Pre-genesis participants

This section contains details on the pre-genesis process and instructions on how participants can generate, sign, and submit each available type of pre-genesis transaction.

Genesis state

A chain always starts from an initial state (the 'genesis state'), which is defined in its genesis files. Among other things, these files define an initial allocation of tokens to various accounts, along with an initial validator set and stake distribution.

Initial token allocations are given in the chain'sbalances.toml file, and the initial validator set and stake distribution are derived as a result of applying the pre-genesis transactions given in the chain'stransactions.toml file.

To produce an initial validator set in a decentralized manner, network participants (i.e. users and operators) are given the opportunity to generate and submit a pre-genesis transaction in which they can initialize their validators and/or bond stake to a genesis validator.

Prospective pre-genesis transactions are then collected in a single location (typically a GitHub repo) where they can be reviewed by thenetwork coordinator and selected for inclusion in the genesis block. The finalized set of pre-genesis transactions are published in the chain configuration.tar.gz file used to bootstrap all nodes. Should validators with a sufficient amount of stake choose to proceed to launch the chain with the proposed configuration, it will signal social consensus of the legitimacy of the genesis state.

Pre-genesis network participants

The pre-genesis network participants are the entities that are participating in the genesis ceremony. The pre-genesis network participants are responsible for:

1. Generating their public and private keys
2. Submitting their public keys and addresses to the network coordinator
3. Generating their pre-genesis transactions
4. Signing their pre-genesis transactions
5. Submitting their pre-genesis transactions to the network coordinator

Notes:

Before getting to the pre-genesis transaction procedure in the next section, it's helpful to keep the following in mind:

Pre-genesis wallet

Whenever you perform a pre-genesis action that requires a public or private key (such as generating or signing pre-genesis transactions), the Namada client will attempt to find them in the pre-genesis wallet, which is located atBASE_DIR/pre-genesis/wallet.toml .

To perform wallet actions such as adding or removing keys with the pre-genesis wallet, use the--pre-genesis flag withnamadaw . For example, to generate a new key in the pre-genesis wallet:

```
namadaw

--pre-genesis

gen

--alias ALIAS
```

Implicit accounts

Unlike established and validator accounts (which exist on-chain)implicit accounts have no pre-genesis logic attached -- they simply exist by virtue of being part of Namada's address space. Possessing the private key that corresponds to an implicit account permits one to spend any tokens that may or may not be present at that address. Therefore, an implicit account can be given a genesis allocation in the chain'sbalances.toml file, by specifying its public key and an amount, but does not require any accompanying pre-genesis transaction.

Pre-genesis process

This section gives instructions on the order of operations and commands used by pre-genesis participants when creating their pre-genesis transaction(s). It only applies before network launch.

Generating keys

Before the genesis ceremony begins, the pre-genesis network participants must generate their public and private keys. The pre-genesis network participants must ensure that their private keys are kept secret and that their public keys are submitted to the network coordinator in due time.

This can be done through the namada cli:

ALIAS

"" namadaw

--pre-genesis

gen

--alias ALIAS After the user has entered their passwords and written down their mnemonic phrase, the Namada cli will save the keys to thepre-genesis folder inside thebase directory .

In order to print the keys, the user can run the following command:

namadaw

--pre-genesis

find

--alias ALIAS The network coordinator should specify the schema in which the pre-genesis network participants should submit their public keys and addresses. A toml file with the following schema is recommended:

[] "public-key" =

"ED25519_PK_PREFIXtpknam1qpjs6jrjuu5cj508ys7ezee4r40v8as6vz4j3ddc9qm68nfhf5n7clp4xse" "address" =

"tnam1qq5skuga54tfs7422hzz8sqvk3s6lhe2dg32jjhd" Remember, a public key begins withtpknam , whereas an address begins withtnam .

Submitting keys and addresses

After the network coordinator has published the location (conventionally a GitHub repository) for the pre-genesis network participants to submit their public keys, the participants are expected to submit their generated public keys (and their corresponding addresses) to the network coordinator in due time. The deadline will be set by the network coordinator.

Generating transactions

Participants are also responsible for generating the genesis-block transactions that set the initial state of the blockchain, including initializing genesis validator accounts and bonding to them. However, it is only possible to make transactions that require value once the balances of the keys become agreed upon (i.e. once thebalances.toml file has been published).

The available pre-genesis transactions are:

- initialize an established account
- initialize a validator from an established account
- bond stake to a genesis validator

More details on generating each type of pre-genesis transaction follow below.

Initialize an established account

You can initialize an established account pre-genesis using theinit-genesis-established-account command.

First, you must add the public key(s) you wish to associate with the established account to your pre-genesis wallet, if not already present.

You can add a public key to your wallet, separate from its corresponding private key, with the following command:

namadaw --pre-genesis add --value PUBLIC_KEY --alias ALIAS You can then generate the pre-genesis transaction that will initialize the account and write it to atoml file with the following command:

namadac

utils

init-genesis-established-account \ --path OUTPUT_FILE \ --aliases ALIAS Thealiases argument specifies the public key(s) to be associated with the account, and thepath argument specifies the output file.

Note: After initializing a pre-genesis established account you may proceed to "promoting" it to a pre-genesis validator account (see below), however you are not required to do so.

If theinit-genesis-established-account command is successful, you will see output similar to the following:

```
Derived
established
account
address:
{your
tnam
address}
IMPORTANT:
keep
a
note
of
this
address,
especially
if
you
plan
to
use
it
for
a
validator
account
in
the
future!
Wrote
genesis
tx
to:
```

{output

file

location} This will generate atransactions.toml file that contains the pre-genesis transaction that will establish the account. Thetransactions.toml file should look similar to:

[[established_account]] vp =

"vp_user" threshold =

1 public_keys = ["tpknam1qr872zwdvw4u4nkpl0ykmvhyvxw7j0u6g7ymz03d7he0jr3szkuwczddjp2"] You can also initialize the account as amulti-sig by providing multiple aliases as a comma-separated list and providing thethreshold parameter:

namadac

utils

init-genesis-established-account \ --path OUTPUT_FILE \ --aliases ALIAS1 , ALIAS2 , ALIAS3 \ --threshold THRESHOLD

Initialize a pre-genesis validator account

This step will require thebalances.toml file to have been published by the network coordinator. Note: To initialize a pre-genesis validator, you must first have followed the steps above to initialize a pre-genesis established account and generate a transactiontoml file.

The following command will generate a pre-genesis transaction which initializes a validator account:

Ensure BASE_DIR is set to the base directory

Ensure ESTABLISHED_ACCOUNT_ADDRESS is set

Ensure TX_FILE_PATH is set

EMAIL

"" DISCORD_HANDLE = ""

This is optional

AVATAR_URL

""

This is optional, and expects a URL to an image the size of 128x128 pixels

WEBSITE

""

This is optional

DESCRIPTION

""

This is optional

VALIDATOR_NAME

""

This is optional

SELF_BOND_AMOUNT

1000000

Set this to the amount of NAM you want to self-bond. This must be less than or equal to the amount of NAM allocated to the pre-genesis keys in the balances.toml that was published by the network coordinator.

VALIDATOR_ALIAS

"" IP_ADDRESS = "" namadac

utils

init-genesis-validator \ --address ESTABLISHED_ACCOUNT_ADDRESS \ --alias VALIDATOR_ALIAS \ --net-address

"IP_ADDRESS:26656" \ --commission-rate

0.05 \ --max-commission-rate-change

0.01 \ --self-bond-amount SELF_BOND_AMOUNT \ --email EMAIL \ --name VALIDATOR_NAME \ --discord-handle DISCORD_HANDLE \ --avatar AVATAR_URL \ --website WEBSITE \ --description DESCRIPTION \ --path TX_FILE_PATH TheSELF_BOND_AMOUNT must be less than or equal to the amount of NAM allocated to the pre-genesis keys in thebalances.toml that was published by the network coordinator. It will correspond to the self-bonded stake of the validator account at genesis.

The--net-address specifies the network address of the validator node given by "IP:PORT". It is also possible to use DNS names instead of IP:PORT addresses. This is used to bootstrap peer discovery between validators on network launch.

The--alias flag is the moniker name of the validator account in your wallet.

The--commission-rate and--max-commision-rate flags are the commission rate and the maximum permitted commission rate change of the validator account per epoch. See thevalidator docs for more info.

The--email flag is the email address of the validator account. This is used for the validator account to receive notifications from the network coordinator and other participants.

The optional flags--name ,--discord-handle ,--avatar ,--website and--description configure additional optional on-chain metadata that identify your validator.

The--address flag is thetnam... address of the established account that was generated in the previous section.

The--path flag is the path to thetransactions.toml file that was generated in the previous section. The above command will append the necessary transactions to the file given byTX_FILE_PATH and your newly generated validator keys will be written toBASE_DIR/pre-genesisVALIDATOR_ALIAS/validator-wallet.toml .

Successful execution will output the new validator address:

Validator account address: tnam1q8lsztqqhpjxdwzw88mqqc2mun7dvpxvas3v2dxk â ï¸ IMPORTANT: Make a secure backup of the validator keys created in this step. They are located in the folderBASE_DIR/pre-genesisVALIDATOR_ALIAS

Bond to a pre-genesis validator account

This step will require thebalances.toml file to have been published by the network coordinator. You can generate a pre-genesis transaction to bond tokens to a genesis validator (assuming you have tokens allocated to your account at genesis).

You can generate a pre-genesis transaction to bond tokens to a genesis validator (assuming you have tokens allocated to your account at genesis). To do so, first add your genesis account to your pre-genesis wallet using its mnemonic or private key.

Then, generate the pre-genesis bond transaction and write it to atoml file with:

namadac

utils

genesis-bond \ --validator TARGET_VALIDATOR \ --amount AMOUNT \ --source YOUR_PUBLIC_KEY \ --path OUTPUT_FILE

Signing transactions

Seebelow for instructions on how to sign with a Ledger hardware wallet.

For validator initialization and bonding transactions, after generating your pre-genesis transaction (but before submitting it for inclusion in the genesis block), you must sign it with the relevant private key. You can do this using thesign-genesis-txs command.

The command will check for the necessary private key in thepre-genesis wallet . If you are signing an init-validator transaction, it will also check for the validator keys atBASE_DIR/pre-genesisVALIDATOR_ALIAS/validator-wallet.toml .

To sign an init-validator transaction, use:

namadac

utils

sign-genesis-txs

--path INPUT_FILE --output OUTPUT_FILE --alias VALIDATOR_ALIAS When signing a bond transaction, thealias is omitted:

namadac

utils

sign-genesis-txs

--path INPUT_FILE --output OUTPUT_FILE Transactions that only initialize an established account do not need to be signed (attempting to sign one will simply output the original file).

Submitting transactions

At this point, participants will submit their signed transactiontoml file(s) for review and possible inclusion in the genesis block. The network coordinator will provide specific instructions on when and how to do this, along with a submission deadline. Most often, a dedicated GitHub repo will be created where the files can be submitted via pull request. By convention, a directory for each pre-genesis network participant is created in the git repository. The signed transactionstoml file is then submitted to its respective directory.

Using a Ledger hardware wallet

You can sign your pre-genesis transactions using a Ledger hardware wallet.

First, refer to theHardware Wallet section for instructions on how to install the Namada app on your Ledger device and how to use it with the Namada CLI.

Once your Ledger is ready, connect it via USB and open the Namada Ledger app.

The general procedure of generating and signing pre-genesis transactions is the same as given above with a few minor variations.

Deriving your account

Before doing anything else, begin by deriving a Ledger account and adding it to the pre-genesis wallet using this command:

namadaw

--pre-genesis

derive

--use-device

--alias ALIAS (You will be asked to confirm the operation on your device.)

If successful, you should see the following output:

Output: Using HD derivation path m/44'/877'/0'/0'/0' Successfully added a key and an address with alias: "..." The default derivation path ism/44'/877'/0'/0'/0 . You can use the flag--hd-path to specify a different derivation path. Usenamadaw derive --help for more information.

You can list your wallet contents with:

namadaw

--pre-genesis

list You should see your address and public key listed under theALIAS you provided. Beside them should be the label(external) indicating that they are associated with a device. You will need to make note of your public key in order to complete the remaining steps.

Signing the pre-genesis transactiontoml

The flag--use-device is used to indicate that a Ledger will be used for signing. When attempting to sign, have the device connected to your computer with the Namada Ledger app open.

Ensure the 'expert mode' toggle in the Ledger app is set todisabled .

Signing aninitialize-validator

pre-genesis transaction

To create a pre-genesis initialize-validator transaction, begin by using the Ledger account public key toinitialize an established account . In this command,ALIAS should be the alias given to your Ledger account duringderivation :

namadac

utils

init-genesis-established-account \ --path OUTPUT_FILE \ --aliases ALIAS Proceed withinitializing a validator account according to the usual procedure.

Finally, sign using the following command:

namadac

utils

sign-genesis-txs \ --path INPUT_FILE \ --output OUTPUT_FILE \ --alias VALIDATOR_ALIAS \ --use-device It's important to note that both the Ledger and the validator soft wallet created when initializing your validator account are required to sign the transaction.

- The Ledger account is checked against the public key in the[[established_account]]
- section and used to generate the corresponding signature
- The validator soft wallet (located insideBASE_DIR/pre-genesisVALIDATOR_ALIAS/
-) is used to generate the corresponding
- signatures ([validator_account.protocol_key]
- ,[validator_account.tendermint_node_key]
- , etc.)

If either is not accessible, signing will fail.

Signing abond

pre-genesis transaction

To create a pre-genesis bond transaction, provide the public key associated with your Ledger account as theSOURCE :

```
namadac

utils

genesis-bond \ --validator TARGET_VALIDATOR \ --source SOURCE \ --amount AMOUNT \ --path OUTPUT_FILE
```

Signing the transaction is very similar to using a soft wallet:

```
namadac

utils

sign-genesis-txs \ --path INPUT_FILE \ --output OUTPUT_FILE \ --use-device
```

Troubleshooting

Refer to theTroubleshooting section of the Hardware Wallet page for help resolving common errors.

The network coordinator Setting up a local network