll be setting up four different components. It's useful to understand what each of these components does before you start deploying your chain. The OP Stack can be deployed as a L3, which includes additional considerations. The following information assumes you're deploying a L2 Rollup on Ethereum.

Smart Contracts

The OP Stack uses a set of smart contracts on the L1 blockchain to manage aspects of the Rollup. Each OP Stack chain has its own set of L1 smart contracts that are deployed when the chain is created.

⚠ Standard OP Stack chains should only use governance approved and audited smart contracts. The monorepo has them tagged with the following patternop-contracts/vX.X.X and you can review the release notes for details on the changes. Read more about the details on ourSmart Contract Release Section .

Sequencer Node

OP Stack chains use a Sequencer node to gather proposed transactions from users and publish them to the L1 blockchain. OP Stack chains rely on at least one of these Sequencer nodes. You can also run additional replica (non-Sequencer) nodes.

Consensus Client

OP Stack Rollup nodes, like Ethereum nodes, have a consensus client. The consensus client is responsible for determining the list and ordering of blocks and transactions that are part of your blockchain. Several implementations of the OP Stack consensus client exist, includingop-node (maintained by the Optimism Foundation),magi (opens in a new tab) (maintained by a16z) andhildr (opens in a new tab) (maintained by OptimismJ).

Execution Client

OP Stack nodes, like Ethereum nodes, also have an execution client. The execution client is responsible for executing transactions and maintaining the state of the blockchain. Various implementations of the OP Stack execution client exist, includingop-geth (maintained by Optimism Foundation),op-erigon (opens in a new tab) (maintained by Test in Prod), andop-nethermind (opens in a new tab) .

Batcher

The Batcher is a service that publishes transactions from the Rollup to the L1 blockchain. The Batcher runs continuously alongside the Sequencer and publishes The Batcher continuously publishes transactions alongside the Sequencer in regular batches.

Proposer

The Proposer is a service responsible for publishing transactionsresults (in the form of L2 state roots) to the L1 blockchain. This allows smart contracts on L1 to read the state of the L2, which is necessary for cross-chain communication and reconciliation between state changes.

Software Dependencies

Dependency Version Version Check Command git(opens in a new tab) ^2 git --version go(opens in a new tab) ^1.21 go version node(opens in a new tab) ^20 node --version pnpm(opens in a new tab) ^8 pnpm --version foundry(opens in a new tab) ^0.2.0 forge --version make(opens in a new tab) ^3 make --version jq(opens in a new tab) ^1.6 jq --version direnv(opens in a new tab) ^2 direnv --version

Notes on Specific Dependencies

node

We recommend using the latest LTS version of Node.js (currently v20)nvm (opens in a new tab) is a useful tool that can help you manage multiple versions of Node.js on your machine. You may experience unexpected errors on older versions of Node.js.

foundry

It's recommended to use the scripts in the monorepo'spackage.json for managingfoundry to ensure you're always working

with the correct version. This approach simplifies the installation, update, and version checking process.

direnv

Parts of our tutorial use direnv (opens in a new tab) as a way of loading environment variables from.envrc files into your shell. This means you won't have to manually export environment variables every time you want to use them.direnv only ever has access to files that you explicitly allow it to see.

After installingdirenv (opens in a new tab) , you will need tomake sure that direnv is hooked into your shell(opens in a new tab) . Make sure you've followed the guide on thedirenv website(opens in a new tab), thenclose your terminal and reopen it so that the changes take effect (orsource your config file if you know how to do that).

Make sure that you have correctly hookeddirenv into your shell by modifying your shell configuration file (like~/.bashrc or~/.zshrc). If you haven't edited a config file then you probably haven't configureddirenv properly (and things might not work later).

Next Steps

- Discover how todeploy the smart contracts
- .
- Find out how to create yourgenesis file
- .
- Explore some chain operatorbest practices
- .

Run a Custom Gas Token ChainContract Deployment