

# Send Tokens

## Testnet Faucet

On a testnet, getting tokens is usually done via a faucet. You can get tokens for testing purposes using the [Secret Network faucet](#).

## Query Account Balance

After receiving tokens to your address, you can view your account's balance by typing:

...

Copy `secretcliquerybankbalances`

...

...

Copy

```
secretcli query bank balances
```

```
secret1kcy20p0cs2wakeqz00xgs5m0cmj65283xqmvfs | jq
```

```
{ "balances": [ { "denom": "usct", "amount": "99942000" } ], "pagination": { "next_key": null, "total": "0" } }
```

...

Get your using:

...

Copy

## Use the -a flag to display the address only

```
secretclikeysshow-a
```

...

You can also supply your address with the following command:

...

Copy

**This grabs the desired account address and uses it as an input to the**

```
secretcli query bank balances command
```

```
secretcliquerybankbalances(secretclikeysshow-a)
```

...

Note: When querying an account balance with zero tokens, you will get the error: No account with address was found in the state. This can also happen if you fund the account before your node is fully synced. These are both normal.

## Send Tokens

Use the following command to send tokens from one account to another:

...

Copy secretclitxbanksend10uscr

...

Note: The amount argument accepts the format. You may want to cap the maximum gas consumed by transactions via the --gas flag.

If you pass --gas=auto , the gas supply is automatically estimated before transaction execution.

Inaccurate gas estimates may occur in-between the end of the simulation and the actual execution of a transaction. An adjustment needs to be applied on top of the original estimate for the transaction to be broadcasted successfully. Adjustment are controlled via the --gas-adjustment flag, with a default value of 1.0.

To view updated balances of origin and destination accounts use:

...

Copy secretcli query bank balances secretcli query bank balances

...

Flags

Height

You can also check balances at any block height using the --height flag:

...

Copy secretcli query bank balances --height=

...

Example

...

Copy secretcli query bank balances \ secret1kcy20p0cs2wakeqz00xgs5m0cmj65283xqmvfs \ --height=3415892|jq

...

Example Output

...

Copy { "balances": [ { "denom": "uscr", "amount": "2000" } ], "pagination": { "next\_key": null, "total": "0" } }

...

Note

You can add a note (previously called 'memo') to any transaction using the --note flag:

...

Copy secretclitxbanksend \ \ \ \ --note"

...

...

Copy secretcli tx bank send \ secret1kcy20p0cs2wakeqz00xgs5m0cmj65283xqmvfs \ secret193w2cdzhjt9vf9ehw9clmmyg5jvnj7dslaqc0 \ 10uscr \ --note 'Hello this is a test' | jq

confirm transaction before signing and broadcasting [y/N]: y

```
{ "height": "0", "txhash": "9E41B419CAFDc245706B0B6C689299AFEF932F2864F88F6FBA417DE516F39B2A",
"codespace": "", "code": 0, "data": "", "raw_log": [], "logs": [], "info": "", "gas_wanted": "0", "gas_used": "0", "tx": null,
"timestamp": "", "events": [] }
```

...

Dry Run

You can simulate a transaction without actually broadcasting it by appending the `--dry-run` flag:

...

```
Copy secretclitxbanksend\\ 10uscr \ --chain-id= \ --dry-run
```

...

...

```
Copy // Some code
```

...

### Generate Only

Furthermore, you can build a transaction and print its JSON format to STDOUT by appending `--generate-only` to the list of arguments:

...

```
Copy secretcli tx bank send 10uscr \ --chain-id= \ --generate-only > unsignedSendTx.json
```

...

...

```
Copy secretcli tx sign \ --chain-id= \ --from= \ unsignedSendTx.json > signedSendTx.json
```

...

Note: The `--generate-only` flag prevents `secretcli` from accessing the local keybase. When the flag is supplied must be an address.

### Validate Signatures

You can validate transaction signatures by typing the following:

...

```
Copy secretcli tx sign --validate-signatures --from=signedSendTx.json
```

...

### Broadcast Signed Transaction

You can broadcast the signed transaction to a node by providing the JSON file using:

...

```
Copy secretcli tx broadcast --node=signedSendTx.json
```

...

Last updated 1 year ago On this page \* [Testnet Faucet](#) \* [Query Account Balance](#) \* [Send Tokens](#) \* [Flags](#) \* [Height](#) \* [Note](#) \* [Dry Run](#) \* [Generate Only](#)

Was this helpful? [Edit on GitHub](#) [Export as PDF](#)