# #

Tx

Tx module allows you to sign or broadcast transactions

# #

Available Commands

Name Description sign Sign transactions generated offline broadcast Broadcast a signed transaction to the networkmultisign Sign the same transaction by multiple accounts tx Query for a transaction by hash in a committed blocktxs Search for transactions that match the exact given events where results are paginated

# #

iris tx sign

Sign transactions in generated offline file. The file created with the --generate-only flag.

iris tx sign< file> [flags]

#

Flags

Name, shorthand Type Required Default Description --append bool true true Attach a signature to an existing signature. --from string true

Key name for signature --offline bool true

Offline mode. --signature-only bool true

Print only the generated signature, then exit --multisig string

true Address of the multisig account on behalf of which the transaction shall be signed

#

Generate an offline tx

TIP

You can generate any type of txs offline by appending the flag--generate-only We use a transfer tx in the following examples:

iris tx bank send iaa1w9lvhwlvkwqvg08q84n2k4nn896u9pqx93velx iaa15uys54epmd2xzhcn32szps56wvev40tt908h62 10iris --chain-id= irishub --generate-only Theunsigned.json should look like:

{ "type" : "cosmos-sdk/StdTx" , "value" : { "msg" : [{ "type" : "cosmos-sdk/MsgSend" , "value" : { "from_address" : "iaa1w9lvhwlvkwqvg08q84n2k4nn896u9pqx93velx" , "to_address" : "iaa15uys54epmd2xzhcn32szps56wvev40tt908h62" , "amount" : [{ "denom" : "iris" , "amount" : "10" }] } }] , "fee" : { "amount" : [ ] , "gas" : "200000" } , "signatures" : null , "memo" : "" } }

#

Sign tx offline

iris tx sign unsigned.json--name

< key-name>

signed.tx Thesigned.json should look like:

{ "type" : "auth/StdTx" , "value" : { "msg" : [{ "type" : "cosmos-sdk/Send" , "value" : { "inputs" : [{ "address" : "iaa106nhdckyf996q69v3qdxwe6y7408pvyvyxzhxh" , "coins" : [{ "denom" : "uiris" , "amount" : "1000000" }] }] , "outputs" : [{ "address" : "iaa1893x4l2rdshytfzvfpduecpswz7qtpstevr742" , "coins" : [{ "denom" : "uiris" , "amount" : "1000000" }] }] } }] ,

"fee" : { "amount" : [{ "denom" : "uiris" , "amount" : "4000000" }] , "gas" : "200000" } , "signatures" : [{ "pub_key" : { "type" : "tendermint/PubKeySecp256k1" , "value" : "Auouudrg0P86v2kq2lykdr97AJYGHyD6BJXAQtjR1gzd" } , "signature" : "sJewd6lKjma49rAiGVfdT+V0YYerKNx6ZksdumVCvuItqGm24bEN9msh7IJ12Sil1lYjqQjdAcjVCX/77FKlIQ==" , "account_number" : "0" , "sequence" : "3" }] , "memo" : "test" } } Note thesignature in thesigned.json should no longer be empty after signing.

Now it's ready tobroadcast the signed tx to the IRIS Hub.

#

iris tx broadcast

This command is used to broadcast an offline signed transaction to the network.

#

Broadcast offline signed transaction

iris tx broadcast signed.json --chain-id= irishub

#

iris tx multisign

Sign a transaction by multiple accounts. The tx could be broadcasted only when the number of signatures meets the multisig-threshold.

iris tx multisign< file> < key-name> < [signature] .. .> [flags]

#

Generate an offline tx by multisig key

TIP

No multisig key?Create one iris tx bank send< from> < to> 10iris--fees = 0 .3iris --chain-id= irishub--from = < multisig-keyname> --generate-only> unsigned.json

#

Sign the multisig tx

#

Query the multisig address

iris keys show< multisig-keyname>

#

Sign theunsigned.json

Assume the multisig-threshold is 2, here we sign theunsigned.json by 2 of the signers

Sign the tx by signer-1:

iris tx sign unsigned.json--from

< signer-keyname-1

--chain-id= irishub--multisig = < multisig-address> --signature-only> signed-1.json Sign the tx by signer-2:

iris tx sign unsigned.json--from

< signer-keyname-2

--chain-id= irishub--multisig = < multisig-address> --signature-only> signed-2.json

#

Merge the signatures

Merge all the signatures intosigned.json

iris tx multisign --chain-id= irishub unsigned.json< multisig-keyname> signed-1.json signed-2.json> signed.json Now you canbroadcast the signed tx .

#

iris query tx

iris query tx[hash] [flags]

#

iris query txs

iris query txs--events 'message.sender=&message.action=xxxx' --page 1 --limit 30 Among the possible values ofmessage.action :

module Msg action bank cosmos-sdk/MsgSend transfer cosmos-sdk/MsgMultiSend transfer distribution cosmos-sdk/MsgModifyWithdrawAddress set_withdraw_address cosmos-sdk/MsgWithdrawValidatorCommission withdraw_commission cosmos-sdk/MsgWithdrawDelegatorReward withdraw_rewards gov cosmos-sdk/MsgSubmitProposal submit_proposal cosmos-sdk/MsgDeposit proposal_deposit cosmos-sdk/MsgVote proposal_vote stake cosmos-sdk/MsgCreateValidator create_validator cosmos-sdk/MsgEditValidator edit_validator cosmos-sdk/MsgDelegate delegate cosmos-sdk/MsgBeginRedelegate redelegate cosmos-sdk/MsgUndelegate unbond slashing cosmos-sdk/MsgUnjail unjail coinswap irismod/MsgSwapOrder swap irismod/MsgAddLiquidity add_liquidity irismod/MsgRemoveLiquidity remove_liquidity htlc irismod/MsgCreateHTLC create_htlc irismod/MsgClaimHTLC claim_htlc irismod/MsgRefundHTLC refund_htlc nft irismod/MsgIssueDenom issue_denom irismod/MsgMintNFT mint_nft irismod/MsgBurnNFT burn_nft irismod/MsgTransferNFT transfer_nft irismod/MsgEditNFT edit_nft record irismod/MsgCreateRecord create_record token irismod/MsgIssueToken issue_token irismod/MsgEditToken edit_token irismod/MsgTransferTokenOwner transfer_token_owner irismod/MsgMintToken mint_token