Mastering Bitcoin Ch.6

Transactions

Kun-jung Wu

Department of Economics
National Taiwan University

R06323008@ntu.edu.tw

March 11, 2018

Overview

1. 1. Transaction behind the Scene
2. 2. Transaction Input and Output
3. 3. Transaction Scripts and Scripting Language
4. 4. Digital Signatures (ECDSA)

1. Decode the Transaction

The transaction is stored in a JSON structure.

vin

```
{
  "version": 1,
  "locktime": 0,
  "vin": [
    {
      "txid": "7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65
      "vout": 0,
      "scriptSig" : "3045022100884d142d86652a3f47ba4746ec719bbfbd040
      "sequence": 4294967295
    }
  ],
```

1. Decode the Transaction

vout

```
  "vout": [
    {
      "value": 0.01500000,
      "scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e05
    },
    {
      "value": 0.08450000,
      "scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa
    }
  ]
}
```

2. Transaction Input and Output

UTXO

- Unspent transaction outputs
- The balance of the wallet is the sum of all UTXO the wallet's private key can control.
- Integer number in satoshis
- Must be consumed entirely. Cannot be cut in half.
- Coinbase is created when the block is created (Only outpus)

2. Transaction Input and Output

Transaction Outputs

- Every transaction creates outputs (two parts)
- An amount of bitcoin in satoshis
- locking script (or witness script, scriptPubKey) is a cryptographic puzzle that determines the conditions required to spend the output
- transaction is serialized in Bitcoin Network

2. Transaction Input and Output

Transaction Inputs

- txid is the reference to the UTXO
- vout is the output index
- scriptSig which satisfies the conditions placed on the UTXO, unlocking it for spending
- sequence number

Once the transaction is broadcast to the network, every node will retrieve the UTXO and validate the transaction.

Transaction Fees

- Small cost can deter abuse of the system.
- Miners serve the transaction with the highest fee first. (Market forces)
- Based the size of transaction, not the value of bitcoin.
- Fee is calculated by the difference between outputs and inputs.
- `https://bitcoinfees.earn.com/`

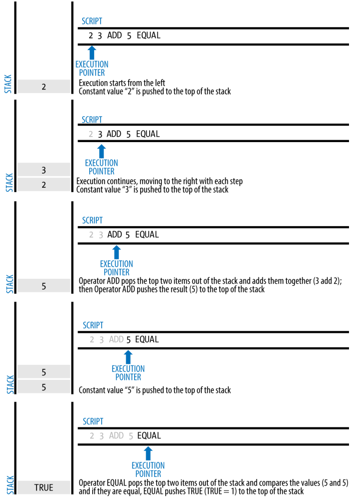
3. Transaction Scripts and Scripting Language

Transaction Script

- Forth-like reverse-polish notation stack-based
- Turning Incomplete: no loops or complex flow control
- Ensure finite complexity
- Stateless Verification: all the information needed to execute a script is contained within the script. (ensures the same result for every machine)

3. Transaction Scripts and Scripting Language

Stack Execution



Execution starts from the left
Constant value "2" is pushed to the top of the stack

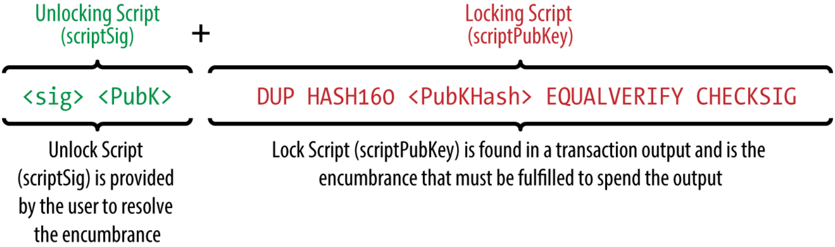
Execution continues, moving to the right with each step
Constant value "3" is pushed to the top of the stack

Operator ADD pops the top two items out of the stack and adds them together (3 add 2);
then Operator ADD pushes the result (5) to the top of the stack

Constant value "5" is pushed to the top of the stack

Operator EQUAL pops the top two items out of the stack and compares the values (5 and 5)
and if they are equal, EQUAL pushes TRUE (TRUE = 1) to the top of the stack

3. Transaction Scripts and Scripting Language



Unlocking Script (scriptSig)

+

Locking Script (scriptPubKey)

`<sig> <PubK>`

`DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG`

Unlock Script (scriptSig) is provided by the user to resolve the encumbrance

Lock Script (scriptPubKey) is found in a transaction output and is the encumbrance that must be fulfilled to spend the output

3. Transaction Scripts and Scripting Language

Script Construction

- Locking Script: a spending condition placed on an output: it specifies the conditions that must be met to spend the output in the future
- Unlocking Script: a script that "solves," or satisfies, the conditions placed on an output by a locking script and allows the output to be spent.

3. Transaction Scripts and Scripting Language

Pay-to-Public-Key-Hash (P2PKH)

- Most common
- Locking Script: a spending condition placed on an output: it specifies the conditions that must be met to spend the output in the future
- Unlocking Script: a script that "solves," or satisfies, the conditions placed on an output by a locking script and allows the output to be spent.

3. Transaction Scripts and Scripting Language

- These outputs contain a locking script that locks the output to a public key hash, more commonly known as a bitcoin address. An output locked by a P2PKH script can be unlocked (spent) by presenting a public key and a digital signature created by the corresponding private key
- The Cafe Public Key Hash is equivalent to the bitcoin address of the cafe, without the Base58Check encoding.

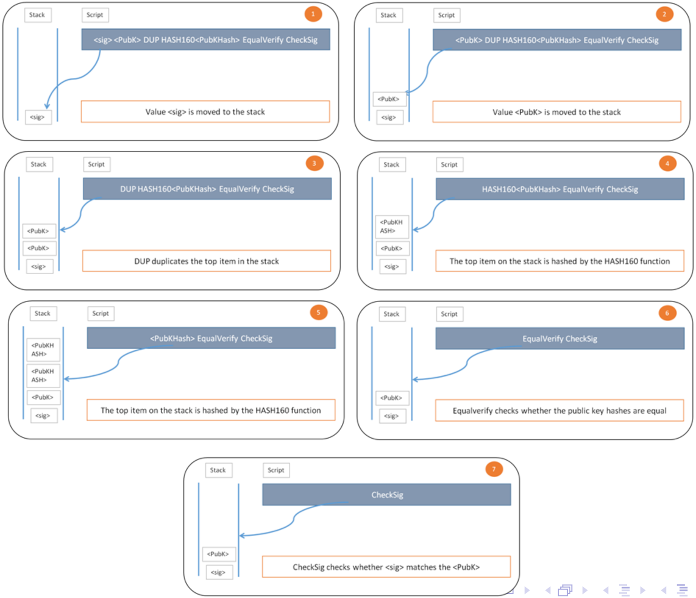
locking script

```
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG
```

unlocking script

```
<Cafe Signature> <Cafe Public Key>
```

3. Transaction Scripts and Scripting Language



Value <sig> is moved to the stack

Value <PubK> is moved to the stack

DUP duplicates the top item in the stack

The top item on the stack is hashed by the HASH160 function

The top item on the stack is hashed by the HASH160 function

Equalverify checks whether the public key hashes are equal

CheckSig checks whether <sig> matches the <PubK>

4. Digital Signatures(ECDSA)

Signature Hash Types(SIGHASH)

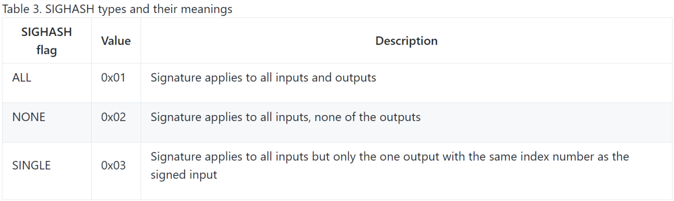


Table 3. SIGHASH types and their meanings

SIGHASH flag	Value	Description
ALL	0x01	Signature applies to all inputs and outputs
NONE	0x02	Signature applies to all inputs, none of the outputs
SINGLE	0x03	Signature applies to all inputs but only the one output with the same index number as the signed input

4. Digital Signatures(ECDSA)

Signature Hash Types(SIGHASH)

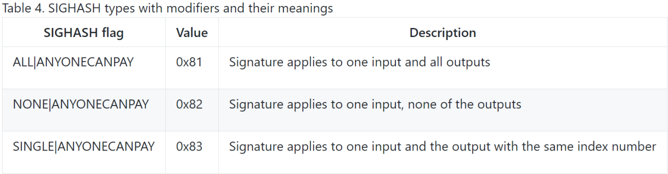


Table 4. SIGHASH types with modifiers and their meanings

SIGHASH flag	Value	Description
ALL|ANYONECANPAY	0x81	Signature applies to one input and all outputs
NONE|ANYONECANPAY	0x82	Signature applies to one input, none of the outputs
SINGLE|ANYONECANPAY	0x83	Signature applies to one input and the output with the same index number

The End