Testing Apps for OP Mainnet

For the most part, running applications on OP Mainnet is identical to running them on Ethereum, so the testing is identical too. In this guide, you learn the best practices for OP Mainnet testing where there are differences.

Unit tests and single layer integration tests

The vast majority of tests do not involve any OP Mainnet-specific features. In those cases, while you could test everything on OP Mainnet or a test network, that would normally be inefficient. Most Ethereum development stacks include features that make testing easier, which normal Ethereum clients, such as geth (and our modified version, op-geth) don't support. Therefore, it is a good idea to run the majority of tests, which do not rely on OP Mainnet-specific features, in the development stack. It is a lot faster.

It is a best practice to design and run thorough tests across an OP test network, either in your local development environment or on the test network , depending on your use case. Running proper testing is key to identifying fringe cases where the equivalence between OP Mainnet and Ethereum breaks down (or where Ethereum mainnet itself and the development stack may be non-equivalent in a production environment).

Multilayer integration tests

Some apps need OP Mainnet-specific features that aren't available as part of the development stack. For example, if your decentralized application relies on inter-domain communication , the effort of developing a stub to let you debug it in a development stack is probably greater than the hassle of having the automated test go to a local development environment each time.

Integration with other products

In many cases a decentralized application requires the services of other contracts. For example, Perpetual v. 2(opens in a new tab) cannot function without Uniswap v. 3(opens in a new tab) .

If that is the case you can use mainnet forking(opens in a new tab) . It works with OP Mainnet. Alternatively, you can connect to our test network if those contracts are also deployed there (in many cases they are).

Running a Local Development Environment Security Model & FAQ