CLI Commands

Here you will find a reference to the commands available in the Aztec CLI.

Installation

Docker

The CLI will be installed automatically via Docker by running the command to install and start the sandboxinstructions here .

Update

The CLI comes with an update command.

aztec-cli update . --contract src/contract1 --contract src/contract2 This command does a few things to manage updates:

- It looks for apackage.json
- and updates all@aztec/
- dependencies to the versions the sandbox expects.
- It looks forNargo.toml
- at the--contract
- paths specified and updates allaztec.nr
- dependencies to the versions the sandbox expects.
- It outputs the changes.

You can specify a version to update to with the--aztec-version flag, but it defaults tolatest so this is typically not necessary.

info The update command won't update the CLI itself. To update these follow theupdating instructions which point to our curl command.

Compile

You can find more information about compiling contractson this page .

Creating Accounts

The first thing we want to do is create a couple of accounts. We will use thecreate-account command which will generate a new private key for us, register the account on the sandbox, and deploy a simple account contract whichuses a single key for privacy and authentication :

create-account % aztec-cli create-account Created new account:

Address: 0x20d3321707d53cebb168568e25c5c62a853ae1f0766d965e00d6f6c4eb05d599 Public key: 0x02d18745eadddd496be95274367ee2cbf0bf667b81373fb6bed715c18814a09022907c273ec1c469fcc678738bd8efc3e9053fe1acbb11fa32da0d6881a1370e Private key: 0x2aba9e7de7075deee3e3f4ad1e47749f985f0f72543ed91063cc97a40d851f1e Partial address: 0x72bf7c9537875b0af267b4a8c497927e251f5988af6e30527feb16299042ed Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L204-L212 Once the account is set up, the CLI returns the resulting address, its privacy key, and partial address. You can read more about thesehere .

Save the Address and Private key as environment variables. We will be using them later.

export

ADDRESS

< Address printed when you run the command

export

PRIVATE_KEY

< Private key printed when you run the command

Alternatively, we can also manually generate a private key and use it for creating the account, either via a-k option or by setting thePRIVATE_KEY environment variable.

create-account-from-private-key % aztec-cli generate-private-key

Private Key: 0x12684562c8676e66be100878434b01286a757dea468233f818b906f66fb34984 Public Key: 0x1003732857c052c1d6af4dd74b5631863a056c90a586c4e3ea6d94782ee712d317cdb713ed1ba02d3df0ac2b581d269490f9e24916c1b677c7259444aa0ad66b

% aztec-cli create-account --private-key 0x12684562c8676e66be100878434b01286a757dea468233f818b906f66fb34984

Created new account:

Address: 0x26e831b1b146d1faf0c1d27fc72f2243887e9963cc87a6b3af64fe6481920a80 Public key: 0x1003732857c052c1d6af4dd74b5631863a056c90a586c4e3ea6d94782ee712d317cdb713ed1ba02d3df0ac2b581d269490f9e24916c1b677c7259444aa0ad66b Partial address: 0x01e5e7b2abbfb98a93b7549ae80faa6886f8ea8e8f412416fb330b565fd2b4ed Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L162-L176 For all commands that require a user's private key, the CLI will look for thePRIVATE_KEY environment variable in absence of an optional argument.

Let's double check that the accounts have been registered with the sandbox using theget-accounts command:

get-accounts % aztec-cli get-accounts Accounts found:

Address: 0x0c8a6673d7676cc80aaebe7fa7504cf51daa90ba906861bfad70a58a98bf5a7d Public Key: 0x27c20118733174347b8082f578a7d8fb84b3ad38be293715eee8119ee5cd8a6d0d6b7d8124b37359663e75bcd2756f544a93b821a06f8e33fba68cc8029794d9

Partial Address: 0x1c6484e22441e5ca43bba53495d0cdc911da299150fde1191bcb330b64716ff9

Address: 0x226f8087792beff8d5009eb94e65d2a4a505b70baf4a9f28d33c8d620b0ba972 Public Key: 0x08145e8e8d46f51cda8d4c9cad81920236366abeafb8d387002bad879a3e87a81570b04ac829e4c007141d856d5a36d3b9c464e0f3c1c99cdbadaa6bb93f3257 Partial Address: 0x1833e53112953e6830a230cfc2895caed604f6395bbfafa730da26c5bf53c0a9

Address: 0x0e1f60e8566e2c6d32378bdcadb7c63696e853281be798c107266b8c3a88ea9b Public Key: 0x13e6151ea8e7386a5e7c4c5221047bf73d0b1b7a2ad14d22b7f73e57c1fa00c614bc6da69da1b581b09ee6cdc195e5d58ae4dce01b63bbb744e58f03855a94dd Partial Address: 0x30034aaf5d78821effa4827a132357110a49a4f37b6e384d884e233595bcf342

Address: 0x01b18c2044bbedd4a2e5f67cf6858370ccfb2b869b2000abe2f4ca12d9cc166e Public Key: 0x240845f1179e3fbaa6ce587d44722b3452bbdaa11deb29553196b23534985d432b746bcf2f0e7046eb13f0ca0c4fedd027dc80b64384f50d6a14ad248faa941a Partial Address: 0x03834845fc488d1454f195abe7d52b3393f6902eee080c90cd694c63572f7160 Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L233-L252 You will see that a number of accounts exist that we did not create. The Sandbox initializes itself with 3 default accounts. Save one of the printed accounts (not the one that you generated above) in an environment variable. We will use it later.

export

ADDRESS2

< Account address printed by the above command

Deploying a Token Contract

We will now deploy a token contract using thedeploy command, and set an address of the admin via a constructor argument. You can find the contract we are deployinghere (or write it for yourself inthis tutorial!) Make sure to replace this address with one of the two you created earlier.

deploy % aztec-cli deploy TokenContractArtifact --private-key PRIVATE_KEY --args ADDRESS TokenName TKN 18

Contract deployed at 0x1ae8eea0dc265fb7f160dae62cc8912686d8a9ed78e821fbdd8bcedc54c06d0f Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L266-L270 Save the contract address as an environment variable. We will use it later.

export

CONTRACT_ADDRESS

< Your new contract address

- --args
-
    - Arguments to the constructor of the contract. In this case we have set an address as admin.

The CLI tells us that the contract was successfully deployed. We can use thecheck-deploy command to verify that a contract has been successfully deployed to that address:

check-deploy % aztec-cli check-deploy --contract-address CONTRACT_ADDRESS

Contract found at 0x1ae8eea0dc265fb7f160dae62cc8912686d8a9ed78e821fbdd8bcedc54c06d0f Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L288-L292

Sending a Transaction

We can now send a transaction to the network. We will mint funds in the public domain. To form and submit the transaction we will use thesend command ofaztec-cli . Thesend command expect the function name as the first unnamed argument and the following named arguments:

- --args
-
    - The list of arguments to the function call.
- --contract-artifact
-
    - The artifact of the contract to call.
- --contract-address
-
    - The deployed address of the contract to call.
- --private-key
-
    - The private key of the sender.

send % aztec-cli send mint_public \ --args ADDRESS

543

\ --contract-artifact TokenContractArtifact \ --contract-address CONTRACT_ADDRESS

\ --private-key PRIVATE_KEY

Transaction has been mined Transaction hash: 15c5a8e58d5f895c7e3017a706efbad693635e01f67345fa60a64a340d83c78c Status: mined Block number: 5 Block hash: 163697608599543b2bee9652f543938683e4cdd0f94ac506e5764d8b908d43d4 Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L304-L316 We called themint_public function and provided it with the 2 arguments it expects: the recipient's address and the amount to be minted. Make sure to replace all addresses in this command with yours.

The command output tells us the details of the transaction such as its hash and status. We can use this hash to query the receipt of the transaction at a later time:

get-tx-receipt % aztec-cli get-tx-receipt 15c5a8e58d5f895c7e3017a706efbad693635e01f67345fa60a64a340d83c78c

Transaction receipt: { "txHash" :

"15c5a8e58d5f895c7e3017a706efbad693635e01f67345fa60a64a340d83c78c" , "status" :

"mined" , "error" :

"" , "blockHash" :

"163697608599543b2bee9652f543938683e4cdd0f94ac506e5764d8b908d43d4" , "blockNumber" :

5 , "origin" :

"0x2337f1d5cfa6c03796db5539b0b2d5a57e9aed42665df2e0907f66820cb6eebe" } Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L337-L349

Calling an Unconstrained (View) Function

Now that themint_public tx has been settled we can call thebalance_of_public unconstrained function:

call % aztec-cli call balance_of_public -a ADDRESS -c TokenContractArtifact -ca CONTRACT_ADDRESS

View result: 543n Source code: yarn-project/end-to-end/src/cli_docs_sandbox.test.ts#L369-L373 Thecall command calls a read-only method on a contract, one that will not generate a transaction to be sent to the network. The arguments here are:

- --args
-
  - The address for which we want to retrieve the balance.
- --contract-artifact
-
  - The artifact of the contract we are calling.
- --contract-address
-
  - The address of the deployed contract

As you can see from the result, this address has a public balance of 543, as expected.Edit this page

Previous Creating Schnorr Accounts Next Sandbox Reference