Shuttle

Release v0.3.0

Meheret Tesfaye

Jun 23, 2020

CONTENTS

CHAPTER

ONE

SHUTTLE

Cryptocurrencies were created to make it possible for advanced, encrypted payments to be made between two or more people digitally, without the parties involved having to trust each other for the payment be completed. In other words, cryptocurrencies make it possible to send money reliably to other people over the internet without the money being double spent, and without people getting scammed out of their money when they try to make these digital payments.

Note: Hash Time Lock Contracts (HTLCs) are a perfect example of a payment technology for cryptocurrencies which makes all of the aforementioned things possible.

Shuttle is a python library for cross-chain atomic swaps between the networks of two cryptocurrencies. Cross-chain atomic swaps are the cheapest and most secure way to swap cryptocurrencies. It's a brand new decentralized payment environment based on Hash Time Lock Contracts (HTLCs) protocol.

CHAPTER

TWO

WHAT IS A HTLC?

A Hash Time Lock contract (HTLC) is essentially a type of payment in which two people agree to a financial arrangement where one party will pay the other party a certain amount of cryptocurrency, such as Bitcoin or Bytom assets. However, because these contracts are Time Locked, the receiving party only has a certain amount of time to accept the payment, otherwise the money can be returned to the sender.

Hash time lock contracts can help to eliminate the need for third parties in contracts between two parties. Third parties that are often involved in contracts are lawyers, banks, etc. Lawyers are often required to draw up contracts, and banks are often required to help store money and then transfer it to the receiving party in the contract.

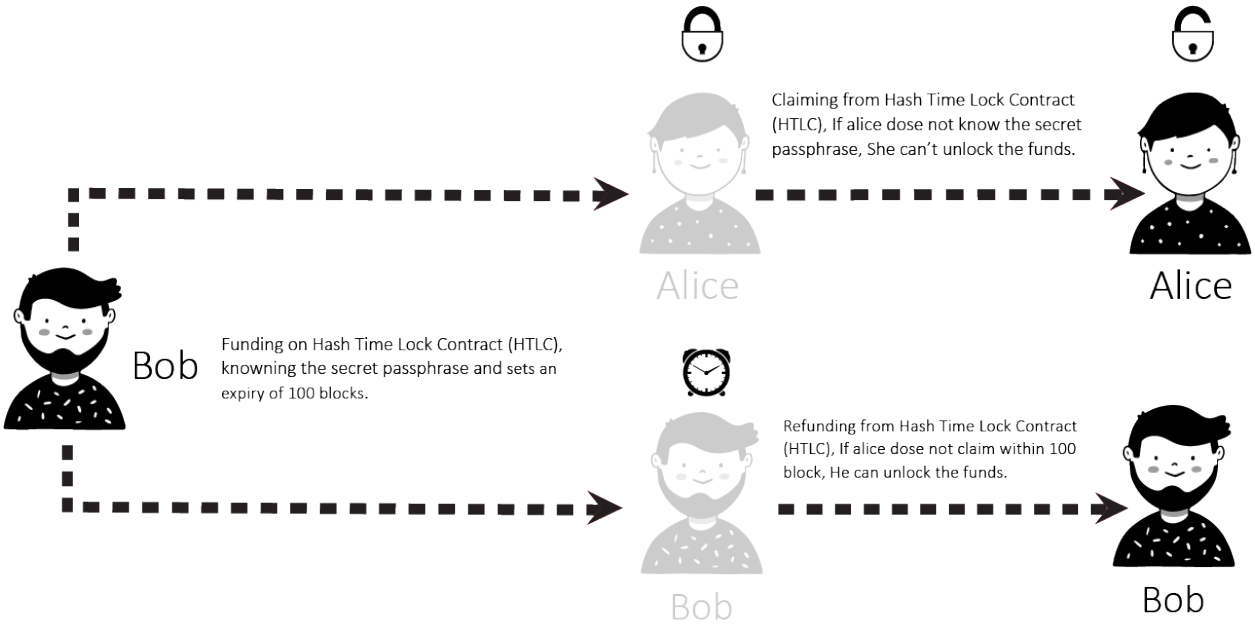
With hash time lock contracts, two parties could hypothetically set up contracts and transfer money without the need for third parties. This is because the sending party could create the conditional payment, and then the receiving party could agree to it, receive it, and help validate the transaction in the process.

This could potentially revolutionize the way that many businesses interact with one another and dramatically speed up the time that it takes for business deals to be set up.

2.1 How do HTLC work?

The way that Hash Time Lock Contracts work is that the person who will be making the payment sets up a specific hash, which represents the amount of money that will be paid. To receive the payment, the recipient will have to create a cryptographic proof of payment, and he or she will have to do this within the specified amount of time. The amount of time that the recipient has to accept the payment can vary significantly from one Time Locked contract to the next. If the recipient meets the deadline, then the money will be theirs, if he or she fails to meet the deadline, it won't. So, there is an often a lot at stake when it comes to meeting deadlines from hash Time Locked contracts, when cryptocurrencies are being exchanged.

The amount of time that the recipient has to accept the payment can vary significantly from one Time Locked contract to the next. If the recipient meets the deadline, then the money will be theirs, if he or she fails to meet the deadline, it won't. So, there is an often a lot at stake when it comes to meeting deadlines from hash Time Locked contracts, when cryptocurrencies are being exchanged.

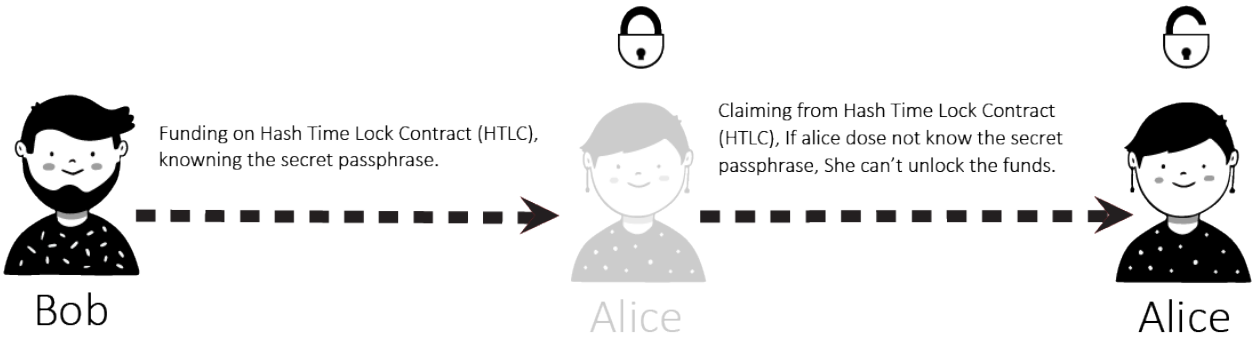


A Hash Time Lock Contract or HTLC is a class of payments that uses Hash Locked and Time Locked to require that the receiver of a payment either acknowledge receiving the payment prior to a deadline by generating cryptographic proof of payment or forfeit the ability to claim the payment, returning(refunding) it to the payer.

Hash Time Lock Contracts (HTLCs) allow payments to be securely routed across multiple payment channels which is super important because it is not optimal for a person to open a payment channel with everyone he/she is transacting with.

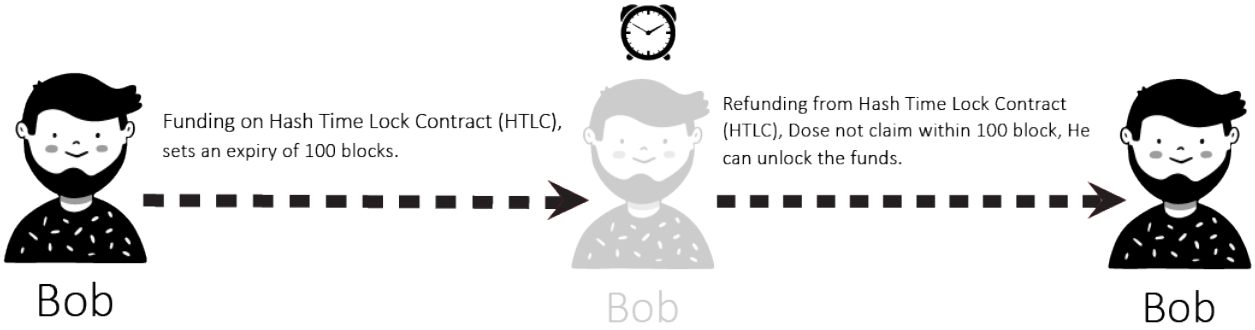
2.1.1 Hash Locked

A Hash Locked functions like "two-factor authentication" (2FA). It requires the intended recipient to provide the correct secret passphrase to claim the funds.



2.1.2 Time Locked

A Time Locked adds a "timeout" expiration date to a payment. It requires the intended recipient to claim the funds prior to the expiry. Otherwise, the transaction defaults to enabling the original sender of funds to claim a refund.



2.2 Benefits of HTLC

There are many benefits to these types of contracts. First, because they are time sensitive, it prevents the person who is making the payment from having to wait indefinitely to find out whether or not his or her payment goes through. Second, the person who makes the payment will not have to waste his or her money if the payment is not accepted. It will simply be returned.

2.2.1 Time Sensitivity

The time sensitive nature of the transaction prevents the sender from having to wait forever to find out whether their payment went through. If the time runs out, the funds will just be sent back to the sender, so they don't have to worry and can wait for the process to unfold.

2.2.2 Trustless system

As is the case with all smart contracts, trust is not needed as the rules are already coded into the contract itself. Hash Time Lock Contracts take this one step further by implementing a time limit for recipients to acknowledge the payment.

2.2.3 Validation of the blockchain

Transactions are validated because of the cryptographic proof of payment required by the receiver.

2.2.4 Private Information's

There are no complicated account setups or KYC/AML restrictions. Trade directly from your wallet with a counterparty of your choice. Only the parties involved know the details of the trade.

2.2.5 Trading across multiple cryptocurrencies

HTLC makes Cross-chain transactions easier and more secure than ever. Cross chain transactions are the next step in the evolution of cryptocurrency adoption. The easier it becomes to unite the hundreds of blockchain's that currently exist in silos, the faster the technology as a whole can begin to scale and achieve mass adoption.

CHAPTER

THREE

INSTALLING SHUTTLE

The easiest way to install Shuttle is via pip.

```
$ pip install pyshuttle
```

For the versions available, see the tags on this repository.

3.1 Development

We welcome pull requests. To get started, just fork this github repository, clone it locally, and run:

```
$ pip install -e . -r requirements.txt
```

Once you have installed, type `shuttle` to verify that it worked:

```
$ shuttle
Usage: shuttle [OPTIONS] COMMAND [ARGS]...

Options:
  -v, --version  Show Shuttle version and exit.
  -h, --help     Show this message and exit.

Commands:
  bitcoin  Select Bitcoin provider.
  bytom    Select Bytom provider.
```

3.2 Dependencies

Shuttle has the following dependencies:

- bytom-wallet-desktop - version 1.1.0 or greater.
- pip - To install packages from the Python Package Index and other indexes
- python3 version 3.6 or greater, python3-dev

CHAPTER

FOUR

COMMAND LINE INTERFACE (CLI)

After you have installed, type `shuttle` to verify that it worked:

```
$ shuttle
Usage: shuttle [OPTIONS] COMMAND [ARGS]...

Options:
  -v, --version  Show Shuttle version and exit.
  -h, --help     Show this message and exit.

Commands:
  bitcoin  Select Bitcoin provider.
  bytom    Select Bytom provider.
```

4.1 shuttle

```
shuttle [OPTIONS] COMMAND [ARGS]...
```

Options

-v, --version
Show Shuttle version and exit.

4.1.1 bitcoin

Select Bitcoin provider.

```
shuttle bitcoin [OPTIONS] COMMAND [ARGS]...
```

claim

Select Bitcoin claim transaction builder.

```
shuttle bitcoin claim [OPTIONS]
```

Options

-t, --transaction <transaction>
Set Bitcoin fund transaction id. [required]

-ra, --recipient-address <recipient_address>
Set Bitcoin recipient address. [required]

-a, --amount <amount>
Set Bitcoin amount to claim. [required]

-v, --version <version>
Set Bitcoin transaction version.

-n, --network <network>
Set Bitcoin network.

decode

Select Bitcoin transaction raw decoder.

```
shuttle bitcoin decode [OPTIONS]
```

Options

-r, --raw <raw>
Set Bitcoin transaction raw. [required]

fund

Select Bitcoin fund transaction builder.

```
shuttle bitcoin fund [OPTIONS]
```

Options

-sa, --sender-address <sender_address>
Set Bitcoin sender address. [required]

-a, --amount <amount>
Set Bitcoin amount to fund on HTLC. [required]

-b, --bytecode <bytecode>
Set Bitcoin HTLC bytecode. [required]

-v, --version <version>
Set Bitcoin transaction version.

-n, --network <network>
Set Bitcoin network.

htlc

Select Bitcoin Hash Time Lock Contract (HTLC) builder.

```
shuttle bitcoin htlc [OPTIONS]
```

Options

-sh, --secret-hash <secret_hash>
Set secret 256 hash. [required]

-ra, --recipient-address <recipient_address>
Set Bitcoin recipient address. [required]

-sa, --sender-address <sender_address>
Set Bitcoin sender address. [required]

-sq, --sequence <sequence>
Set Bitcoin sequence/expiration block.

-n, --network <network>
Set Bitcoin network.

refund

Select Bitcoin refund transaction builder.

```
shuttle bitcoin refund [OPTIONS]
```

Options

-t, --transaction <transaction>
Set Bitcoin fund transaction id. [required]

-sa, --sender-address <sender_address>
Set Bitcoin sender address. [required]

-a, --amount <amount>
Set Bitcoin amount to refund. [required]

-v, --version <version>
Set Bitcoin transaction version.

-n, --network <network>
Set Bitcoin network.

sign

Select Bitcoin transaction raw signer.

```
shuttle bitcoin sign [OPTIONS]
```

Options

-p, --private <private>
Set Bitcoin private key. [required]

-r, --raw <raw>
Set Bitcoin unsigned transaction raw. [required]

-b, --bytecode <bytecode>
Set Bitcoin witness HTLC bytecode.

-s, --secret <secret>
Set secret key.

-sq, --sequence <sequence>
Set Bitcoin sequence/expiration block.

-v, --version <version>
Set Bitcoin transaction version. [default: 2]

submit

Select Bitcoin transaction raw submitter.

```
shuttle bitcoin submit [OPTIONS]
```

Options

-r, --raw <raw>
Set signed Bitcoin transaction raw. [required]

4.1.2 bytom

Select Bytom provider.

```
shuttle bytom [OPTIONS] COMMAND [ARGS]...
```

claim

Select Bytom claim transaction builder.

```
shuttle bytom claim [OPTIONS]
```

Options

-t, --transaction <transaction>
Set Bytom fund transaction id. [required]

-rg, --recipient-guid <recipient_guid>
Set Bytom recipient GUID. [required]

-a, --amount <amount>
Set Bytom amount to claim. [required]

-as, --asset <asset>
Set Bytom asset id. [required]

-n, --network <network>
Set Bytom network.

decode

Select Bytom transaction raw decoder.

```
shuttle bytom decode [OPTIONS]
```

Options

-r, --raw <raw>
Set Bytom transaction raw. [required]

fund

Select Bytom unsigned transaction builder.

```
shuttle bytom fund [OPTIONS]
```

Options

-sg, --sender-guid <sender_guid>
Set Bytom sender GUID. [required]

-a, --amount <amount>
Set Bytom amount to fund on HTLC. [required]

-as, --asset <asset>
Set Bytom asset id. [required]

-b, --bytecode <bytecode>
Set Bytom HTLC bytecode. [required]

-n, --network <network>
Set Bytom network.

htlc

Select Bytom Hash Time Lock Contract (HTLC) builder.

```
shuttle bytom htlc [OPTIONS]
```

Options

-sh, --secret-hash <secret_hash>
Set secret 256 hash. [required]

-rp, --recipient-public <recipient_public>
Set Bytom recipient public key. [required]

-sp, --sender-public <sender_public>
Set Bytom sender public key. [required]

-sq, --sequence <sequence>
Set Bytom sequence/expiration block.

-n, --network <network>
Set Bytom network.

refund

Select Bytom refund transaction builder.

```
shuttle bytom refund [OPTIONS]
```

Options

-t, --transaction <transaction>
Set Bytom fund transaction id. [required]

-sg, --sender-guid <sender_guid>
Set Bytom sender GUID. [required]

-a, --amount <amount>
Set Bytom amount to refund. [required]

-as, --asset <asset>
Set Bytom asset id. [required]

-n, --network <network>
Set Bytom network.

sign

Select Bytom transaction raw signer.

```
shuttle bytom sign [OPTIONS]
```

Options

-xp, --xprivate <xprivate>
Set Bytom xprivate key. [required]

-r, --raw <raw>
Set Bytom unsigned transaction raw. [required]

-ac, --account <account>
Set Bytom derivation from account. [default: 1]

-c, --change <change>
Set Bytom derivation from change. [default: False]

-ad, --address <address>
Set Bytom derivation from address. [default: 1]

-b, --bytecode <bytecode>
Set Bytom witness HTLC bytecode.

-s, --secret <secret>
Set secret key.

-p, --path <path>
Set Bytom derivation from path.

-i, --indexes <indexes>
Set Bytom derivation from indexes.

submit

Select Bytom transaction raw submitter.

```
shuttle bytom submit [OPTIONS]
```

Options

-r, --raw <raw>
Set signed Bytom transaction raw. [required]

CHAPTER

FIVE

BITCOIN

Bitcoin is a cryptocurrency. It is a decentralized digital currency without a central bank or single administrator that can be sent from user to user on the peer-to-peer bitcoin network without the need for intermediaries.

5.1 Wallet

The implementation of Hierarchical Deterministic (HD) wallets generator for Bitcoin blockchain.

class shuttle.providers.bitcoin.wallet.**Wallet**(*network='testnet'*)

Bitcoin Wallet class.

Parameters **network** (*str*) – Bitcoin network, defaults to testnet.

Returns Wallet – Bitcoin wallet instance.

Note: Bitcoin has only two networks, `mainnet` and `testnet`.

address(*public_key=None*)

Get Bitcoin wallet address.

Parameters **public_key** (*str*) – Bitcoin address, default is None.

Returns str – Bitcoin address.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.address()
"mm357rHaKqVmhEhFFwUhz6mRVAHkJaDTKt"
```

balance(*address=None*, *network='testnet'*)

Get Bitcoin wallet balance.

Parameters

- **address** (*str*) – Bitcoin balance, default is None.
- **network** (*str*) – Bitcoin balance, default is testnet.

Returns int – Bitcoin balance.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
```

(continues on next page)

(continued from previous page)

```
>>> wallet.balance()
1000000
```

compressed(*public_key=None*)

Get Bitcoin wallet compressed public key.

Parameters **public_key** (*str*) – Bitcoin public key, default is None.

Returns str – Bitcoin compressed public key.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.compressed()
"03afa8301b068c2c184e0a3e77183dc95ec1130371c02ed172bec8f3bfbad6b173"
```

from_address(*address*)

Initiate Bitcoin wallet from address.

Parameters **address** (*str.*) – Bitcoin wallet private key.

Returns Wallet – Bitcoin wallet instance.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_address("mqLyrNDjpENRMZAoDpspH7kR9RtgvhWzYE")
<shuttle.providers.bitcoin.wallet.Wallet object at 0x040DA268>
```

from_passphrase(*passphrase*, *compressed=True*)

Initiate Bitcoin wallet from passphrase.

Parameters

- **passphrase** (*str.*) – Bitcoin wallet passphrase.
- **compressed** (*bool*) – Bitcoin public key compressed, default is True.

Returns Wallet – Bitcoin wallet instance.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_passphrase("meherett")
```

from_private_key(*private_key*, *compressed=True*)

Initiate Bitcoin wallet from private key.

Parameters

- **private_key** (*str.*) – Bitcoin wallet private key.
- **compressed** (*bool*) – Bitcoin public key compressed, default is True.

Returns Wallet – Bitcoin wallet instance.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_private_key(
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
<shuttle.providers.bitcoin.wallet.Wallet object at 0x040DA268>
```

hash(*public_key=None*)
Get Bitcoin wallet hash.

Parameters **public_key** (*str*) – Bitcoin hash, default is None.

Returns str – Bitcoin hash.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.hash()
"3c8acde1c7cf370d970725f13eff03bf74b3fc61"
```

p2pkh(*address=None*)
Get Bitcoin wallet p2pkh.

Parameters **address** (*str*) – Bitcoin p2pkh, default is None.

Returns str – Bitcoin p2pkh.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.p2pkh()
"76a9143c8acde1c7cf370d970725f13eff03bf74b3fc6188ac"
```

p2sh(*address=None*)
Get Bitcoin wallet p2sh.

Parameters **address** (*str*) – Bitcoin p2sh, default is None.

Returns str – Bitcoin p2sh.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.p2sh()
"a914a3c4995d9cd0303e5f89ee1433212c797d04ee5d87"
```

private_key()
Get Bitcoin wallet private key.

Returns str – Bitcoin private key.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.private_key()
"d4f5c55a45c004660b95ec833bb24569eba1559f214e90efa6e8d0b3afa14394"
```

public_key(*private_key=None*, *compressed=True*)
Get Bitcoin wallet public key.

Parameters

- **private_key** (*str*) – Bitcoin private key, default is None.
- **compressed** (*bool*) – Bitcoin public key compressed, default is True.

Returns str – Bitcoin public key.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett", compressed=False)
>>> wallet.public_key()

↪"04afa8301b068c2c184e0a3e77183dc95ec1130371c02ed172bec8f3bfbad6b17334244f64fe877d5e483690
↪"
```

uncompressed (*public_key=None*)
Get Bitcoin wallet uncompressed public key.

Parameters **public_key** (*str*) – Bitcoin public key, default is None.

Returns str – Bitcoin uncompressed public key.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.uncompressed()

↪"04afa8301b068c2c184e0a3e77183dc95ec1130371c02ed172bec8f3bfbad6b17334244f64fe877d5e483690
↪"
```

unspent (*address=None*, *network='testnet'*, *limit=15*)
Get Bitcoin wallet unspent transaction output.

Parameters

- **address** (*str*) – Bitcoin balance, default is None.
- **network** (*str*) – Bitcoin balance, default is testnet.
- **limit** (*int*) – Bitcoin balance, default is 15.

Returns list – Bitcoin unspent transaction outputs.

```
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet")
>>> wallet.from_passphrase("meherett")
>>> wallet.unspent()
[{'index': 0, 'hash':
↪'be346626628199608926792d775381e54d8632c14b3ce702f90639481722392c', 'output_
↪index': 1, 'amount': 12340, 'script':
↪'76a9146bce65e58a50b97989930e9a4ff1ac1a77515ef188ac'}]
```

5.2 Hash Time Lock Contract (HTLC)

Bitcoin Hash Time Lock Contract (HTLC).

class shuttle.providers.bitcoin.htlc.**HTLC** (*network='testnet'*)
Bitcoin Hash Time Lock Contract (HTLC) class.

Parameters **network** (*str*) – Bitcoin network, defaults to testnet.

Returns HTLC – Bitcoin HTLC instance.

Note: Bitcoin has only two networks, `mainnet` and `testnet`.

---

address()
Get Bitcoin Hash Time Lock Contract (HTLC) address.

Returns str – Bitcoin Hash Time Lock Contract (HTLC) address.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.init(secret_hash, recipient_address, sender_address, 100)
>>> htlc.address()
"2N729UBGZB3xjsGFRgKivy4bSjkaJGMVSpB"
```

bytecode()
Get Bitcoin htlc bytecode.

Returns str – Bitcoin Hash Time Lock Contract (HTLC) bytecode.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.init(secret_hash, recipient_address, sender_address, 100)
>>> htlc.bytecode()

↪"63aa20b9b9a0c47ecee7fd94812573a7b14afa02ec250dbdb5875a55c4d02367fcc2ab8876a9147b7c4431a43
↪"
```

from_bytecode(*bytecode*)
Initiate Bitcoin Hash Time Lock Contract (HTLC) from bytecode.

Parameters **bytecode** (*str.*) – Bitcoin bytecode.

Returns HTLC – Bitcoin Hash Time Lock Contract (HTLC) instance.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.from_bytecode(htlc_bytecode)
<shuttle.providers.bitcoin.htlc.HTLC object at 0x0409DAF0>
```

from_opcode(*opcode*)
Initiate Bitcoin Hash Time Lock Contract (HTLC) from opcode script.

Parameters **opcode** (*str.*) – Bitcoin opcode script.

Returns HTLC – Bitcoin Hash Time Lock Contract (HTLC) instance.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.from_opcode(htlc_opcode_script)
<shuttle.providers.bitcoin.htlc.HTLC object at 0x0409DAF0>
```

hash()
Get Bitcoin Hash Time Lock Contract (HTLC) hash.

Returns str – Bitcoin Hash Time Lock Contract (HTLC) hash.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.init(secret_hash, recipient_address, sender_address, 100)
>>> htlc.hash()
"a914971894c58d85981c16c2059d422bcde0b156d04487"
```

init(*secret_hash*, *recipient_address*, *sender_address*, *sequence=1000*)
Initialize Bitcoin Hash Time Lock Contract (HTLC).

Parameters

- **secret_hash** (*hash*) – secret sha-256 hash.
- **recipient_address** (*str*) – Bitcoin recipient address.
- **sender_address** (*str*) – Bitcoin sender address.
- **sequence** (*int*) – Bitcoin sequence number of expiration block, defaults to Bitcoin config sequence (15).

Returns HTLC – Bitcoin Hash Time Lock Contract (HTLC) instance.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.init(secret_hash, recipient_address, sender_address, 100)
<shuttle.providers.bitcoin.htlc.HTLC object at 0x0409DAF0>
```

opcode()

Get Bitcoin htlc opcode.

Returns str – Bitcoin Hash Time Lock Contract (HTLC) opcode.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> htlc = HTLC(network="testnet")
>>> htlc.init(secret_hash, recipient_address, sender_address, 100)
>>> htlc.opcode()
"OP_IF OP_HASH256␣
↪b9b9a0c47ecee7fd94812573a7b14afa02ec250dbdb5875a55c4d02367fcc2ab OP_
↪EQUALVERIFY OP_DUP OP_HASH160 7b7c4431a43b612a72f8229935c469f1f6903658 OP_
↪EQUALVERIFY OP_CHECKSIG OP_ELSE OP_5 OP_CHECKSEQUENCEVERIFY OP_DROP OP_DUP␣
↪OP_HASH160 6bce65e58a50b97989930e9a4ff1ac1a77515ef1 OP_EQUALVERIFY OP_
↪CHECKSIG OP_ENDIF"
```

5.3 Transaction

Bitcoin transaction in blockchain network.

class shuttle.providers.bitcoin.transaction.**Transaction**(*version=2*, *network='testnet'*)

Bitcoin Transaction class.

Parameters

- **version** (*int*) – Bitcoin transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns Transaction – Bitcoin transaction instance.

Note: Bitcoin has only two networks, `mainnet` and `testnet`.

---

fee()

Get Bitcoin transaction fee.

Returns int – Bitcoin transaction fee.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000)
>>> claim_transaction.fee()
576
```

hash()
Get Bitcoin transaction hash.

Returns str – Bitcoin transaction hash or transaction id.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> from shuttle.providers.bitcoin.transaction import FundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q",␣
↪1000)
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.hash()
"9cc0524fb8e7b2c5fecaee4eb91d43a3dc5cc18e9906abcb35a5732ff52efcc7"
```

json()
Get Bitcoin transaction json format.

Returns dict – Bitcoin transaction json format.

```
>>> from shuttle.providers.bitcoin.transaction import RefundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd", wallet,␣
↪10000)
>>> refund_transaction.json()
{"hex":
↪"02000000012c392217483906f902e73c4bc132864de58153772d7926896099816226663 4be0100000000ffffff
↪", "txid": "9cc0524fb8e7b2c5fecaee4eb91d43a3dc5cc18e9906abcb35a5732ff52efcc7
↪", "hash": "9cc0524fb8e7b2c5fecaee4eb91d43a3dc5cc18e9906abcb35a5732ff52efcc7
↪", "size": 117, "vsize": 117, "version": 2, "locktime": 0, "vin": [{"txid":
↪"be34662662819960892 6792d775381e54d8632c14b3ce702f90639481722392c", "vout":␣
↪1, "scriptSig": {"asm": "", "hex": ""}, "sequence": "4294967295"}], "vout":␣
↪[{"value": "0.00001000", "n": 0, "scriptPubKey": {"asm": "OP_HASH160␣
↪971894c58d85981c16c2059d422bcde0b156d044 OP_EQUAL", "hex":
↪"a914971894c58d85981c16c2059d422bcde0b156d04487", "type": "p2sh", "address
↪": "2N729UBGZB3xjsGFRgKivy4bSjkaJGMVSpB"}}, {"value": "0.00010662", "n": 1,
↪"scriptPubKey": {"asm": "OP_DUP OP_HASH160␣
↪6bce65e58a50b97989930e9a4ff1ac1a77515ef1 OP_EQUALVERIFY OP_CHECKSIG", "hex
↪": "76a9146bce65e58a50b97989930e9a4ff1ac1a77515ef188ac", "type": "p2pkh",
↪"address": "mqLyrNDjpENRMZAoDpspH7kR9RtgvhWzYE"}}]}
```

raw()
Get Bitcoin transaction raw.

Returns str – Bitcoin transaction raw.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000)
>>> claim_transaction.raw()

↪"02000000012c392217483906f902e73c4bc132864de58153772d792689609981622666 34be0100000000ffff
↪"
```

type()
Get Bitcoin signature transaction type.

Returns str – Bitcoin signature transaction type.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000)
>>> claim_transaction.type()
"bitcoin_claim_unsigned"
```

5.3.1 FundTransaction

class shuttle.providers.bitcoin.transaction.**FundTransaction**(*version=2*, *network='testnet'*)
Bitcoin FundTransaction class.

Parameters

- **version** (*int*) – Bitcoin transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns FundTransaction – Bitcoin fund transaction instance.

> **Warning:** Do not forget to build transaction after initialize fund transaction.

build_transaction(*wallet*, *htlc*, *amount*, *locktime=0*)
Build Bitcoin fund transaction.

Parameters

- **wallet** (bitcoin.wallet.Wallet) – Bitcoin sender wallet.
- **htlc** (bitcoin.htlc.HTLC) – Bitcoin hash time lock contract (HTLC).
- **amount** (*int*) – Bitcoin amount to fund.
- **locktime** (*int*) – Bitcoin transaction lock time, defaults to 0.

Returns FundTransaction – Bitcoin fund transaction instance.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> from shuttle.providers.bitcoin.transaction import FundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q",␣
↪1000)
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(wallet=sender_wallet, htlc=htlc,␣
↪amount=10000)
<shuttle.providers.bitcoin.transaction.FundTransaction object at 0x0409DAF0>
```

sign (*solver*)

Sign Bitcoin fund transaction.

Parameters **solver** (bitcoin.solver.FundSolver) – Bitcoin fund solver.

Returns FundTransaction – Bitcoin fund transaction instance.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> from shuttle.providers.bitcoin.transaction import FundTransaction
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q",␣
↪1000)
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> fund_solver = FundSolver(
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.sign(solver=fund_solver)
<shuttle.providers.bitcoin.transaction.FundTransaction object at 0x0409DAF0>
```

unsigned_raw()

Get Bitcoin unsigned fund transaction raw.

Returns str – Bitcoin unsigned fund transaction raw.

```
>>> from shuttle.providers.bitcoin.htlc import HTLC
>>> from shuttle.providers.bitcoin.transaction import FundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q",␣
↪1000)
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

5.3.2 ClaimTransaction

class shuttle.providers.bitcoin.transaction.**ClaimTransaction**(*network='testnet'*, *version=2*)

Bitcoin ClaimTransaction class.

Parameters

- **version** (*int*) – Bitcoin transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns Transaction – Bitcoin transaction instance.

> **Warning:** Do not forget to build transaction after initialize claim transaction.

build_transaction(*transaction_id*, *wallet*, *amount*, *locktime=0*)

Build Bitcoin claim transaction.

Parameters

- **transaction_id** (*str*) – Bitcoin fund transaction id to redeem.
- **wallet** (bitcoin.wallet.Wallet) – Bitcoin recipient wallet.
- **amount** (*int*) – Bitcoin amount to withdraw.
- **locktime** (*int*) – Bitcoin transaction lock time, defaults to 0.

Returns ClaimTransaction – Bitcoin claim transaction instance.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(transaction_id=
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪wallet=recipient_wallet, amount=10000)
<shuttle.providers.bitcoin.transaction.ClaimTransaction object at 0x0409DAF0>
```

sign(*solver*)

Sign Bitcoin claim transaction.

Parameters **solver** (bitcoin.solver.ClaimSolver) – Bitcoin claim solver.

Returns ClaimTransaction – Bitcoin claim transaction instance.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.solver import ClaimSolver
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_solver = ClaimSolver(recipient_wallet.private_key(), "Hello Meheret!
↪", "3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb",␣
↪recipient_wallet.address(), "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q", 1000)
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000)
>>> claim_transaction.sign(solver=claim_solver)
<shuttle.providers.bitcoin.transaction.ClaimTransaction object at 0x0409DAF0>
```

unsigned_raw()
Get Bitcoin unsigned claim transaction raw.

Returns str – Bitcoin unsigned claim transaction raw.

```
>>> from shuttle.providers.bitcoin.transaction import ClaimTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000)
>>> claim_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

5.3.3 RefundTransaction

class shuttle.providers.bitcoin.transaction.**RefundTransaction**(*version=2*, *network='testnet'*)
Bitcoin RefundTransaction class.

Parameters

- **version** (*int*) – Bitcoin transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns Transaction – Bitcoin transaction instance.

Warning: Do not forget to build transaction after initialize refund transaction.

build_transaction(*transaction_id*, *wallet*, *amount*, *locktime=0*)
Build Bitcoin refund transaction.

Parameters

- **transaction_id** (*str*) – Bitcoin fund transaction id to redeem.
- **wallet** (bitcoin.wallet.Wallet) – Bitcoin sender wallet.
- **amount** (*int*) – Bitcoin amount to withdraw.
- **locktime** (*int*) – Bitcoin transaction lock time, defaults to 0.

Returns RefundTransaction – Bitcoin refund transaction instance.

```
>>> from shuttle.providers.bitcoin.transaction import RefundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(transaction_id=
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪wallet=sender_wallet, amount=10000)
<shuttle.providers.bitcoin.transaction.RefundTransaction object at 0x0409DAF0>
```

sign(*solver*)
Sign Bitcoin refund transaction.

Parameters **solver** (bitcoin.solver.RefundSolver) – Bitcoin refund solver.

Returns RefundTransaction – Bitcoin refund transaction instance.

```
>>> from shuttle.providers.bitcoin.transaction import RefundTransaction
>>> from shuttle.providers.bitcoin.solver import RefundSolver
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett1234")
>>> refund_solver = RefundSolver(sender_wallet.private_key(),
↪"3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", sender_wallet.address(), 1000)
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd", sender_
↪wallet, 10000)
>>> refund_transaction.sign(solver=refund_solver)
<shuttle.providers.bitcoin.transaction.RefundTransaction object at 0x0409DAF0>
```

unsigned_raw()

Get Bitcoin unsigned refund transaction raw.

Returns str – Bitcoin unsigned refund transaction raw.

```
>>> from shuttle.providers.bitcoin.transaction import RefundTransaction
>>> from shuttle.providers.bitcoin.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_passphrase("meherett1234")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd", sender_
↪wallet, 10000)
>>> refund_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

5.4 Solver

Bitcoin solver.

5.4.1 FundSolver

class shuttle.providers.bitcoin.solver.**FundSolver**(*private_key*)

Bitcoin FundSolver class.

Parameters **private_key** (*str*) – Bitcoin sender private key.

Returns FundSolver – Bitcoin fund solver instance.

```
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
<shuttle.providers.bitcoin.solver.FundSolver object at 0x03FCCA60>
```

5.4.2 ClaimSolver

class shuttle.providers.bitcoin.solver.**ClaimSolver**(*private_key*, *secret*, *secret_hash=None*, *recipient_address=None*, *sender_address=None*, *sequence=1000*, *bytecode=None*)

Bitcoin ClaimSolver class.

Parameters

- **private_key** (*str*) – Bitcoin sender private key.
- **secret** (*str*) – Secret password/passphrase.
- **secret_hash** (*str*) – Secret witness password/passphrase hash, defaults to None.
- **recipient_address** (*str*) – Bitcoin witness recipient address, defaults to None.
- **sender_address** (*str*) – Bitcoin witness sender address, defaults to None.
- **sequence** (*int*) – Bitcoin witness sequence number(expiration block), defaults to 1000.
- **bytecode** (*str*) – Bitcoin witness HTLC bytecode, defaults to None.

Returns ClaimSolver – Bitcoin claim solver instance.

```
>>> from shuttle.providers.bitcoin.solver import ClaimSolver
>>> from shuttle.utils import sha256
>>> claim_solver = ClaimSolver(private_key=
↪"6bc3b581f3dea1963f9257ec2a0195969babee3704e6ba7cd2ec535140b9816f", secret=
↪"Hello Meheret!", secret_hash=sha256("Hello Meheret!".encode()).hex(),␣
↪recipient_address="muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", sender_address=
↪"mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q", sequence=1000)
<shuttle.providers.bitcoin.solver.ClaimSolver object at 0x03FCCA60>
```

5.4.3 RefundSolver

class shuttle.providers.bitcoin.solver.**RefundSolver**(*private_key*, *secret_hash=None*, *recipient_address=None*, *sender_address=None*, *sequence=1000*, *bytecode=None*)

Bitcoin RefundSolver class.

Parameters

- **private_key** (*str*) – Bitcoin sender private key.
- **secret_hash** (*str*) – Secret witness password/passphrase hash, defaults to None.
- **recipient_address** (*str*) – Bitcoin witness recipient address, defaults to None.
- **sender_address** (*str*) – Bitcoin witness sender address, defaults to None.
- **sequence** (*int*) – Bitcoin witness sequence number(expiration block), defaults to 1000.
- **bytecode** (*str*) – Bitcoin witness HTLC bytecode, defaults to None.

Returns RefundSolver – Bitcoin refund solver instance.

```
>>> from shuttle.providers.bitcoin.solver import RefundSolver
>>> from shuttle.utils import sha256
>>> refund_solver = RefundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b", secret_
↪hash=sha256("Hello Meheret!".encode()).hex(), recipient_address=
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", sender_address=
↪"mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q", sequence=1000)
<shuttle.providers.bitcoin.solver.RefundSolver object at 0x03FCCA60>
```

5.5 Signature

Bitcoin signature.

class shuttle.providers.bitcoin.signature.**Signature**(*network='testnet'*, *version=2*)

Bitcoin Signature class.

Parameters

- **version** (*int*) – Bitcoin transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns Transaction – Bitcoin transaction instance.

Note: Bitcoin has only two networks, `mainnet` and `testnet`.

---

fee()

Get Bitcoin transaction fee.

Returns int – Bitcoin transaction fee.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.fee()
678
```

hash()

Get Bitcoin signature transaction hash.

Returns str – Bitcoin signature transaction hash or transaction id.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
```

(continues on next page)

(continued from previous page)

```
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.hash()
"285ffc86ebece50f208bbfc1e72fb7c99991a3cf4d1b43cf93657838a4ae23ad"
```

json()

Get Bitcoin signature transaction json format.

Returns str – Bitcoin signature transaction json format.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.json()
{'hex':
↪'0200000001888be7ec065097d95664763f276d425552d735fb1d974ae78bf72106dca0f391010000006b48304
↪', 'txid': '285ffc86ebece50f208bbfc1e72fb7c99991a3cf4d1b43cf93657838a4ae23ad
↪', 'hash': '285ffc86ebece50f208bbfc1e72fb7c99991a3cf4d1b43cf93657838a4ae23ad
↪', 'size': 224, 'vsize': 224, 'version': 2, 'locktime': 0, 'vin': [{'txid':
↪'91f3a0dc0621f78be74a971dfb35d75255426d273f766456d9975006ece78b88', 'vout':␣
↪1, 'scriptSig': {'asm':
↪'3045022100c90f072ca3cd1ac446bbc952f007ddd82b930e416cfb7e07b0b56ec5065970b102202dcd28c92d9
↪03c56a6005d4a8892d28cc3f7265e5685b548627d59108973e474c4e26f69a4c84', 'hex':
↪'483045022100c90f072ca3cd1ac446bbc952f007ddd82b930e416cfb7e07b0b56ec5065970b102202dcd28c92
↪'}, 'sequence': '4294967295'}], 'vout': [{'value': '0.00010000', 'n': 0,
↪'scriptPubKey': {'asm': 'OP_HASH160␣
↪2bb013c3e4beb08421dedcf815cb65a5c388178b OP_EQUAL', 'hex':
↪'a9142bb013c3e4beb08421dedcf815cb65a5c388178b87', 'type': 'p2sh', 'address
↪': '2MwEDybGC34949zgzWX4M9FHmE3crDSUydP'}}, {'value': '0.00974268', 'n': 1,
↪'scriptPubKey': {'asm': 'OP_DUP OP_HASH160␣
↪64a8390b0b1685fcbf2d4b457118dc8da92d5534 OP_EQUALVERIFY OP_CHECKSIG', 'hex
↪': '76a91464a8390b0b1685fcbf2d4b457118dc8da92d553488ac', 'type': 'p2pkh',
↪'address': 'mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q'}}]}
```

raw()

Get Bitcoin signature transaction raw.

Returns str – Bitcoin signature transaction raw.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.raw()

↪"0200000001888be7ec065097d95664763f276d425552d735fb1d974ae78bf72106dca0f391010000006b48304
↪"
```

sign(*unsigned_raw*, *solver*)
Sign unsigned transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bitcoin unsigned transaction raw.
- **solver** (bitcoin.solver.FundSolver, bitcoin.solver.ClaimSolver, bitcoin.solver.RefundSolver) – Bitcoin solver

Returns FundSignature, ClaimSignature, RefundSignature – Bitcoin signature instance.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(unsigned_raw=bitcoin_fund_unsigned_raw, solver=fund_solver)
<shuttle.providers.bitcoin.signature.FundSignature object at 0x0409DAF0>
```

signed_raw()
Get Bitcoin signed transaction raw.

Returns str – Bitcoin signed transaction raw.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.signed_raw()

↪"eyJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkNzM1ZmIxZDk3NGFlN
↪"
```

type()
Get Bitcoin signature transaction type.

Returns str – Bitcoin signature transaction type.

```
>>> from shuttle.providers.bitcoin.signature import Signature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(private_key=
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> signature = Signature(network="testnet")
>>> signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
>>> signature.type()
"bitcoin_fund_signed"
```

5.5.1 FundSignature

class shuttle.providers.bitcoin.signature.**FundSignature**(*network='testnet'*, *version=2*)

Bitcoin FundSignature class.

Parameters

- **version** (*int*) – Bitcoin fund signature transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns FundSignature – Bitcoin fund signature instance.

sign(*unsigned_raw*, *solver*)

Sign unsigned fund transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bitcoin unsigned fund transaction raw.
- **solver** (bitcoin.solver.FundSolver) – Bitcoin fund solver.

Returns FundSignature – Bitcoin fund signature instance.

```
>>> from shuttle.providers.bitcoin.signature import FundSignature
>>> from shuttle.providers.bitcoin.solver import FundSolver
>>> bitcoin_fund_unsigned_raw =
↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTg4OGJlN2VjMDY1MDk3ZDk1NjY0NzYzZjI3NmQ0MjU1NTJkN
↪"
>>> fund_solver = FundSolver(
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b")
>>> fund_signature = FundSignature(network="testnet")
>>> fund_signature.sign(bitcoin_fund_unsigned_raw, fund_solver)
<shuttle.providers.bitcoin.signature.FundSignature object at 0x0409DAF0>
```

5.5.2 ClaimSignature

class shuttle.providers.bitcoin.signature.**ClaimSignature**(*network='testnet'*, *version=2*)

Bitcoin ClaimSignature class.

Parameters

- **version** (*int*) – Bitcoin claim signature transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns ClaimSignature – Bitcoin claim signature instance.

sign(*unsigned_raw*, *solver*)

Sign unsigned claim transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bitcoin unsigned claim transaction raw.
- **solver** (bitcoin.solver.ClaimSolver) – Bitcoin claim solver.

Returns ClaimSignature – Bitcoin claim signature instance.

```
>>> from shuttle.providers.bitcoin.signature import ClaimSignature
>>> from shuttle.providers.bitcoin.solver import ClaimSolver
>>> bitcoin_claim_unsigned_raw =
↪"eyJmZWUiOiA1NzYsICJyYXciOiAiMDIwMDAwMDAwMTUyYzIzZGM2NDU2N2IxY2ZhZjRkNzc2NjBjNzFjNzUxZjkwZ
↪"
>>> claim_solver = ClaimSolver(
↪"6bc3b581f3dea1963f9257ec2a0195969babee3704e6ba7cd2ec535140b9816f", "Hello␣
↪Meheret!", "3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb
↪",  "muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB",
↪"mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q", 1000)
>>> claim_signature = ClaimSignature(network="testnet")
>>> claim_signature.sign(unsigned_raw=bitcoin_claim_unsigned_raw,␣
↪solver=claim_solver)
<shuttle.providers.bitcoin.signature.ClaimSignature object at 0x0409DAF0>
```

5.5.3 RefundSignature

class shuttle.providers.bitcoin.signature.**RefundSignature**(*network='testnet'*, *version=2*)

Bitcoin RefundSignature class.

Parameters

- **version** (*int*) – Bitcoin refund signature transaction version, defaults to 2.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns RefundSignature – Bitcoin claim signature instance.

sign(*unsigned_raw*, *solver*)

Sign unsigned refund transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bitcoin unsigned refund transaction raw.
- **solver** (bitcoin.solver.RefundSolver) – Bitcoin refund solver.

Returns RefundSignature – Bitcoin refund signature instance.

```
>>> from shuttle.providers.bitcoin.signature import RefundSignature
>>> from shuttle.providers.bitcoin.solver import RefundSolver
>>> bitcoin_refund_unsigned_raw =
↪"eyJmZWUiOiA1NzYsICJyYXciOiAiMDIwMDAwMDAwMTUyYzIzZGM2NDU2N2IxY2ZhZjRkNzc2NjBjNzFjNzUxZjkwZ
↪"
>>> refund_solver = RefundSolver(
↪"92cbbc5990cb5090326a76feeb321cad01048635afe5756523bbf9f7a75bf38b",
↪"3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb",
↪"muTnffLDR5LtFeLR2i3WsKVfdyvzfyPnVB", "mphBPZf15cRFcL5tUq6mCbE84XobZ1vg7Q",␣
↪1000)
>>> refund_signature = RefundSignature(network="testnet")
>>> refund_signature.sign(unsigned_raw=bitcoin_refund_unsigned_raw,␣
↪solver=refund_solver)
<shuttle.providers.bitcoin.signature.RefundSignature object at 0x0409DAF0>
```

5.6 Remote Procedure Call (RPC)

Bitcoin remote procedure call.

shuttle.providers.bitcoin.rpc.**decoded_transaction_raw**(*transaction_raw*, *network='testnet'*, *timeout=60*)

Get decoded transaction raw.

Parameters

- **transaction_raw** (*str*) – Bitcoin transaction raw.
- **network** (*str*) – Bitcoin network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bitcoin decoded transaction raw.

```
>>> from shuttle.providers.bitcoin.rpc import decoded_transaction_raw
>>> decoded_transaction_raw(transaction_raw, "testnet")
{...}
```

shuttle.providers.bitcoin.rpc.**get_balance**(*address*, *network='testnet'*, *timeout=60*)

Get Bitcoin balance.

Parameters

- **address** (*str*) – Bitcoin address.
- **network** (*str*) – Bitcoin network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns int – Bitcoin balance.

```
>>> from shuttle.providers.bitcoin.rpc import get_balance
>>> get_balance(bitcoin_address, "mainnet")
25800000
```

shuttle.providers.bitcoin.rpc.**get_transaction_detail**(*transaction_id*, *network='testnet'*, *timeout=60*)

Get transaction detail.

Parameters

- **transaction_id** (*str*) – Bitcoin transaction hash or transaction id.
- **network** (*str*) – Bitcoin network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bitcoin transaction detail.

```
>>> from shuttle.providers.bitcoin.rpc import get_transaction_detail
>>> get_transaction_detail(transaction_id, "testnet")
{...}
```

shuttle.providers.bitcoin.rpc.**get_unspent_transactions**(*address*, *network='testnet'*, *include_script=True*, *limit=15*, *timeout=60*)

Get Bitcoin unspent transaction output (UTXO).

Parameters

- **address** (*str*) – Bitcoin address.
- **network** (*str*) – Bitcoin network, defaults to testnet.
- **include_script** (*bool*) – Bitcoin include script, defaults to True.
- **limit** (*int*) – Bitcoin utxo's limit, defaults to 15.
- **timeout** (*int*) – request timeout, default to 15.

Returns list – Bitcoin utxo's.

```
>>> from shuttle.providers.bitcoin.rpc import get_unspent_transactions
>>> get_unspent_transactions(bitcoin_address, "testnet")
[...]
```

shuttle.providers.bitcoin.rpc.**submit_payment**(*tx_raw*, *network='testnet'*, *timeout=60*)

Submit transaction raw to Bitcoin blockchain.

Parameters

- **tx_raw** (*str*) – Bitcoin transaction raw.
- **network** (*str*) – Bitcoin network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bitcoin decoded transaction raw.

```
>>> from shuttle.providers.bitcoin.rpc import submit_payment
>>> submit_payment(transaction_raw, "testnet")
{...}
```

5.7 Utils

Bitcoin Utils.

shuttle.providers.bitcoin.utils.**address_to_hash**(*address*, *network='testnet'*)

Get hash from address.

Parameters

- **address** (*str*) – Bitcoin address.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns P2pkhScript, P2shScript – Bitcoin p2pkh or p2sh script instance.

```
>>> from shuttle.providers.bitcoin.utils import address_to_hash
>>> address_to_hash("mrmtGq2HMmqAogSsGDjCtXUpxrb7rHThFH", "testnet")
"7b7c4431a43b612a72f8229935c469f1f6903658"
```

shuttle.providers.bitcoin.utils.**decode_transaction_raw**(*transaction_raw*)

Decode Bitcoin transaction raw.

Parameters **transaction_raw** (*str*) – Bitcoin transaction raw.

Returns dict – decoded Bitcoin transaction.

```
>>> from shuttle.providers.bitcoin.utils import decode_transaction_raw
>>> decode_transaction_raw(transaction_raw)
{...}
```

shuttle.providers.bitcoin.utils.**double_sha256**(*data*)
Double SHA256 hash.

Parameters **data** (*bytes*) – encoded data.

Returns bytearray – hashed double sha256.

```
>>> from shuttle.providers.bitcoin.utils import double_sha256
>>> double_sha256("Hello Meheret!".encode())
b"..."
```

shuttle.providers.bitcoin.utils.**fee_calculator**(*transaction_input=1*, *transaction_output=1*)
Bitcoin fee calculator.

Parameters

- **transaction_input** (*int*) – transaction input numbers, defaults to 1.
- **transaction_output** (*int*) – transaction output numbers, defaults to 1.

Returns int – Bitcoin fee.

```
>>> from shuttle.providers.bitcoin.utils import fee_calculator
>>> fee_calculator(2, 9)
1836
```

shuttle.providers.bitcoin.utils.**is_address**(*address*, *network=None*)
Check Bitcoin address.

Parameters

- **address** (*str*) – Bitcoin address.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns bool – Bitcoin valid/invalid address.

```
>>> from shuttle.providers.bitcoin.utils import is_address
>>> is_address(bitcoin_address, "testnet")
True
```

shuttle.providers.bitcoin.utils.**script_from_address**(*address*, *network='testnet'*)
Get script from address.

Parameters

- **address** (*str*) – Bitcoin address.
- **network** (*str*) – Bitcoin network, defaults to testnet.

Returns P2pkhScript, P2shScript – Bitcoin p2pkh or p2sh script instance.

```
>>> from shuttle.providers.bitcoin.utils import script_from_address
>>> script_from_address("mrmtGq2HMmqAogSsGDjCtXUpxrb7rHThFH", "testnet")
P2pkhScript('7b7c4431a43b612a72f8229935c469f1f6903658')
```

shuttle.providers.bitcoin.utils.**sha256**(*data*)
SHA256 hash.

Parameters **data** (*bytes*) – encoded data.

Returns bytearray – hashed sha256.

```
>>> from shuttle.providers.bitcoin.utils import sha256
>>> sha256("Hello Meheret!".encode())
b"..."
```

shuttle.providers.bitcoin.utils.**submit_transaction_raw**(*transaction_raw*)

Submit transaction raw to Bitcoin blockchain.

Parameters **transaction_raw** (*str*) – Bitcoin transaction raw.

Returns dict – Bitcoin transaction id, fee, type and date.

```
>>> from shuttle.providers.bitcoin.utils import submit_transaction_raw
>>> submit_transaction_raw(transaction_raw)
{...}
```

CHAPTER

SIX

BYTOM

Bytom is a protocol of multiple byte assets. Heterogeneous byte-assets operate in different forms on the Bytom Blockchain and atomic assets (warrants, securities, dividends, bonds, intelligence information, forecasting information and other information that exist in the physical world) can be registered, exchanged, gambled via Bytom.

6.1 Wallet

The implementation of Hierarchical Deterministic (HD) wallets generator for Bytom blockchain.

class shuttle.providers.bytom.wallet.**Wallet**(*network='testnet'*, *account=1*, *change=False*, *address=1*, *path=None*, *indexes=None*)

Bytom Wallet class.

Parameters

- **network** (*str*) – Bytom network, defaults to testnet.
- **account** (*int*) – Bytom derivation account, defaults to 1.
- **change** (*bool*) – Bytom derivation change, defaults to False.
- **address** (*int*) – Bytom derivation address, defaults to 1.
- **path** (*str*) – Bytom derivation path, defaults to None.
- **indexes** (*list*) – Bytom derivation indexes, defaults to None.

Returns Wallet – Bytom wallet instance.

Note: Bytom has only three networks, `mainnet`, `solonet` and `testnet`.

---

address()

Get Bytom wallet address.

Returns str – Bytom address.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.address()
"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7"
```

balance()
Get Bytom wallet balance.

Returns int – Bytom balance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.balance()
2450000000
```

expand_xprivate_key()
Get Bytom wallet expand xprivate key.

Returns str – Bytom expand xprivate key.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.expand_xprivate_key()

↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee5102416c643cfb46ab1ae5a524c
↪"
```

from_entropy(*entropy*)
Initiate Bytom wallet from entropy.

Parameters **entropy** (*str.*) – Bytom wallet entropy.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_entropy("...")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

from_mnemonic(*mnemonic*)
Initiate Bytom wallet from mnemonic.

Parameters **mnemonic** (*str.*) – Bytom wallet mnemonic.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

from_public_key(*public*)
Initiate Bytom wallet from public key.

Parameters **public** (*str.*) – Bytom wallet public key.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
```

(continues on next page)

(continued from previous page)

```
>>> wallet.from_public_key(
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

from_seed(*seed*)
Initiate Bytom wallet from seed.

Parameters **seed** (*str.*) – Bytom wallet seed.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_seed(
↪"baff3e1fe60e1f2a2d840d304acc98d1818140c79354a353b400fb019bfb256bc392d7aa9047adff1f14bce03
↪")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

from_xprivate_key(*xprivate_key*)
Initiate Bytom wallet from xprivate key.

Parameters **xprivate_key** (*str.*) – Bytom wallet xprivate key.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_xprivate_key(
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f9971136718
↪")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

from_xpublic_key(*xpublic_key*)
Initiate Bytom wallet from xpublic key.

Parameters **xpublic_key** (*str.*) – Bytom wallet xpublic key.

Returns Wallet – Bytom wallet instance.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_xpublic_key(
↪"16476b7fd68ca2acd92cfc38fa353e75d6103f828276f44d587e660a6bd7a5c5ef4490504bd2b6f9971136718
↪")
<shuttle.providers.bytom.wallet.Wallet object at 0x040DA268>
```

guid()
Get Bytom wallet blockcenter guid.

Returns str – Bytom blockcenter guid.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.guid()
"f0ed6ddd-9d6b-49fd-8866-a52d1083a13b"
```

indexes()
Get Bytom wallet derivation indexes.

Returns list – Bytom derivation indexes.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.indexes()
['2c000000', '99000000', '01000000', '00000000', '01000000']
```

path()
Get Bytom wallet derivation path.

Returns str – Bytom derivation path.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet", change=True, address=3)
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.path()
"m/44/153/1/1/3"
```

private_key()
Get Bytom wallet private key.

Returns str – Bytom private key.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.private_key()

↪"e07af52746e7cccd0a7d1fba6651a6f474bada481f34b1c5bab5e2d71e36ee515803ee0a6682fb19e279d8f4f
↪"
```

program()
Get Bytom wallet control program.

Returns str – Bytom control program.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.program()
"00142cda4f99ea8112e6fa61cdd26157ed6dc408332a"
```

public_key()
Get Bytom wallet public key.

Returns str – Bytom public key.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
```

(continues on next page)

(continued from previous page)

```
>>> wallet.public_key()

↪"e07af52746e7cccd0a7d1fba6651a6f474bada481f34b1c5bab5e2d71e36ee515803ee0a6682fb19e279d8f4f
↪"
```

seed()
Get Bytom wallet seed.

Returns str – Bytom seed.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.seed()

↪"baff3e1fe60e1f2a2d840d304acc98d1818140c79354a353b400fb019bfb256bc392d7aa9047adff1f14bce03
↪"
```

xprivate_key()
Get Bytom wallet xprivate key.

Returns str – Bytom xprivate key.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.xprivate_key()

↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪"
```

xpublic_key()
Get Bytom wallet xpublic key.

Returns str – Bytom xpublic key.

```
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="mainnet")
>>> wallet.from_mnemonic("indicate warm sock mistake code spot acid ribbon␣
↪sing over taxi toast")
>>> wallet.xpublic_key()

↪"16476b7fd68ca2acd92cfc38fa353e75d6103f828276f44d587e660a6bd7a5c5ef4490504bd2b6f997113671
↪"
```

6.2 Hash Time Lock Contract (HTLC)

Bytom Hash Time Lock Contract (HTLC).

class shuttle.providers.bytom.htlc.**HTLC**(*network='testnet'*)
Bytom Hash Time Lock Contract (HTLC) class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns HTLC – Bytom HTLC instance.

Note: Bytom has only three networks, `mainnet`, `solonet` and `testnet`.

address()
Get Bytom Hash Time Lock Contract (HTLC) address.

Returns str – Bytom Hash Time Lock Contract (HTLC) address.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.utils import sha256
>>> htlc = HTLC(network="mainnet")
>>> htlc.init(sha256("Hello Meheret!".encode()).hex(),
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",
↪"d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01", 1000,␣
↪False)
>>> htlc.address()
"bm1qk0r8l7ecl2vpacmg42wleptt6ckxhy7ljp5aanxczkv3r3rvy94q4a2zpc"
```

bytecode()
Get Bytom htlc bytecode.

Returns str – Bytom Hash Time Lock Contract (HTLC) bytecode.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.utils import sha256
>>> htlc = HTLC(network="mainnet")
>>> htlc.init(sha256("Hello Meheret!".encode()).hex(),
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",
↪"d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01", 1000,␣
↪False)
>>> htlc.bytecode()

↪"02e80320d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea012091ff7f525ff4087
↪"
```

hash()
Get Bytom Hash Time Lock Contract (HTLC) hash.

Returns str – Bytom Hash Time Lock Contract (HTLC) hash.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.utils import sha256
>>> htlc = HTLC(network="mainnet")
>>> htlc.init(sha256("Hello Meheret!".encode()).hex(),
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",
↪"d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01", 1000,␣
↪False)
```

(continues on next page)

(continued from previous page)

```
>>> htlc.hash()
"b3c67ffb38fa981ee368aa9dfc856bd62c6b93df9069deccd8159911c46c216a"
```

init(*secret_hash*, *recipient_public*, *sender_public*, *sequence=1000*, *use_script=False*)

Initialize Bytom Hash Time Lock Contract (HTLC).

Parameters

- **secret_hash** (*str*) – secret sha-256 hash.
- **recipient_public** (*str*) – Bytom recipient public key.
- **sender_public** (*str*) – Bytom sender public key.
- **sequence** (*int*) – Bytom sequence number(expiration block), defaults to Bytom config sequence.
- **use_script** (*bool*) – Initialize HTLC by using script, default to False.

Returns HTLC – Bytom Hash Time Lock Contract (HTLC) instance.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.utils import sha256
>>> htlc = HTLC(network="mainnet")
>>> htlc.init(secret_hash=sha256("Hello Meheret!".encode()).hex(), recipient_
↪public="91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",␣
↪sender_public=
↪"d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01",␣
↪sequence=1000, use_script=False)
<shuttle.providers.bytom.htlc.HTLC object at 0x0409DAF0>
```

opcode()

Get Bytom htlc opcode.

Returns str – Bytom Hash Time Lock Contract (HTLC) opcode.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.utils import sha256
>>> htlc = HTLC(network="mainnet")
>>> htlc.init(sha256("Hello Meheret!".encode()).hex(),
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",
↪"d4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01", 1000,␣
↪False)
>>> htlc.opcode()
"0xe803 0xd4351a0e743e6f10b35122ac13c0bb1445423a641754182d53f0677cc3d7ea01␣
↪0x91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2␣
↪0x3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb DEPTH␣
↪0x547a6416000000557aa888537a7cae7cac631f000000537acd9f6972ae7cac FALSE␣
↪CHECKPREDICATE"
```

6.3 Transaction

Bitcoin transaction in blockchain network.

class shuttle.providers.bytom.transaction.**Transaction**(*network='testnet'*, *guid=None*, *inputs=None*, *outputs=None*, *tx=None*)

Bytom Transaction class.

Parameters

- **network** (*str*) – Bytom network, defaults to testnet.
- **guid** (*str*) – Bytom blockcenter guid, defaults to None.
- **inputs** (*list*) – Bytom transaction inputs, defaults to None.
- **outputs** (*list*) – Bytom transaction outputs, defaults to None.
- **tx** (*dict*) – Bytom transaction, defaults to None.

Returns Transaction – Bytom transaction instance.

Note: Bytom has only three networks, `mainnet`. `solonet` and `testnet`.

---

fee()

Get Bitcoin transaction fee.

Returns int – Bitcoin transaction fee.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_mnemonic("hint excuse␣
↪upgrade sleep easily deputy erase cluster section other ugly limit")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> claim_transaction.fee()
10000000
```

hash()

Get Bytom transaction hash.

Returns str – Bytom transaction hash or transaction id.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
```

(continues on next page)

(continued from previous page)

```
>>> fund_transaction.hash()
"2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492"
```

json()

Get Bytom transaction json format.

Returns dict – Bytom transaction json format.

```
>>> from shuttle.providers.bytom.transaction import RefundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"481c00212c552fbdf537fcc88c1006a69bdd3130f593965f6ff4f91818a1c6e1", sender_
↪wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> refund_transaction.json()
{"hash": "2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492",
↪"status_fail": false, "size": 379, "submission_timestamp": 0, "memo": "",
↪"inputs": [{"script": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a",
↪"address": "bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 2450000000, "type": "spend"}], "outputs": [{"utxo_id":
↪"5edccebe497893c289121f9e365fdeb34c97008b9eb5a9960fe9541e7923aabc", "script
↪":
↪"01642091ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e220ac13c0bb1445423a6
↪", "address": "smart contract", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 1000, "type": "control"}, {"utxo_id":
↪"f8cfbb692db1963be88b09c314adcc9e19d91c6c019aa556fb7cb76ba8ffa1fa", "script
↪": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 2439999000, "type": "control"}], "fee": 10000000, "balances": [{"asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": "-10001000"}], "types": ["ordinary"]}
```

raw()

Get Bytom transaction raw.

Returns str – Bytom transaction raw.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_mnemonic("hint excuse␣
↪upgrade sleep easily deputy erase cluster section other ugly limit")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> claim_transaction.raw()

↪"070100010160015e7f2d7ecec3f61d30d0b2968973a3ac8448f0599ea20dce883b48c903c4d6e87fffffffff
↪"
```

signatures()

Get Bytom transaction signatures(signed datas).

Returns list – Bytom transaction signatures.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
>>> from shuttle.providers.bytom.solver import FundSolver
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_solver = FundSolver(
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.sign(solver=fund_solver)
>>> fund_transaction.signatures()
[[
↪'8ca69a01def05118866681bc7008971efcff40895285297e0d6bd791220a36d6ef85a11abc48438de21f0256c
↪']]
```

type()

Get Bitcoin signature transaction type.

Returns str – Bitcoin signature transaction type.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_mnemonic("hint excuse␣
↪upgrade sleep easily deputy erase cluster section other ugly limit")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> claim_transaction.type()
"bitcoin_claim_unsigned"
```

unsigned_datas(*detail=False*)

Get Bytom transaction unsigned datas with instruction.

Parameters **detail** (*bool*) – Bytom unsigned datas to see detail, defaults to False.

Returns list – Bytom transaction unsigned datas.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_transaction = FundTransaction(network="testnet")
```

(continues on next page)

(continued from previous page)

```
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.unsigned_datas()
[{'datas': ['38601bf7ce08dab921916f2c723acca0451d8904649bbec16c2076f1455dd1a2
↪'], 'public_key':
↪'91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2', 'network
↪': 'mainnet', 'path': 'm/44/153/1/0/1'}]
```

6.3.1 FundTransaction

class shuttle.providers.bytom.transaction.**FundTransaction**(*network='testnet'*)

Bytom FundTransaction class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns FundTransaction – Bytom fund transaction instance.

Warning: Do not forget to build transaction after initialize fund transaction.

build_transaction(*wallet*, *htlc*, *amount*, *asset='ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff'*)

Build Bytom fund transaction.

Parameters

- **wallet** (bytom.wallet.Wallet) – Bytom sender wallet.
- **htlc** (bytom.htlc.HTLC) – Bytom hash time lock contract (HTLC).
- **amount** (*int*) – Bytom amount to fund.
- **asset** (*str*) – Bytom asset id, defaults to BTM asset.

Returns FundTransaction – Bytom fund transaction instance.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(wallet=sender_wallet, htlc=htlc,␣
↪amount=10000)
<shuttle.providers.bytom.transaction.FundTransaction object at 0x0409DAF0>
```

sign(*solver*)

Sign Bytom fund transaction.

Parameters **solver** (bytom.solver.FundSolver) – Bytom fund solver.

Returns FundTransaction – Bytom fund transaction instance.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
```

(continues on next page)

(continued from previous page)

```
>>> from shuttle.providers.bytom.solver import FundSolver
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_solver = FundSolver(
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.sign(solver=fund_solver)
<shuttle.providers.bytom.transaction.FundTransaction object at 0x0409DAF0>
```

unsigned_raw()
Get Bytom unsigned fund transaction raw.

Returns str – Bytom unsigned fund transaction raw.

```
>>> from shuttle.providers.bytom.htlc import HTLC
>>> from shuttle.providers.bytom.transaction import FundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> htlc = HTLC(network="testnet").init(
↪"821124b554d13f247b1e5d10b84e44fb1296f18f38bbaa1bea34a12c843e0158",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> fund_transaction = FundTransaction(network="testnet")
>>> fund_transaction.build_transaction(sender_wallet, htlc, 10000)
>>> fund_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

6.3.2 ClaimTransaction

class shuttle.providers.bytom.transaction.**ClaimTransaction**(*network='testnet'*)
Bytom ClaimTransaction class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns ClaimTransaction – Bytom claim transaction instance.

Warning: Do not forget to build transaction after initialize claim transaction.

build_transaction(*transaction_id*, *wallet*, *amount*, *asset='ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff'*)
Build Bytom claim transaction.

Parameters

- **transaction_id** (*str*) – Bytom fund transaction id to redeem.
- **wallet** (bytom.wallet.Wallet) – Bytom recipient wallet.

- **amount** (*int*) – Bytom amount to withdraw.
- **asset** (*str*) – Bytom asset id, defaults to BTM asset.

Returns ClaimTransaction – Bytom claim transaction instance.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_mnemonic("hint excuse␣
↪upgrade sleep easily deputy erase cluster section other ugly limit")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(transaction_id=
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪wallet=recipient_wallet, amount=10000, asset=
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
<shuttle.providers.bytom.transaction.ClaimTransaction object at 0x0409DAF0>
```

sign(*solver*)

Sign Bytom claim transaction.

Parameters **solver** (bytom.solver.ClaimSolver) – Bytom claim solver.

Returns ClaimTransaction – Bytom claim transaction instance.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.solver import ClaimSolver
>>> from shuttle.providers.bytom.wallet import Wallet
>>> recipient_wallet = Wallet(network="testnet").from_mnemonic("hint excuse␣
↪upgrade sleep easily deputy erase cluster section other ugly limit")
>>> claim_solver = ClaimSolver(recipient_wallet.xprivate_key(), "Hello␣
↪Meheret!", "3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb
↪", recipient_wallet.public_key(),
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd",␣
↪recipient_wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> claim_transaction.sign(solver=claim_solver)
<shuttle.providers.bytom.transaction.ClaimTransaction object at 0x0409DAF0>
```

unsigned_raw()

Get Bytom unsigned claim transaction raw.

Returns str – Bytom unsigned claim transaction raw.

```
>>> from shuttle.providers.bytom.transaction import ClaimTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> wallet = Wallet(network="testnet").from_mnemonic("hint excuse upgrade␣
↪sleep easily deputy erase cluster section other ugly limit")
>>> claim_transaction = ClaimTransaction(network="testnet")
>>> claim_transaction.build_transaction(
↪"1006a6f537fcc4888c65f6ff4f91818a1c6e19bdd3130f59391c00212c552fbd", wallet,␣
↪10000, "ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> claim_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

6.3.3 RefundTransaction

class shuttle.providers.bytom.transaction.**RefundTransaction**(*network='testnet'*)
Bytom RefundTransaction class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns RefundTransaction – Bytom refund transaction instance.

> **Warning:** Do not forget to build transaction after initialize refund transaction.

build_transaction(*transaction_id*, *wallet*, *amount*, *asset='ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff'*)
Build Bytom refund transaction.

Parameters

- **transaction_id** (*str*) – Bytom fund transaction id to redeem.
- **wallet** (bytom.wallet.Wallet) – Bytom sender wallet.
- **amount** (*int*) – Bytom amount to withdraw.
- **asset** (*str*) – Bytom asset id, defaults to BTM asset.

Returns RefundTransaction – Bytom refund transaction instance.

```
>>> from shuttle.providers.bytom.transaction import RefundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(transaction_id=
↪"481c00212c552fbdf537fcc88c1006a69bdd3130f593965f6ff4f91818a1c6e1",␣
↪wallet=sender_wallet, amount=10000, asset=
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
<shuttle.providers.bytom.transaction.RefundTransaction object at 0x0409DAF0>
```

sign(*solver*)
Sign Bytom refund transaction.

Parameters **solver** (bytom.solver.RefundSolver) – Bytom refund solver.

Returns RefundTransaction – Bytom refund transaction instance.

```
>>> from shuttle.providers.bytom.transaction import RefundTransaction
>>> from shuttle.providers.bytom.solver import RefundSolver
>>> from shuttle.providers.bytom.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> refund_solver = RefundSolver(wallet.xprivate_key(),
↪"3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e", wallet.
↪public_key(), 1000)
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"481c00212c552fbdf537fcc88c1006a69bdd3130f593965f6ff4f91818a1c6e1", sender_
↪wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> refund_transaction.sign(solver=refund_solver)
<shuttle.providers.bytom.transaction.RefundTransaction object at 0x0409DAF0>
```

unsigned_raw()
Get Bytom unsigned refund transaction raw.

Returns str – Bytom unsigned refund transaction raw.

```
>>> from shuttle.providers.bytom.transaction import RefundTransaction
>>> from shuttle.providers.bytom.wallet import Wallet
>>> sender_wallet = Wallet(network="testnet").from_mnemonic("indicate warm␣
↪sock mistake code spot acid ribbon sing over taxi toast")
>>> refund_transaction = RefundTransaction(network="testnet")
>>> refund_transaction.build_transaction(
↪"481c00212c552fbdf537fcc88c1006a69bdd3130f593965f6ff4f91818a1c6e1", sender_
↪wallet, 10000,
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff")
>>> refund_transaction.unsigned_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

6.4 Solver

Bytom solver.

6.4.1 FundSolver

class shuttle.providers.bytom.solver.**FundSolver**(*xprivate_key*, *account=1*, *change=False*, *address=1*, *path=None*, *indexes=None*)

Bytom FundSolver class.

Parameters

- **xprivate_key** (*str*) – Bytom sender xprivate key.
- **account** (*int*) – Bytom derivation account, defaults to 1.
- **change** (*bool*) – Bytom derivation change, defaults to False.
- **address** (*int*) – Bytom derivation address, defaults to 1.
- **path** (*str*) – Bytom derivation path, defaults to None.
- **indexes** (*list*) – Bytom derivation indexes, defaults to None.

Returns FundSolver – Bytom fund solver instance.

```
>>> from shuttle.providers.bytom.solver import FundSolver
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f99711367189245
↪")
<shuttle.providers.bytom.solver.FundSolver object at 0x03FCCA60>
```

6.4.2 ClaimSolver

class shuttle.providers.bytom.solver.**ClaimSolver**(*xprivate_key*, *secret*, *secret_hash=None*, *recipient_public=None*, *sender_public=None*, *sequence=1000*, *bytecode=None*, *account=1*, *change=False*, *address=1*, *path=None*, *indexes=None*)

Bytom ClaimSolver class.

Parameters

- **xprivate_key** (*str*) – Bytom sender xprivate key.
- **secret** (*str*) – Secret password/passphrase.
- **secret_hash** (*str*) – Secret password/passphrase hash, defaults to None.
- **recipient_public** (*str*) – Bytom recipient public key, defaults to None.
- **sender_public** (*str*) – Bytom sender public key, defaults to None.
- **sequence** (*int*) – Bytom sequence number(expiration block), defaults to 1000.
- **bytecode** (*str*) – Bytom witness HTLC bytecode, defaults to None.
- **account** (*int*) – Bytom derivation account, defaults to 1.
- **change** (*bool*) – Bytom derivation change, defaults to False.
- **address** (*int*) – Bytom derivation address, defaults to 1.
- **path** (*str*) – Bytom derivation path, defaults to None.
- **indexes** (*list*) – Bytom derivation indexes, defaults to None.

Returns ClaimSolver – Bytom claim solver instance.

```
>>> from shuttle.providers.bytom.solver import ClaimSolver
>>> from shuttle.utils import sha256
>>> recipient_xprivate_key =
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f99711367189245
↪"
>>> claim_solver = ClaimSolver(xprivate_key=recipient_xprivate_key, secret="Hello␣
↪Meheret!", secret_hash=sha256("Hello Meheret!".encode()).hex(), recipient_
↪public="3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",␣
↪sender_public="91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2
↪", sequence=1000)
<shuttle.providers.bytom.solver.ClaimSolver object at 0x03FCCA60>
```

6.4.3 RefundSolver

class shuttle.providers.bytom.solver.**RefundSolver**(*xprivate_key*, *secret_hash=None*, *recipient_public=None*, *sender_public=None*, *sequence=1000*, *bytecode=None*, *account=1*, *change=False*, *address=1*, *path=None*, *indexes=None*)

Bytom RefundSolver class.

Parameters

- **xprivate_key** (*str*) – Bytom sender xprivate key.
- **secret_hash** (*str*) – Secret password/passphrase hash, defaults to None.
- **recipient_public** (*str*) – Bytom recipient public key, defaults to None.
- **sender_public** (*str*) – Bytom sender public key, defaults to None.
- **sequence** (*int*) – Bytom sequence number(expiration block), defaults to 1000.
- **bytecode** (*str*) – Bytom witness HTLC bytecode, defaults to None.
- **account** (*int*) – Bytom derivation account, defaults to 1.
- **change** (*bool*) – Bytom derivation change, defaults to False.
- **address** (*int*) – Bytom derivation address, defaults to 1.
- **path** (*str*) – Bytom derivation path, defaults to None.
- **indexes** (*list*) – Bytom derivation indexes, defaults to None.

Returns RefundSolver – Bytom refund solver instance.

```
>>> from shuttle.providers.bytom.solver import RefundSolver
>>> from shuttle.utils import sha256
>>> sender_xprivate_key =
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f99711367189245
↪"
>>> refund_solver = RefundSolver(xprivate_key=sender_xprivate_key, secret_
↪hash=sha256("Hello Meheret!".encode()).hex(), recipient_public=
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e", sender_
↪public="91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2",␣
↪sequence=1000)
<shuttle.providers.bytom.solver.RefundSolver object at 0x03FCCA60>
```

6.5 Signature

Bytom signature.

class shuttle.providers.bytom.signature.**Signature**(*network='testnet'*)

Bytom Signature class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns Transaction – Bytom transaction instance.

Note: Bytom has only three networks, `mainnet`, `solonet` and `testnet`.

---

fee()

Get Bitcoin transaction fee.

Returns int – Bitcoin transaction fee.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
```

(continues on next page)

(continued from previous page)

```
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.fee()
10000000
```

hash()

Get Bytom signature transaction hash.

Returns str – Bytom signature transaction hash or transaction id.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.hash()
"2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492"
```

json()

Get Bytom signature transaction json format.

Returns dict – Bytom signature transaction json format.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.json()
{"hash": "2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492",
↪"status_fail": false, "size": 379, "submission_timestamp": 0, "memo": "",
↪"inputs": [{"script": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a",
↪"address": "bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 2450000000, "type": "spend"}], "outputs": [{"utxo_id":
↪"5edccebe497893c289121f9e365fdeb34c97008b9eb5a9960fe9541e7923aabc", "script
↪":
↪"01642091ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e220ac13c0bb1445423a6
↪", "address": "smart contract", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 1000, "type": "control"}, {"utxo_id":
↪"f8cfbb692db1963be88b09c314adcc9e19d91c6c019aa556fb7cb76ba8ffa1fa", "script
↪": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": 2439999000, "type": "control"}], "fee": 10000000, "balances": [{"asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount
↪": "-10001000"}], "types": ["ordinary"]}
```

(continues on next page)

(continued from previous page)

raw()
Get Bytom signature transaction raw.

Returns str – Bytom signature transaction raw.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.raw()

↪"070100010160015e7f2d7ecec3f61d30d0b2968973a3ac8448f0599ea20dce883b48c903c4d6e87fffffffff
↪"
```

sign(*unsigned_raw*, *solver*)
Sign unsigned transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bytom unsigned transaction raw.
- **solver** (bytom.solver.FundSolver, bytom.solver.ClaimSolver, bytom.solver.RefundSolver) – Bytom solver

Returns FundSignature, ClaimSignature, RefundSignature – Bytom signature instance.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
<shuttle.providers.bytom.signature.FundSignature object at 0x0409DAF0>
```

signed_raw()
Get Bytom signed transaction raw.

Returns str – Bytom signed transaction raw.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
```

(continues on next page)

(continued from previous page)

```
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.signed_raw()

↪"eyJmZWUiOiA2NzgsICJyYXciOiAiMDIwMDAwMDAwMTJjMzkyMjE3NDgzOTA2ZjkwMmU3M2M0YmMxMzI4NjRkZTU4M
↪"
```

type()

Get Bytom signature transaction type.

Returns str – Bytom signature transaction type.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.type()
"bytom_fund_signed"
```

unsigned_datas(**args*, ***kwargs*)

Get Bytom transaction unsigned datas with instruction.

Returns list – Bytom transaction unsigned datas.

```
>>> from shuttle.providers.bytom.signature import Signature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(xprivate_key=
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> signature = Signature(network="testnet")
>>> signature.sign(bytom_fund_unsigned_raw, fund_solver)
>>> signature.unsigned_datas()
[{'datas': ['38601bf7ce08dab921916f2c723acca0451d8904649bbec16c2076f1455dd1a2
↪'], 'public_key':
↪'91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2', 'network
↪': 'mainnet', 'path': 'm/44/153/1/0/1'}]
```

6.5.1 FundSignature

class shuttle.providers.bytom.signature.**FundSignature**(*network='testnet'*)
Bytom FundSignature class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns FundSignature – Bytom fund signature instance.

sign(*unsigned_raw*, *solver*)
Sign unsigned fund transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bytom unsigned fund transaction raw.
- **solver** (bytom.solver.FundSolver) – Bytom fund solver.

Returns FundSignature – Bytom fund signature instance.

```
>>> from shuttle.providers.bytom.signature import FundSignature
>>> from shuttle.providers.bytom.solver import FundSolver
>>> bytom_fund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> fund_solver = FundSolver(
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪")
>>> fund_signature = FundSignature(network="testnet")
>>> fund_signature.sign(bytom_fund_unsigned_raw, fund_solver)
<shuttle.providers.bytom.signature.FundSignature object at 0x0409DAF0>
```

6.5.2 ClaimSignature

class shuttle.providers.bytom.signature.**ClaimSignature**(*network='testnet'*)
Bytom ClaimSignature class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns ClaimSignature – Bytom claim signature instance.

sign(*unsigned_raw*, *solver*)
Sign unsigned claim transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bytom unsigned claim transaction raw.
- **solver** (bytom.solver.ClaimSolver) – Bytom claim solver.

Returns ClaimSignature – Bytom claim signature instance.

```
>>> from shuttle.providers.bytom.signature import ClaimSignature
>>> from shuttle.providers.bytom.solver import ClaimSolver
>>> bytom_claim_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> recipient_xprivate_key =
↪"58dd4094155bbebf2868189231c47e4e0edbd9f74545f843c9537259e1d7a656983aef283d0ccebecc2d33577
↪"
>>> claim_solver = ClaimSolver(recipient_xprivate_key, "Hello Meheret!",
↪"3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820cabb",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
```

(continues on next page)

(continued from previous page)

```
>>> claim_signature = ClaimSignature(network="testnet")
>>> claim_signature.sign(bytom_claim_unsigned_raw, claim_solver)
<shuttle.providers.bytom.signature.ClaimSignature object at 0x0409DAF0>
```

6.5.3 RefundSignature

class shuttle.providers.bytom.signature.**RefundSignature**(*network='testnet'*)

Bytom RefundSignature class.

Parameters **network** (*str*) – Bytom network, defaults to testnet.

Returns RefundSignature – Bytom claim signature instance.

sign(*unsigned_raw*, *solver*)

Sign unsigned refund transaction raw.

Parameters

- **unsigned_raw** (*str*) – Bytom unsigned refund transaction raw.
- **solver** (bytom.solver.RefundSolver) – Bytom refund solver.

Returns RefundSignature – Bytom refund signature instance.

```
>>> from shuttle.providers.bytom.signature import RefundSignature
>>> from shuttle.providers.bytom.solver import RefundSolver
>>> bytom_refund_unsigned_raw =
↪"eyJmZWUiOiAxMDAwMDAwMCwgImd1aWQiOiAiZjBlZDZkZGQtOWQ2Yi00OWZkLTg4NjYtYTUyZDEwODNhMTNiIiwgI
↪"
>>> sender_xprivate_key =
↪"205b15f70e253399da90b127b074ea02904594be9d54678207872ec1ba31ee51ef4490504bd2b6f997113671
↪"
>>> refund_solver = RefundSolver(sender_xprivate_key,
↪"3a26da82ead15a80533a02696656b14b5dbfd84eb14790f2e1be5e9e45820eeb",
↪"3e0a377ae4afa031d4551599d9bb7d5b27f4736d77f78cac4d476f0ffba5ae3e",
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2", 1000)
>>> refund_signature = RefundSignature(network="testnet")
>>> refund_signature.sign(bytom_refund_unsigned_raw, refund_solver)
<shuttle.providers.bytom.signature.RefundSignature object at 0x0409DAF0>
```

6.6 Remote Procedure Call (RPC)

Bytom remote procedure call.

shuttle.providers.bytom.rpc.**account_create**(*xpublic_key*, *label='1st address'*, *email=None*, *network='testnet'*, *timeout=60*)

Create account in blockcenter.

Parameters

- **xpublic_key** (*str*) – Bytom xpublic key.
- **label** (*str*) – Bytom limit, defaults to 1st address.
- **email** (*str*) – email address, defaults to None.
- **network** (*str*) – Bytom network, defaults to testnet.

- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bytom blockcenter guid, address and label.

```
>>> from shuttle.providers.bytom.rpc import account_create
>>> account_create(xpublic_key, "mainnet")
{"guid": "f0ed6ddd-9d6b-49fd-8866-a52d1083a13b", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "label": "1st address"}
```

shuttle.providers.bytom.rpc.**build_transaction**(*tx*, *network='testnet'*, *timeout=60*)

Build Bytom transaction in blockcenter.

Parameters

- **tx** (*dict*) – Bytom transaction.
- **network** (*str*) – Bytom network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bytom built transaction.

```
>>> from shuttle.providers.bytom.rpc import build_transaction
>>> build_transaction(transaction, "mainnet")
{"tx": {"hash": "2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492
↪", "status_fail": false, "size": 379, "submission_timestamp": 0, "memo": "",
↪"inputs": [{"script": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address
↪": "bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪2450000000, "type": "spend"}], "outputs": [{"utxo_id":
↪"5edccebe497893c289121f9e365fdeb34c97008b9eb5a9960fe9541e7923aabc", "script":
↪"01642091ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e220ac13c0bb1445423a64175
↪", "address": "smart contract", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪1000, "type": "control"}, {"utxo_id":
↪"f8cfbb692db1963be88b09c314adcc9e19d91c6c019aa556fb7cb76ba8ffa1fa", "script":
↪"00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪2439999000, "type": "control"}], "fee": 10000000, "balances": [{"asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount": "-
↪10001000"}], "types": ["ordinary"]}, "raw_transaction":
↪"070100010160015e7f2d7ecec3f61d30d0b2968973a3ac8448f0599ea20dce883b48c903c4d6e87ffffffffffffff
↪", "signing_instructions": [{"derivation_path": ["2c000000", "99000000",
↪"01000000", "00000000", "01000000"], "sign_data": [
↪"37727d44af9801e9723eb325592f4d55cc8d7e3815b1d663d61b7f1af9fc13a7"], "pubkey":
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2"}], "fee":␣
↪10000000}
```

shuttle.providers.bytom.rpc.**decode_tx_raw**(*tx_raw*, *network='testnet'*, *timeout=60*)

Get decoded transaction raw.

Parameters

- **tx_raw** (*str*) – Bytom transaction raw.
- **network** (*str*) – Bytom network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bytom decoded transaction raw.

```
>>> from shuttle.providers.bytom.rpc import decode_tx_raw
>>> decode_tx_raw(transaction_raw, "testnet")
{...}
```

shuttle.providers.bytom.rpc.**get_balance**(*address*, *network='testnet'*, *limit=1*, *page=1*, *timeout=60*)

Get Bytom balance.

Parameters

- **address** (*str*) – Bytom address.
- **network** (*str*) – Bytom network, defaults to testnet.
- **limit** (*str*) – Bytom limit, defaults to 1.
- **page** (*str*) – Bytom network, defaults to 1.
- **timeout** (*int*) – request timeout, default to 15.

Returns int – Bytom balance.

```
>>> from shuttle.providers.bytom.rpc import get_balance
>>> get_balance(bytom_address, "mainnet")
25800000
```

shuttle.providers.bytom.rpc.**get_transaction**(*tx_id*, *network='testnet'*, *timeout=60*)

Get Bytom transaction detail.

Parameters

- **tx_id** (*str*) – Bytom transaction id.
- **network** (*str*) – Bytom network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bytom built transaction.

```
>>> from shuttle.providers.bytom.rpc import get_transaction
>>> get_transaction(transaction_id, "mainnet")
{"tx": {"hash": "2993414225f65390220730d0c1a356c14e91bca76db112d37366df93e364a492
↪", "status_fail": false, "size": 379, "submission_timestamp": 0, "memo": "",
↪"inputs": [{"script": "00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address
↪": "bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪2450000000, "type": "spend"}], "outputs": [{"utxo_id":
↪"5edccebe497893c289121f9e365fdeb34c97008b9eb5a9960fe9541e7923aabc", "script":
↪"01642091ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e220ac13c0bb1445423a64175
↪", "address": "smart contract", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪1000, "type": "control"}, {"utxo_id":
↪"f8cfbb692db1963be88b09c314adcc9e19d91c6c019aa556fb7cb76ba8ffa1fa", "script":
↪"00142cda4f99ea8112e6fa61cdd26157ed6dc408332a", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount":␣
↪2439999000, "type": "control"}], "fee": 10000000, "balances": [{"asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "amount": "-
↪10001000"}], "types": ["ordinary"]}, "raw_transaction":
↪"070100010160015e7f2d7ecec3f61d30d0b2968973a3ac8448f0599ea20dce883b48c903c4d6e87ffffffffffffff
↪", "signing_instructions": [{"derivation_path": ["2c000000", "99000000",
↪"01000000", "00000000", "01000000"], "sign_data": [
↪"37727d44af9801e9723eb325592f4d55cc8d7e3815b1d663d61b7f1af9fc13a7"(continues on next page)
↪"91ff7f525ff40874c4f47f0cab42e46e3bf53adad59adef9558ad1b6448f22e2"}], "fee":␣
↪10000000}
```

(continued from previous page)

shuttle.providers.bytom.rpc.**list_address**(*guid*, *limit=10*, *network='testnet'*, *timeout=60*)

List address from blockcenter.

Parameters

- **guid** (*str*) – Bytom blockcenter guid.
- **limit** (*int*) – blockcenter limit default to 10.
- **network** (*str*) – Bytom network, defaults to testnet.
- **timeout** (*int*) – request timeout, default to 15.

Returns list – Bytom blockcenter list of addresses.

```
>>> from shuttle.providers.bytom.rpc import list_address
>>> list_address(guid, 5 "mainnet")
[{"guid": "f0ed6ddd-9d6b-49fd-8866-a52d1083a13b", "address":
↪"bm1q9ndylx02syfwd7npehfxz4lddhzqsve2fu6vc7", "label": "1st address", "balances
↪": [{"asset": "f37dea62efd2965174b84bbb59a0bd0a671cf5fb2857303ffd77c1b482b84bdf
↪", "balance": "100000000000", "total_received": "100000000000", "total_sent": "0
↪", "decimals": 8, "alias": "Asset", "icon": "", "name":
↪"f37dea62efd2965174b84bbb59a0bd0a671cf5fb2857303ffd77c1b482b84bdf", "symbol":
↪"Asset", "in_usd": "0.00", "in_cny": "0.00", "in_btc": "0.000000"}, {"asset":
↪"ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "balance":
↪"2450000000", "total_received": "4950000000", "total_sent": "2500000000",
↪"decimals": 8, "alias": "btm", "icon": "", "name": "BTM", "symbol": "BTM", "in_
↪usd": "2.90", "in_cny": "20.58", "in_btc": "0.000283"}]}]
```

shuttle.providers.bytom.rpc.**submit_payment**(*guid*, *tx_raw*, *signatures*, *network*, *memo='mock'*, *timeout=60*)

Submit transaction raw to Bytom blockchain.

Parameters

- **guid** (*str*) – Bytom blockcenter id.
- **tx_raw** (*str*) – Bytom transaction raw.
- **signatures** (*list*) – Bytom signed datas.
- **network** (*str*) – Bytom network, defaults to testnet.
- **memo** (*str*) – memo, defaults to mock.
- **timeout** (*int*) – request timeout, default to 15.

Returns dict – Bytom transaction id, fee, type and date.

```
>>> from shuttle.providers.bytom.rpc import submit_payment
>>> submit_payment("guid", transaction_raw, [[...]], "mainent")
{...}
```

6.7 Utils

Bytom Utils.

shuttle.providers.bytom.utils.**contract_arguments**(*amount*, *address*)
Get contract arguments.

Parameters

- **amount** (*int*) – Bytom amount.
- **address** (*str*) – Bytom address.

Returns list – Bytom contract arguments.

```
>>> from shuttle.providers.bytom.utils import contract_arguments
>>> contract_arguments(bytom_amount, bytom_address)
[...]
```

shuttle.providers.bytom.utils.**control_address_action**(*amount*, *asset*, *address*)
Get control address action.

Parameters

- **amount** (*int*) – Bytom amount.
- **asset** (*str*) – Bytom asset.
- **address** (*str*) – Bytom address.

Returns dict – Bytom control address action.

```
>>> from shuttle.providers.bytom.utils import control_address_action
>>> control_address_action(bytom_amount, bytom_asset, bytom_address)
{...}
```

shuttle.providers.bytom.utils.**control_program_action**(*amount*, *asset*, *control_program*)
Get control program action.

Parameters

- **amount** (*int*) – Bytom amount.
- **asset** (*str*) – Bytom asset.
- **control_program** (*str*) – Bytom control program.

Returns dict – Bytom control program action.

```
>>> from shuttle.providers.bytom.utils import control_program_action
>>> control_program_action(bytom_amount, bytom_asset, bytom_control_program)
{...}
```

shuttle.providers.bytom.utils.**decode_transaction_raw**(*transaction_raw*)
Decode Bytom transaction raw.

Parameters **transaction_raw** (*str*) – Bytom transaction raw.

Returns dict – decoded Bytom transaction.

```
>>> from shuttle.providers.bytom.utils import decode_transaction_raw
>>> decode_transaction_raw(transaction_raw)
{...}
```

shuttle.providers.bytom.utils.**find_contract_utxo_id**(*tx_id*, *network*)

Find smart contract UTXO id.

Parameters

- **tx_id** (*str*) – Bytom transaction id or hash.
- **network** (*str*) – Bytom network.

Returns str – UTXO id.

```
>>> from shuttle.providers.bytom.utils import find_contract_utxo_id
>>> find_contract_utxo_id(bytom_transaction_id, "mainnet")
"9059cd0d03e4d4fab70a415169a45be47583f7240115c36cf298d6f261c0a1ac"
```

shuttle.providers.bytom.utils.**spend_account_action**(*account*, *amount*, *asset*)

Get spend account action.

Parameters

- **account** (*str*) – Bytom account.
- **amount** (*int*) – Bytom amount.
- **asset** (*str*) – Bytom asset.

Returns dict – Bytom spend account action.

```
>>> from shuttle.providers.bytom.utils import spend_account_action
>>> spend_account_action(bytom_account, bytom_amount, bytom_asset)
{...}
```

shuttle.providers.bytom.utils.**spend_utxo_action**(*utxo*)

Get spend UTXO action

Parameters **utxo** (*str*) – Bytom butxo id.

Returns dict – Bytom spend utxo action.

```
>>> from shuttle.providers.bytom.utils import spend_utxo_action
>>> spend_utxo_action(bytom_utxo_id)
{...}
```

shuttle.providers.bytom.utils.**spend_wallet_action**(*amount*, *asset*)

Get spend wallet action.

Parameters

- **amount** (*int*) – Bytom amount.
- **asset** (*str*) – Bytom asset.

Returns dict – Bytom spend wallet action.

```
>>> from shuttle.providers.bytom.utils import spend_wallet_action
>>> spend_wallet_action(bytom_amount, bytom_asset)
{...}
```

shuttle.providers.bytom.utils.**submit_transaction_raw**(*transaction_raw*)

Submit transaction raw to Bytom blockchain.

Parameters **transaction_raw** (*str*) – Bytom transaction raw.

Returns dict – Bytom transaction id, fee, type and date.

```
>>> from shuttle.providers.bytom.utils import submit_transaction_raw
>>> submit_transaction_raw(transaction_raw)
{...}
```

PYTHON MODULE INDEX

S

INDEX

Symbols

A

B

C

D

E

F

G

H

I

J

L

M

O

P

R

S

T

U

W

X