Sandbox Reference

Here you will find a reference to everything available within the Sandbox.

Installation

You can run the Sandbox using Docker. See theQuickstart for instructions on installing Docker.

With Docker

bash -i < (curl -s install.aztec.network) This will install the following:

- aztec
-
    - launches various infrastructure subsystems (sequencer, prover, pxe, etc).
- aztec-cli
-
    - a command line tool for interfacing and experimenting with infrastructure.
- aztec-nargo
-
    - aztec's build of nargo, the noir compiler toolchain.
- aztec-sandbox
-
    - a wrapper around docker-compose that launches services needed for sandbox testing.
- aztec-up
-
    - a tool to upgrade the aztec toolchain to the latest, or specific versions.

Once these have been installed, to start the sandbox, run:

aztec-sandbox This will attempt to run the Sandbox with the PXE listening onlocalhost:8080 . You can change the port defined in./.aztec/docker-compose.yml or by setting thePXE_PORT environment variable. Running the install command again will overwrite any changes made to thedocker-compose.yml .

See the full list of configurable environment variableshere .

If you have previously installed the CLI via a node package manager, you will need to uninstall it and remove it from your project dependencies and install it via Docker.

To install a specific version of the sandbox, you can set the environment variableSANDBOX_VERSION

VERSION

< version

bash -i < (curl -s install.aztec.network)

Running

Once the installed, you can run the sandbox with:

aztec-sandbox Alternatively, you can run like so:

cd ~/.aztec &&

docker-compose up

Running Aztec PXE / Node / P2P-Bootstrap node

If you wish to run components of the Aztec network stack separately, you can use theaztec start command with various options for enabling components.

aztec start --node [nodeOptions] --pxe [pxeOptions] --archiver [archiverOptions] --sequencer [sequencerOptions] ---- p2p-bootstrap [p2pOptions] Starting the aztec node alongside a PXE, sequencer or archiver, will attach the components to the node. If you want to e.g. run a PXE separately to a node, you can: Start a node:

aztec start --node [node] --archiver [archiverOptions] Then start a PXE on a separate terminal that connects to that node:

aztec start --pxe nodeUrl = http://localhost:8080

Environment Variables

There are various environment variables you can use when running the whole sandbox or when running on of the available modes.

Sandbox

DEBUG=aztec: *# The level of debugging logs to be displayed. using "aztec":* will log everything.
ETHEREUM_HOST=http://ethereum:8545 # The Ethereum JSON RPC URL. We use an anvil instance that runs in parallel to the sandbox on docker by default. HOST_WORKDIR='{PWD}' # The location to store log outpus. Will use ~/.aztec where the docker-compose.yml file is stored by default. CHAIN_ID=31337 # The Chain ID that the Ethereum host is using.
TEST_ACCOUNTS='true' # Option to deploy 3 test account when sandbox starts. (default: true)
DEPLOY_AZTEC_CONTRACTS='true' # Option to deploy the Aztec contracts when sandbox starts. (default: true)
MODE='sandbox' # Option to start the sandbox or a standalone part of the system. (default: sandbox)
AZTEC_NODE_PORT=8079 # The port that the Aztec node wil be listening to (default: 8079) PXE_PORT=8080 # The port that the PXE will be listening to (default: 8080)

Ethereum Forking (Optional: not enabled by default)

FORK_BLOCK_NUMBER=0 # The block number to fork from FORK_URL="" # The URL of the Ethereum node to fork from

Polling intervals

ARCHIVER_POLLING_INTERVAL_MS=50 P2P_BLOCK_CHECK_INTERVAL_MS=50
SEQ_TX_POLLING_INTERVAL_MS=50 WS_BLOCK_CHECK_INTERVAL_MS=50
PXE_BLOCK_POLLING_INTERVAL_MS=50 ARCHIVER_VIEM_POLLING_INTERVAL_MS=500 Aztec Node

Variables likeDEPLOY_AZTEC_CONTRACTS &AZTEC_NODE_PORT are valid here as described above.TEST_ACCOUNTS cannot be used here because the Aztec node does not control an Aztec account to deploy contracts from.

P2P config

Configuration variables for connecting a Node to the Aztec Node P2P network. You'll need a running P2P-Bootstrap node to connect to.

P2P_ENABLED='false' # A flag to enable P2P networking for this node. (default: false)
P2P_BLOCK_CHECK_INTERVAL_MS=100 # The frequency in which to check for new L2 blocks.
P2P_L2_BLOCK_QUEUE_SIZE=1000 # Size of queue of L2 blocks to store. P2P_TCP_LISTEN_PORT=40400 # The tcp port on which the P2P service should listen for connections. P2P_TCP_LISTEN_IP= #The tcp IP on which the P2P service should listen for connections. PEER_ID_PRIVATE_KEY='' # An optional peer id private key. If blank, will generate a random key. BOOTSTRAP_NODES='' # A list of bootstrap peers to connect to, separated by commas
P2P_ANNOUNCE_HOSTNAME='' # Hostname to announce to the p2p network P2P_ANNOUNCE_PORT='' # Port to announce to the p2p network P2P_KAD_CLIENT='false' # Optional specification to run as a client in the Kademlia routing protocol. P2P_NAT_ENABLED='false' # Whether to enable NAT from libp2p P2P_MIN_PEERS=10 # The minimum number of peers (a peer count below this will cause the node to look for more peers) P2P_MAX_PEERS=100 # The maximum number of peers (a peer count above this will cause the node to refuse connection attempts)

Aztec Contract Addresses

When running a standalone node, you need to have deployed Aztec contracts on your Ethereum host, then declare their addresses as env variables.

REGISTRY_CONTRACT_ADDRESS=0x01234567890abcde01234567890abcde
INBOX_CONTRACT_ADDRESS=0x01234567890abcde01234567890abcde
OUTBOX_CONTRACT_ADDRESS=0x01234567890abcde01234567890abcde
ROLLUP_CONTRACT_ADDRESS=0x01234567890abcde01234567890abcde

Sequencer variables

SEQ_PUBLISHER_PRIVATE_KEY=0x01234567890abcde01234567890abcde # Private key of an ethereum account that will be used by the sequencer to publish blocks. SEQ_MAX_TX_PER_BLOCK=32 # Maximum txs to go on a block. (default: 32) SEQ_MIN_TX_PER_BLOCK=1 # Minimum txs to go on a block. (default: 1) PXE

Variables likeTEST_ACCOUNTS &PXE_PORT are valid here as described above.DEPLOY_AZTEC_CONTRACTS cannot be used here as the PXE does not control an Ethereum account.

AZTEC_NODE_URL='http://localhost:8079' # The address of an Aztec Node URL that the PXE will connect to (default: http://localhost:8079) PXE_PORT=8080 # The port that the PXE will be listening to (default: 8080) TEST_ACCOUNTS='true' # Option to deploy 3 test account when sandbox starts. (default: true) PXE_BLOCK_POLLING_INTERVAL_MS=50 # Interval to check for new L2 blocks. (default: 50) PXE_L2_STARTING_BLOCK=1 # L2 Block to start synching the PXE from (default: 1) P2P Bootstrap Node

The P2P Bootstrap node is a standalone app whose purpose is to assist new P2P network participants in acquiring peers.

P2P_TCP_LISTEN_IP='0.0.0.0' # The IP Address on which to listen for connections. P2P_TCP_LISTEN_PORT=40400 # The port on which to listen for connections. PEER_ID_PRIVATE_KEY='' # The private key to be used by the peer for secure communications with other peers. This key will also be used to derive the Peer ID. P2P_ANNOUNCE_HOSTNAME='' # The IPAddress/Hostname that other peers should use to connect to this node, this may be different to P2P_TCP_LISTEN_IP if e.g. the node is behind a NAT. P2P_ANNOUNCE_PORT='' # The port that other peers should use to connect to this node, this may be different to P2P_TCP_LISTEN_PORT if e.g. the node is behind a NAT.

Cheat Codes

To help with testing, the sandbox is shipped with a set of cheatcodes.

Cheatcodes allow you to change the time of the Aztec block, load certain state or more easily manipulate Ethereum instead of having to write dedicated RPC calls to anvil or hardhat.

You can find the cheat code referencehere .

Contracts

We have shipped a number of example contracts in the@aztec/noir-contracts.jsnpm package . This is included with the cli by default so you are able to use these contracts to test with. To get a list of the names of the contracts run:

example-contracts % aztec-cli example-contracts BenchmarkingContractArtifact CardGameContractArtifact ChildContractArtifact CounterContractArtifact DocsExampleContractArtifact EasyPrivateTokenContractArtifact EasyPrivateVotingContractArtifact EcdsaAccountContractArtifact EscrowContractArtifact ImportTestContractArtifact InclusionProofsContractArtifact LendingContractArtifact ParentContractArtifact PendingNoteHashesContractArtifact PriceFeedContractArtifact SchnorrAccountContractArtifact SchnorrHardcodedAccountContractArtifact SchnorrSingleKeyAccountContractArtifact SlowTreeContractArtifact StatefulTestContractArtifact TestContractArtifact TokenBlacklistContractArtifact TokenBridgeContractArtifact TokenContractArtifact UniswapContractArtifact Source code: /yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L95-L118 You can see all of our example contracts in the monorepohere . Edit this page

Previous CLI Commands Next Cheat Codes