NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Bitcoin

Optimizations in Bitcoin Mechanics

Dr Sourav SEN GUPTA
Lecturer, SCSE, NTU

Bitcoin Optimizations

Transaction Verification

Recall : Bitcoin Transaction

Inputs

HEX | ASM

Index	0	Details	Output
Address	11LjtJgf8AS7misuBjHPJLCE8p6bjG7GT	Value	1.60493224 BTC
Pkscript	76a91400107a44cb1c3957cc23a87ffc5a1de458d1827788ac		
Sigscript	47304402201743b29c473f2fc3a0f36edd87bd479a46f0f4c94e7c29fa3bb1704f8b75cdd1022065f3c3e655671a82dc70f311da7535c402f76bc1e91e2be452ee6dcf5511fbfc012102f3a3fdba423cf1b094ebcf5f87158ffc427ea0c24b4a68d82b7c5b7deaae49fe		
Witness	N/A		

Outputs

Index	0	Details	Unspent
Address	1Nr4GgADbyxaD2MvRGaoEEpxCgLWjhuAjM	Value	0.75760000 BTC
Pkscript	76a914efa1c0e2fe35adbb55dba6ff744eab73e46e249688ac		

Index	1	Details	Unspent
Address	1P8TWBf3RyojFBXEnFLnJ2vBFFEs9nmre5	Value	0.84684905 BTC
Pkscript	76a914f2bbcd7e20fd34cfc8fe0840eee704181138b78488ac		

Single Input UTXO
Pkscript Locks
Sigscript Unlocks

Two Output UTXOs
Pkscript Locks

ref : https://www.blockchain.com/btc/tx/8a39aa4c73cb4b87904db0f0b8c27f24b0b8b9d17bd8087b7dba7342ab8a0be8

Bitcoin Script

Single Input UTXO
Pkscript Locks
Sigscript Unlocks

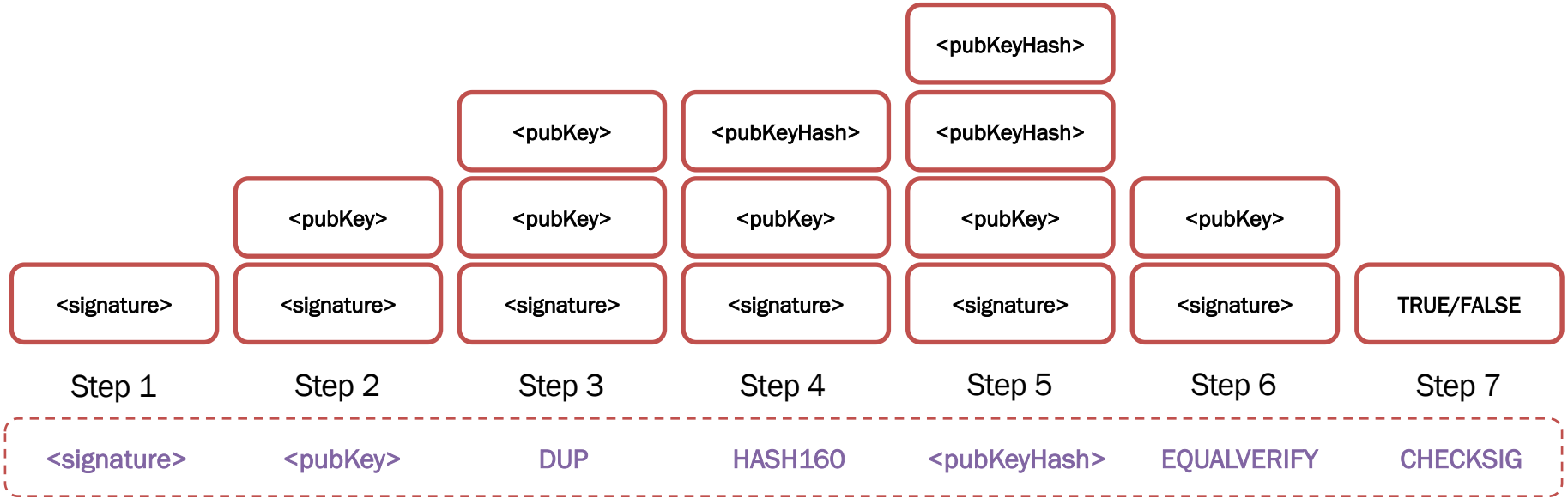
scriptPubKey	DUP HASH160 <pubKeyHash> EQUALVERIFY CHECKSIG	Locking
scriptSig	<signature> <pubKey>	Unlocking

Bitcoin *Script* is a lightweight stack-based execution language.

ref : https://www.blockchain.com/btc/tx/8a39aa4c73cb4b87904db0f0b8c27f24b0b8b9d17bd8087b7dba7342ab8a0be8

Executing Bitcoin Script

scriptSig <signature> <pubKey> Unlocking
scriptPubKey DUP HASH160 <pubKeyHash> EQUALVERIFY CHECKSIG Locking



Properties of Script [1]

Types of **opcode** : Constants, Flow Control, Stack, Splice, Logic, Arithmetic, Crypto, Locktime, Pseudo-Words, Reserved Words (some disabled for various reasons)

Turing Incomplete

- **Limited complexity** with no support for complex control flow logic
- **Predictable execution time** without the fear of infinite loops (DoS)

Stateless Verification

- Execution **does not need any prior state** (or save any afterwards)
- Predictably **execute the same way** on any system in the network

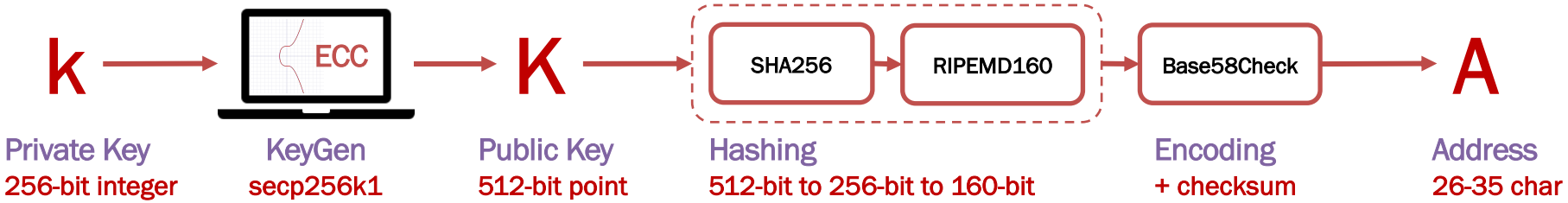
[1] reading: Chapter 5 of the book “Mastering Bitcoin”

Bitcoin Optimizations

Addresses, Keys and Wallets

Recall : Bitcoin Addresses

Key Generation : Elliptic Curve Cryptography (curve secp256k1)
Address Generation : SHA256 and RIPEMD160 (hash functions)



Private Key Signs, Public Key Verifies, and Bitcoin Address denotes Identity.

Types of Bitcoin Address

Depends on the types of Bitcoin Script used in Transactions

- Pay-to-Public-Key : Recipient's <pubKey> used in locking script (deprecated)
- **Pay-to-Public-Key-Hash (P2PKH)** : Standard format with <pubKeyHash>
- **Multi-Signature** (limited to 15 keys) : Multiple addresses in the scripts
- **Pay-to-Script-Hash (P2SH)** : Script Hash as "address" in the locking script
- Data Output (OP_RETURN) : Un-spendable output for non-payment <data>

P2PKH	1BvBMSEYstWetqTFn5Au4m4GFg7xJaNVN2	Standard Bitcoin address since 2009
P2SH	3J98t1WpEZ73CNmQviecrnyiWrnqRhWNLy	Pay-to-Script address since 2012
Bech32	bc1qar0srrr7xfkvy5l643lydnw9re59gtzzwf5mdq	SegWit addresses since 2017

Compressed Keys [2]

Private Key k in Bitcoin is a 256-bit Integer chosen uniformly at random.

Public Key $K = kG$ in Bitcoin is a 512-bit Point on Elliptic Curve secp256k1.

Uncompressed Key $K = (x, y)$	where x = F028892BAD…DC341A, y = 07CF33DA18…505BDB
04F028892BAD … 505BDB	04 denotes uncompressed key of size (8 + 512) = 520 bits
Compressed Key $K = (x, +/-)$	where $y^2 = x^3 + 7 \pmod p$
02F028892BAD … DC341A	02 denotes y is even modulo p for (8 + 256) = 264-bit key
03F028892BAD … DC341A	03 denotes y is odd modulo p for (8 + 256) = 264-bit key

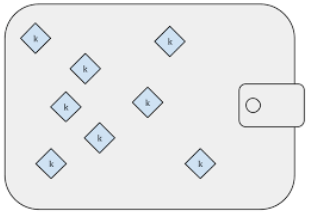
[2] reading: Chapter 4 of the book “Mastering Bitcoin”

Bitcoin Wallets [2]

Wallet is a "container" for Bitcoin Private Keys (file or database or generator).

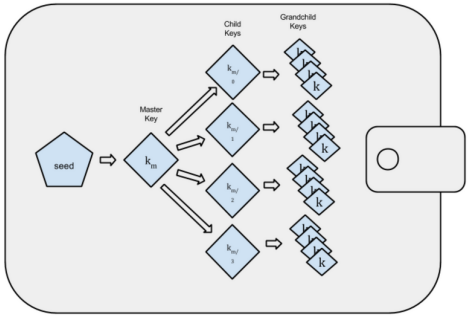
Non-Deterministic (Random) Wallets

- Stores or generates Random Keys when needed
- Uses one key only once as a Transaction Identity



Deterministic (Seeded) Wallets

- Generates Keys (as needed) from common Seed
- The seed is sufficient for Wallet export or backup
- Hierarchical Wallets generate Tree of Private Keys



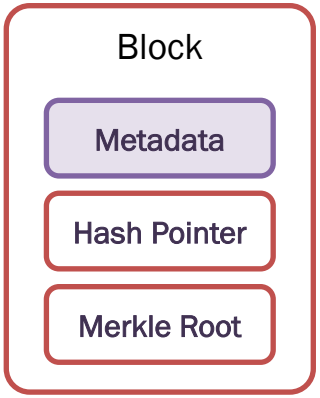
[2] reading: Chapter 4 of the book "Mastering Bitcoin"

Bitcoin Optimizations

Transactions and Blocks

Transaction Aggregation

Mining Nodes create Blocks out of their Transaction Pool according to certain order based on “transaction priority”.



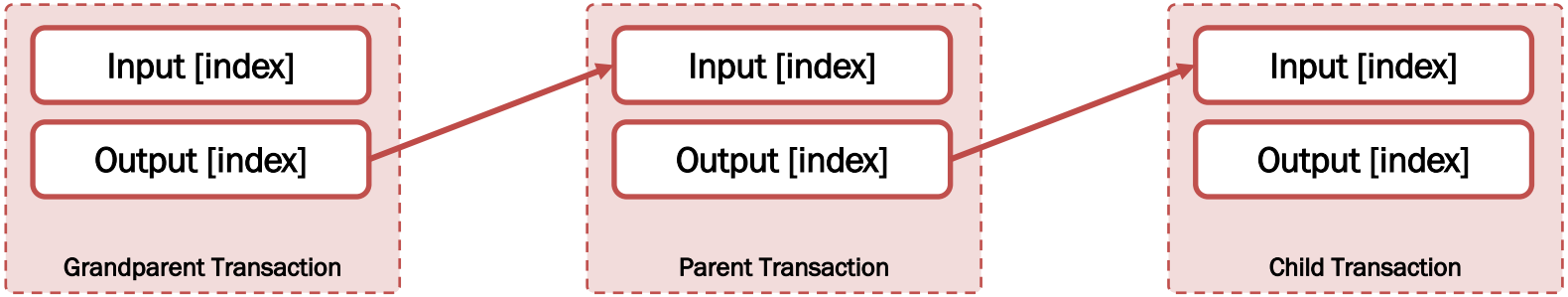
Priority = Sum (Value of input * Input Age) / Tx Size

High priority transactions have reserved space in blocks.
Beyond that, a node may prioritize high fee transactions.

2015 Dual-policy priority-based selection of transactions deprecated to simplify the creation of Block using transactions in the MemPool.

Orphan Transactions

Chain of Transactions : Output of one transaction used as an Input to another.



In case Child Transaction is “seen” before the Parent or Grandparent, it is stored separately in an “Orphan Transaction” pool, to be considered later.

To prevent DoS Attack, there is fixed limit MAX_ORPHAN_TRANSACTIONS.

Bitcoin Genesis

Genesis Block : First block in Bitcoin blockchain, created on 3 January 2009.

Hash = 000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

Statically encoded within Bitcoin core client, thus allowing every Bitcoin Node to start the Bitcoin blockchain with at least one "root" block as ground truth.

Coinbase parameter of the Genesis Block contains "timestamp" from News.

The Times 03/Jan/2009 Chancellor on brink of second bailout for banks.

Think about it : What happens if there is no agreement on Genesis Block?

Bitcoin Blocksize

There has been quite a lot of debate and controversy on this topic to date.

- Originally limited by the number of database locks (effectively **750 KB**).
- 2013 : Updated **limit 1 MB** using a “hard fork” in the Bitcoin blockchain.
- 2015 : Proposals initiated to use larger blocksize limits (**2 MB to 32 MB**).

Segregated Witness (SegWit)

- Signature data and “witness” segregated from transactions in Merkle Tree.
- Transactions still contain Sender-Receiver information as input and output.
- Transactions counted normally but Witness counted as $1/4^{th}$ of actual size.
- Activated in August 2017, effectively allowing the blocksize limit to be larger.

Bitcoin Optimizations

Mining and Consensus

Proof-of-Work Nonce

Block Header can only accommodate a 32-bit Nonce, as per the design.

- With increasing difficulty (> 60 bits to date), the Nonce is not "sufficient".
- Randomness boost with (Nonce, Timestamp) combination is inefficient.
- Nonce space 2^{32} is exhausted within a second by GH to TH/sec mining.

Extra Nonce in Coinbase Transaction

- Extra 64-bit Nonce space is allocated within the data portion of Coinbase.
- This increases the Nonce space to (32 + 64) = 96 bits, that is 2^{96} options.
- Each choice of the Coinbase Nonce requires recomputing the Merkle Root.
- Thus, for each Coinbase Nonce, miners exhaust the Header Nonce search.

Mining Pools

Group of Miners working together to maximize Hash Ratio by accumulation.

- Every miner in the pool mines a block with coinbase address of "manager".
- Miners separate ranges of nonce (if possible) to accumulate hash powers.
- If mining is "successful" for anyone in the pool, the reward is shared to all.
- Mining Share is determined based on work, proved by "nearly valid blocks".

Miner 1
000AF769 ...
00018725 ...
0003AC87 ...

Miner 2
00078BC6 ...

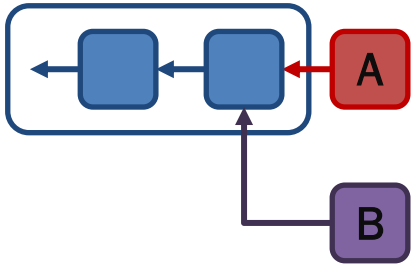
Miner 3
00004BE0 ...
000E10F7 ...

Miner 1 gets the largest share
Even though Miner 3 wins PoW

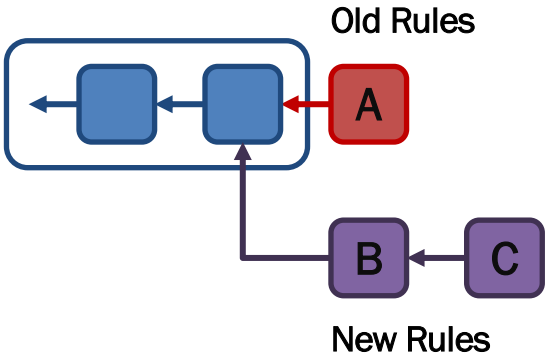
Based on *provable* PoW efforts.

Soft and Hard Forks

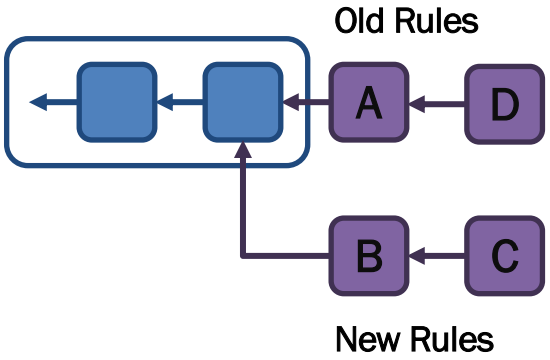
Forks in Bitcoin may arise due to inconsistency, concurrency or "rule change".



Inconsistency / Concurrency
Nodes vote to sustain a chain



Soft Fork with Old and New Rules
Blocks violating New Rules made stale



Hard Fork with Old and New Rules
Non-upgraded nodes reject New Rules