Blockchain Transactions

R. R. Brooks – rrb@acm.org
Clemson University, Electrical and Computer Engineering

July, 2019

These slides

- For programming assignment;
- Gives information on BTC transactions and what we want in program;
- The two are similar but not the same;
- Also explains how to use Openssl for DSA signing;
- We give command line calls;
- We give reference to Google C calls;

Blockchain – Transaction

- For now, only simplest type of transaction [1]
- Pay-to-PubkeyHash
- In general, BTC transactions processed using a variant of Forth.
- Forth is a stack based language developed in 1968.
- Forth was popular with embedded systems in the 1980's ;)
- We will do a transaction with one input and two outputs

[1] https://en.bitcoin.it/wiki/Transaction

Transaction – Format

- Transaction format [2]
- Version – 4 bytes – Data structure version – ex. 01000000
- Input count – variable – Upcoming number of inputs – ex. 01
  - Inputs – next slide – repeated input count number of times
- Output count – variable – Upcoming number of outputs – ex. 02
  - Outputs – future slide – repeated output count number of times
- Locktime – 4 bytes – How long to wait until processing - ex. 00000000

[2] https://learnmeabitcoin.com/guide/transaction-data

Transaction – Input Part 1

- Input: (Has 5 elements)
  1. TXID: – 32 bytes – Hex – Hash of previous transaction "f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6"
     - to calculate
     - Take transaction whose output you want to spend
     - Do SHA256 twice
     - Reverse the byte order
     - Used to find transactions in BTC blockchain
     - *bitcoin-cli getrawtransaction 0e3e2357e806b6cdb1f70b54c3a3a17b6714ee1f0e68bebb44a74b1efd512098*
  2. VOUT – 4 bytes – output number in list – ex: 0

Transaction – Input Part 2

- Input Continued: (fourth element)

4. SciptSig Size – variable – Size of the Script's
5. ScriptSig:[3]
   - "304502206e21798a42fae0e854281abd38bacd1aeed3ee373 8d9e1446618c4571d10 90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e 7ba5fdd7d5d6cc8d25c6b241501"
   - BTC Script
     $< ScriptPubKey = OP\_DUP\ OP\_HASH160 < PublicKeyHash >$
     $OP\_EQUAL\ OP\_CHECKSIG$
     $ScriptSig =< Signature >< PublicKey >$
   - Pushes hash onto stack; pushes signature onto stack;
   - Checks that signature wsa signed with publickey;
   - Checks that hash in signature equals hash on stack;
   - We will not force you to write a Forth scripting interpreter;
   - ScriptSig will be signed public key and public key;
   - Hash of public key from ScriptSig has to match public key hash in output VOUT of TXID indexed transaction;
   - How to EC sign – later;
6. Sequence no – 4 bytes – usually "FFFFFFFF"; irrelevant unless lock time is > 0;

[3] https://www.cryptocompare.com/wallets/guides/bitcoin-transaction

Transaction – Output

- Output: (Has 3 elements)
  1. Value – 8 bytes – Amount of output in Satoshi's 10^8 Satoshi $==$ 1 BTC;[4]
  2. ScriptPubKey Size – Variable length – Size of locking code;
  3. ScriptPubkey: We only do "P2PKH" - Pay to public key hash
     - OP_DUP OP_HASH160 "12ab8dc588ca9d5787dde7eb29569da63c3a238c" OP_EQUALVERIFY OP_CHECKSIG
     - For program give the address used by receipient;
     - Address is byte reversed SHA-256(ShH-256(Public Key))

[4] https://learnmeabitcoin.com/guide/transaction-data

EC Signature

- From command line:[5]
  - Sign file:
    - openssl dgst -ecdsa-with-SHA1 -sign private.pem test.pdf > signature.bin
  - Verify file:
    - openssl dgst -ecdsa-with-SHA1 -verify public.pem -signature signature.bin test.pdf
- C source code[6]
- C code from Google for reference
- I suggest using system() or exec calls with command line.

[5] https://superuser.com/questions/737574/openssl-ecdsa-sign-and-ve
[6] https://chromium.googlesource.com/chromiumos/third_party/openssl

EC Signature

- ☐ Explaind BTC transactions;
- ☐ Explained how transactions should work in program 1;
- ☐ You will need DSA signatures;
- ☐ You will need to verify DSA signatures;
- ☐ Explained how to do DSA signature and verification at command line;
- ☐ Gave reference code;