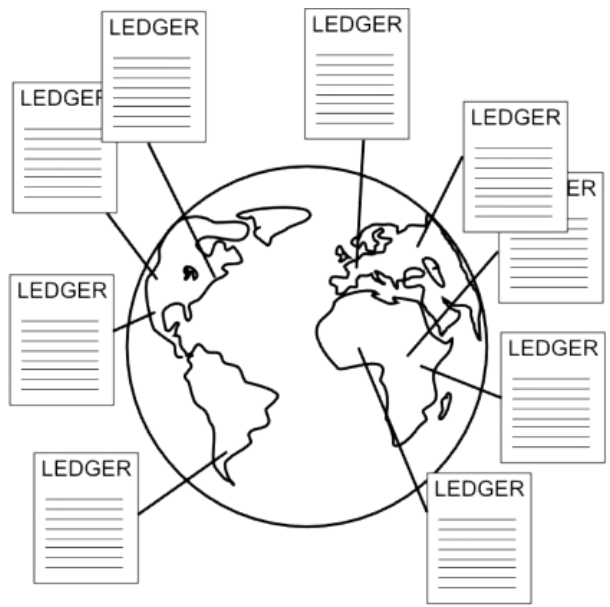
Security Engineering

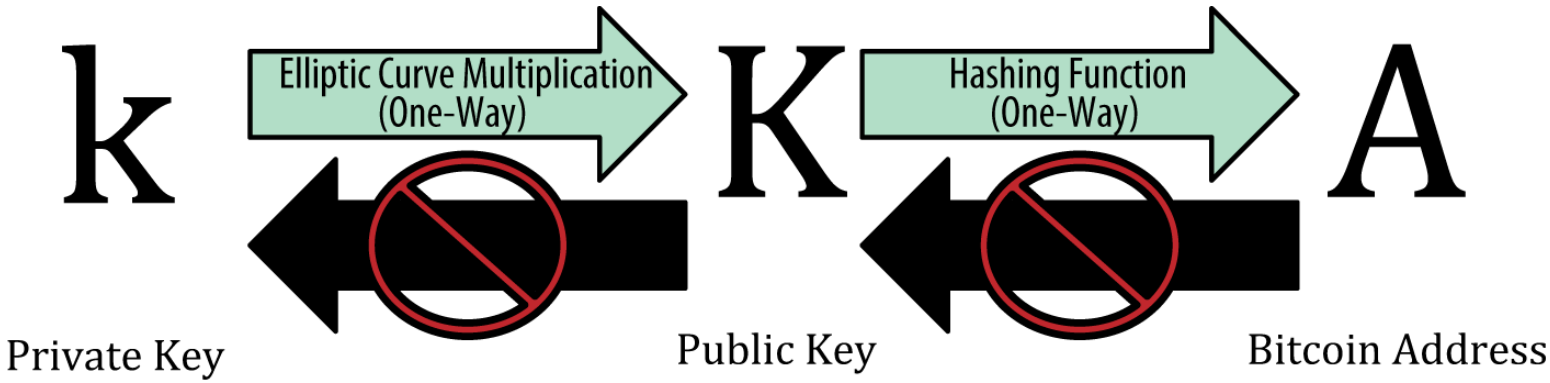
Email: christian.urban at kcl.ac.uk
Office: S1.27 (1st floor Strand Building)
Slides: KEATS (also homework is there)

Recall: Bitcoins

- a crypto currency by Satoshi Nakamoto
- mined by solving special puzzles involving hashes
- transaction history (ledger/blockchain) is P2P distributed (12 GB)
- surely a scam/ponzi scheme!



Bitcoin Keys



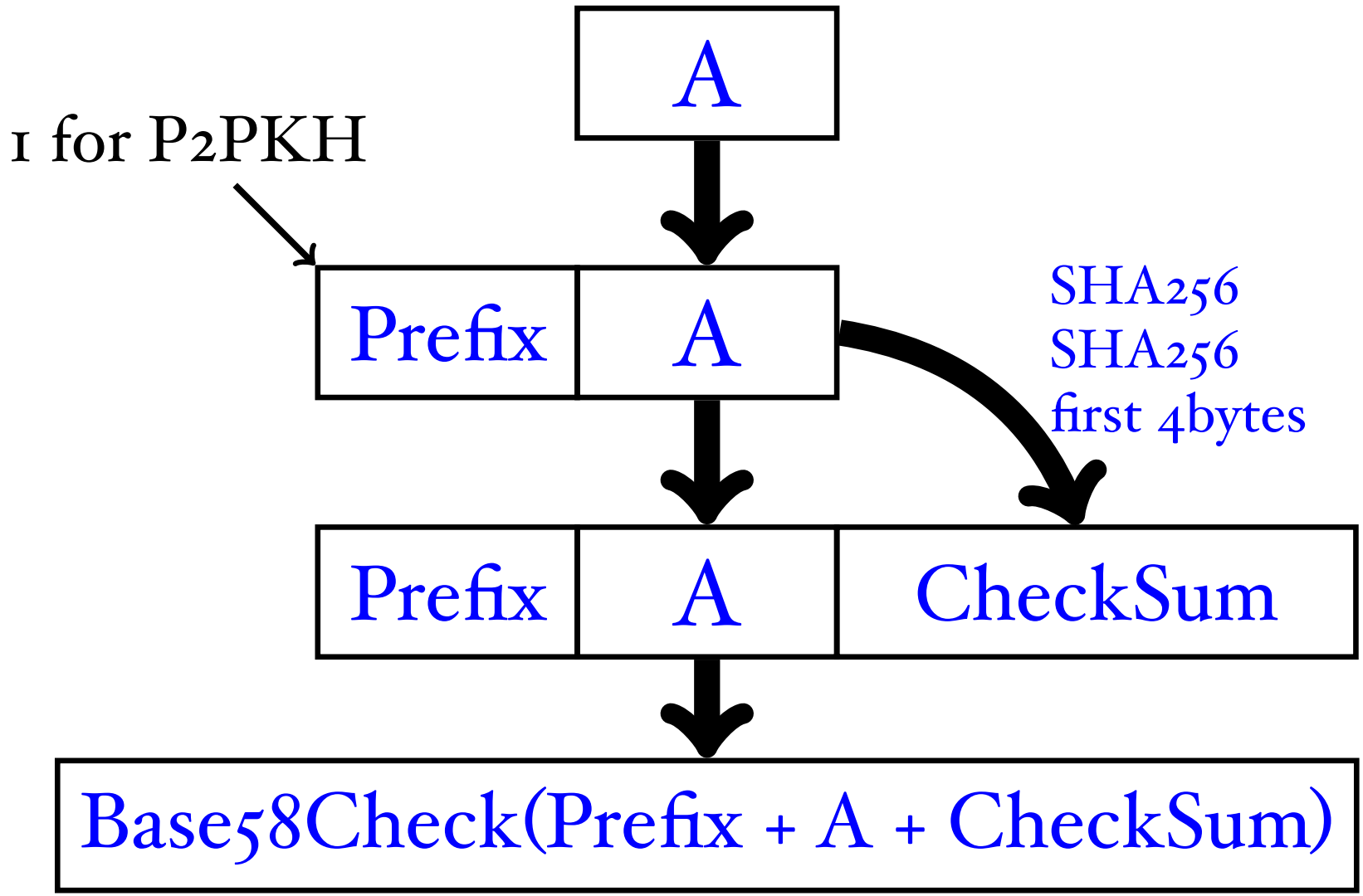
- k private key: 256 bits (randomly chosen)
- K public key: generated from k
- A bitcoin address: 160 Bit/20 Byte number:

$$A \stackrel{\text{def}}{=} RIPEMD160(SHA256(K))$$

RIPEMD160, SHA256 are hash functions

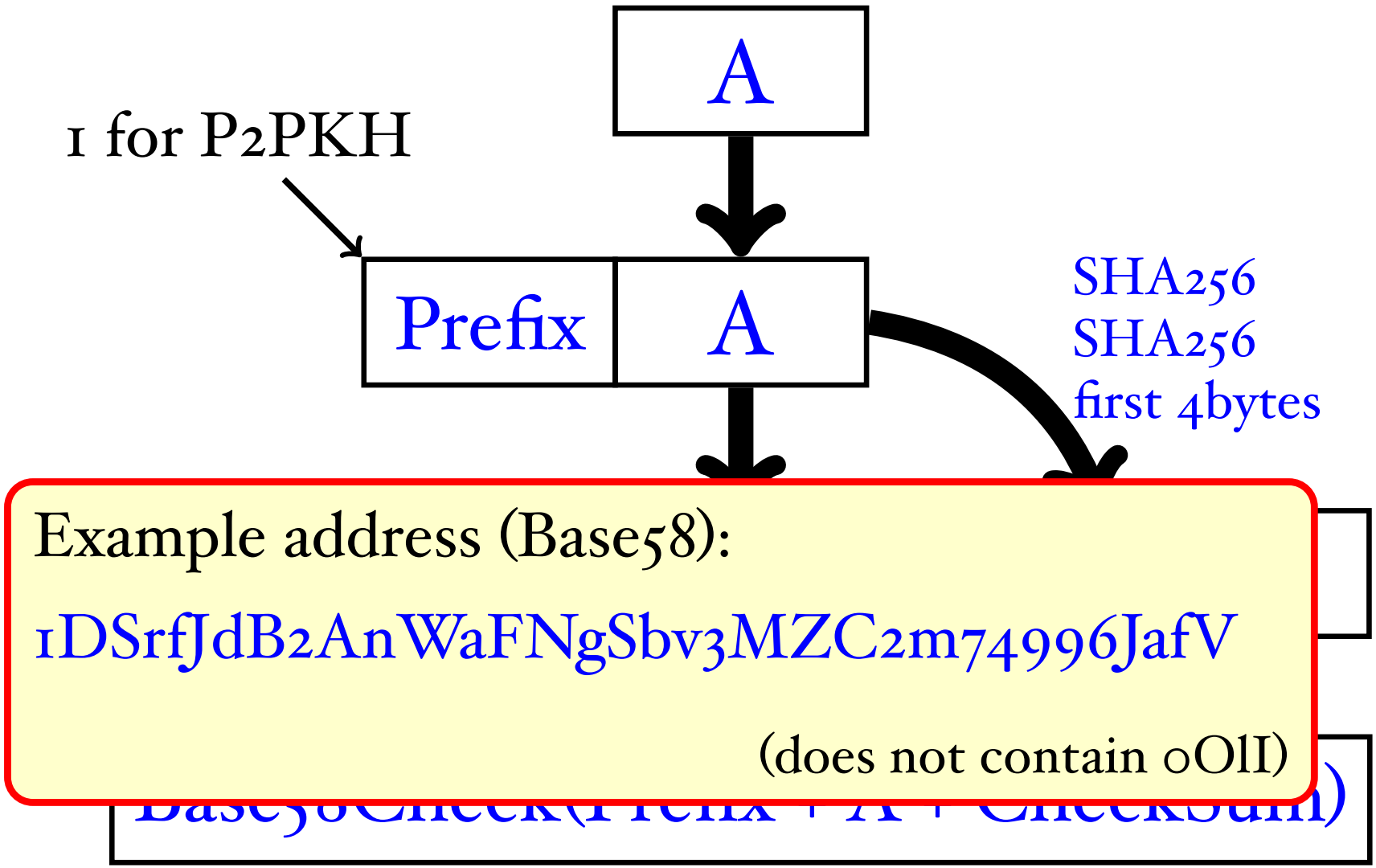
Bitcoin Addresses

The "human readable, checked version" of A:



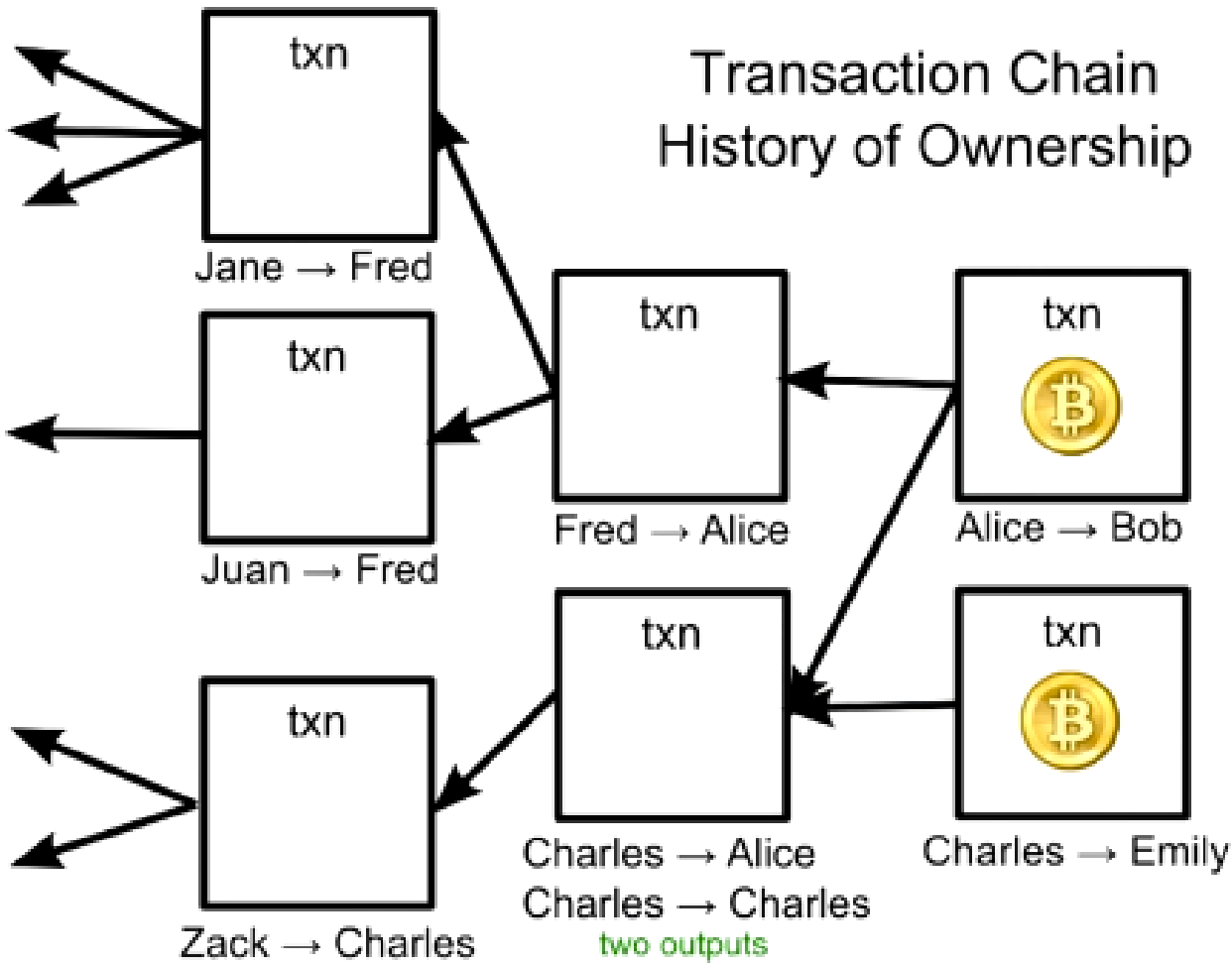
Bitcoin Addresses

The "human readable, checked version" of A:



Transaction Graph

older current



Types of Transactions

- pay-to-public-key-hash (so far: Alice pays Bob)

Types of Transactions

- pay-to-public-key-hash (so far: Alice pays Bob)
- pay-to-script-hash

$$RIPEMD160(SHA256(script))$$

Types of Transactions

- pay-to-public-key-hash (so far: Alice pays Bob)
- pay-to-script-hash

$$RIPEMD160(SHA256(script))$$

- Each transaction, including P2PKH, contains a locking and an unlocking script (locking from output; unlocking from input).
- The scripts are written in a Forth-like language (stack based).
- Running both scripts has to evaluate to True.

Pay-to-Public-Key-Hash

- Alice pays Bob:

```
<Bob's signature>          (unlocking script from input)
<Bob's PKey>

OP_DUP                     (locking script from output)
OP_HASH160
<Bob's PKey Hash>
OP_EQUALVERIFY
OP_CHECKSIG
```

A Transaction Msg


```
{”hash”:”7c4025...”,
 ”ver”:1,
 ”vin_sz”:1,
 ”vout_sz”:1,
 ”lock_time”:0,
 ”size”:224,
 ”in”:[
    {”prev_out”:
      {”hash”:”2007ae...”,
       ”n”:0},
       ”scriptSig”:”304502... 042b2d...”}],
 ”out”:[
   {”value”:”0.31900000”,
    ”scriptPubKey”:”OP_DUP OP_HASH160 a7db6f...
                   OP_EQUALVERIFY OP_CHECKSIG”}]}
```

A Transaction Msg


```
{”hash”:”7c402
 ”ver”:1,
 ”vin_sz”:1,
 ”vout_sz”:1,
 ”lock_time”:0,
 ”size”:224,
 ”in”:[
    {”prev_out”:
      {”hash”:”2007ae...”,
       ”n”:0},
       ”scriptSig”:”304502... 042b2d...”}],
 ”out”:[
   {”value”:”0.31900000”,
    ”scriptPubKey”:”OP_DUP OP_HASH160 a7db6f...
                   OP_EQUALVERIFY OP_CHECKSIG”}]}
```


Question: Sender and receiver are the same; same amount (no time stamps).

Can 2 transactions be exactly the same?

Pay-to-Script-Hash

Bob wants to implement a multi-key/signature scheme in his company:

- Bob tells Alice the hash of a locking script:
- Alice sends the payment to this "hash address"

- Bob has to supply the locking script matching this hash, and the unlocking script

Pay-to-Script-Hash

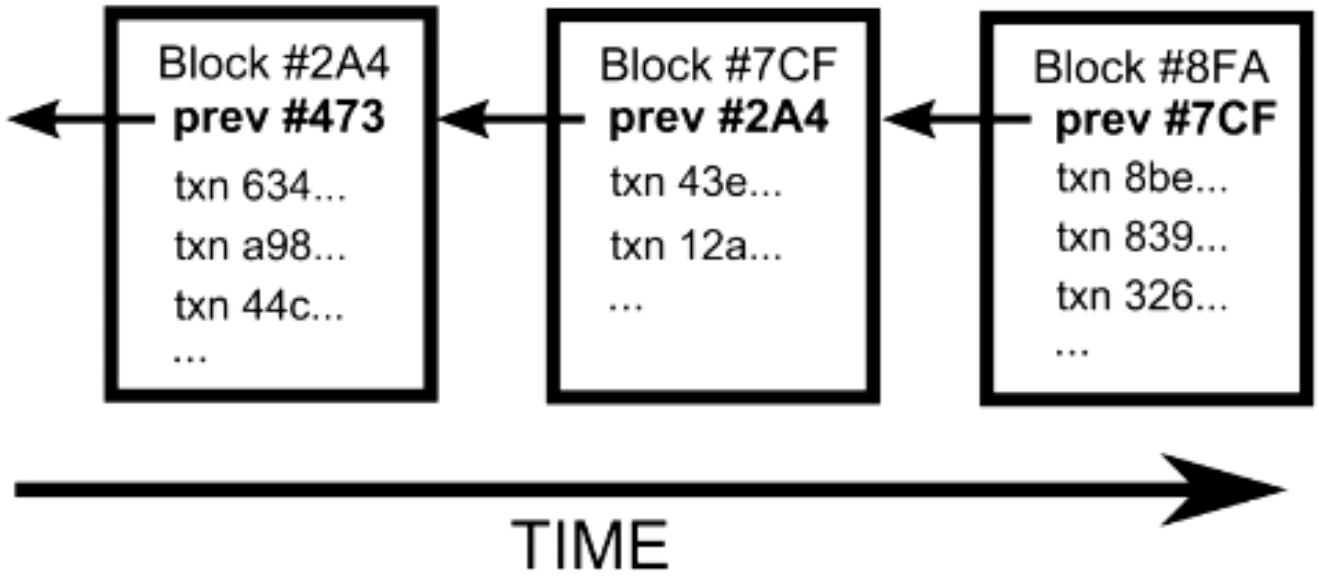
Bob wants to implement a multi-key/signature scheme in his company:

- Bob tells Alice the hash of a locking script:
- Alice sends the payment to this "hash address"

- Bob has to supply the locking script matching this hash, and the unlocking script

- Bob can use this payment to implement 2-out-of-3 signature procedures

Blockchain (Public Ledger)



- each block is hashed and contains a reference to the earlier block; "validates" potentially more than one transaction

Proof-of-Work

The idea is counterintuitive and involves a combination of two ideas:

- to (artificially) make it computationally costly for network users to validate transactions, and
- to reward them for trying to help validate transactions

Proof-of-Work

The idea is counterintuitive and involves a combination of two ideas:

- to (artificially) make it computationally costly for network users to validate transactions, and
- to reward them for trying to help validate transactions

this is called mining: whoever validates a transaction will be awarded with 50 bitcoins — this halves every 210,000 transactions or roughly every 4 years (currently 25 BC); no new bitcoins after 2140 – then only transaction fees

Solving Puzzles

Given a string, say ”Hello, world!”, what is the salt so the hash starts with a long run of zeros?

```
h(”Hello, world!0”) =

1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64
```

Solving Puzzles

Given a string, say ”Hello, world!”, what is the salt so the hash starts with a long run of zeros?

```
h(”Hello, world!0”) =
1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64
h(”Hello, world!1”) =
e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e948a9332a7d8
```

Solving Puzzles

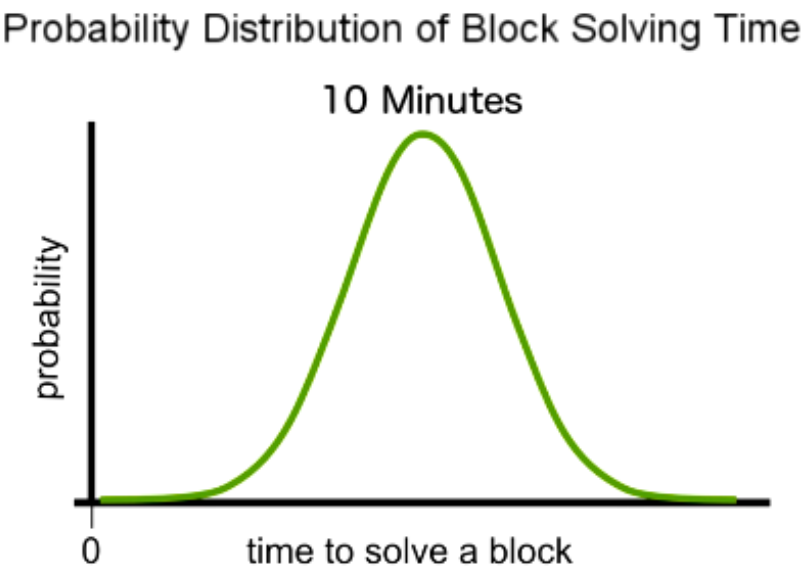
Given a string, say ”Hello, world!”, what is the salt so the hash starts with a long run of zeros?

```
h(”Hello, world!0”) =
1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64
h(”Hello, world!1”) =
e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e948a9332a7d8
…
h(”Hello, world!4250”) =
0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9
```

Hardness

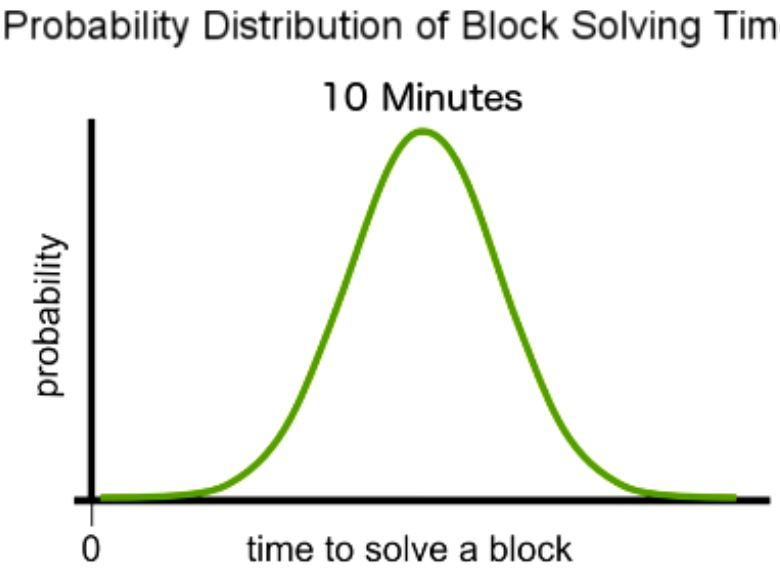
If we want the output hash value to begin with 10 zeroes, say, then we will need, on average, to try $16^{10} \approx 10^{12}$ different salts before we find a suitable nonce.

Hardness can be controlled by setting a target (maximum number).



How to Adjust the Target?

- every 2016 blocks the hardness is adjusted (app 2 weeks)



$$\text{New Difficulty} \stackrel{\text{def}}{=} \text{Old Difficulty} * \frac{\text{Actual time for the last 2016 blocks}}{20160}$$

Hardness

- for example block #277,316 has the hardness

$$\texttt{0x1903a30c}$$

where `19` is the exponent and `03a30c` is the coefficient.

$$target \stackrel{\text{def}}{=} coeffcient * 2^{8*(exponent-3)}$$

in this example the hash has to be smaller that

```
0x0000000000000003A30C000000000000
  0000000000000000000000000000000000
```

Hardness

- for example block #277,316 has the hardness

`0x1903a30c`

where `19` is the exponent and `03a30c` is the coefficient.

$$target \stackrel{\text{def}}{=} coeffcient * 2^{8*(exponent-3)}$$

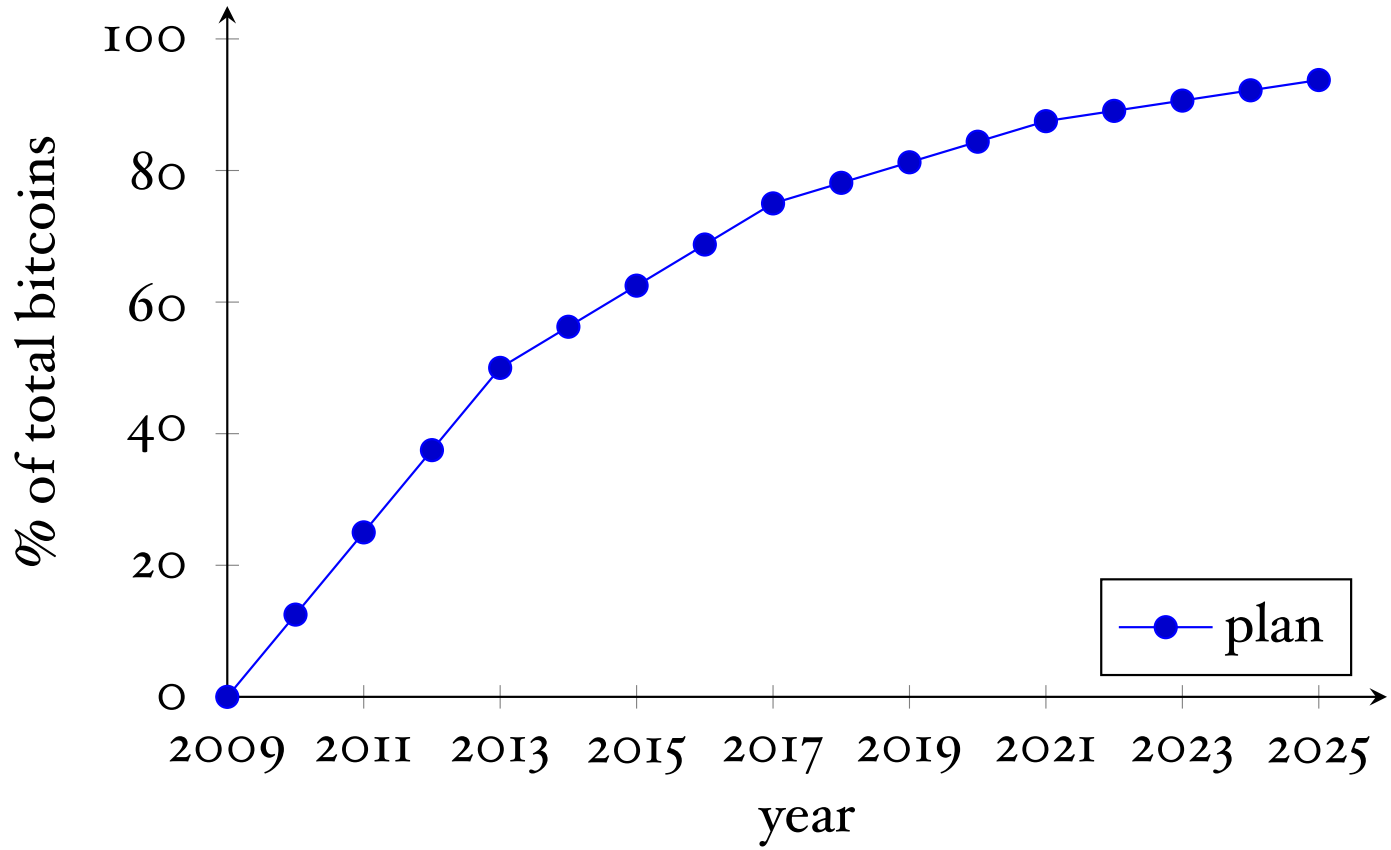
in this example the hash has to be smaller that

```
0x0000000000000003A30C000000000000
  0000000000000000000000000000000000
```

It is fun to see that nowadays mining equipment is so efficient that the hardness is closely related to the cost of electricity.

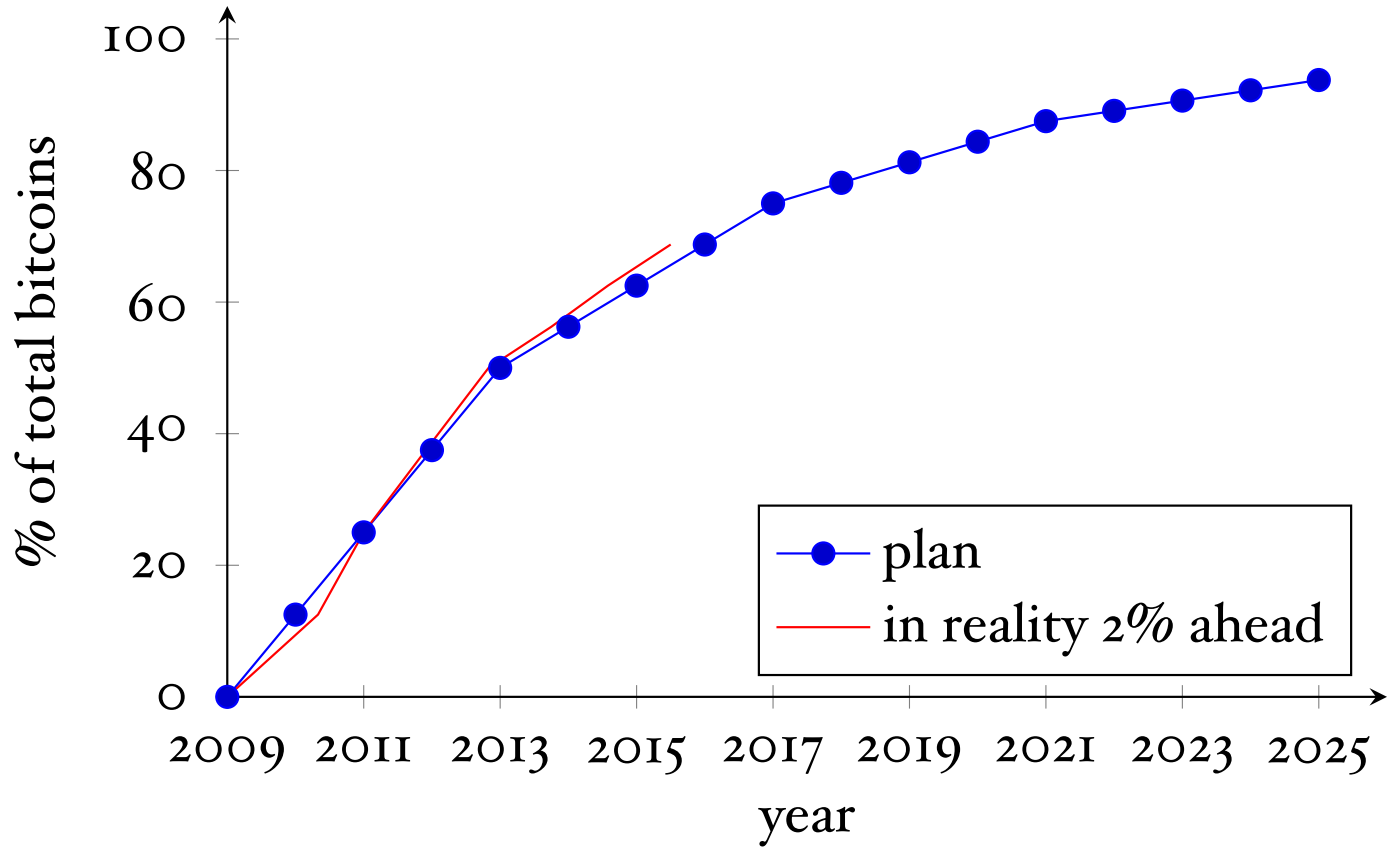
Bitcoin Schedule

- every 210000 blocks the amount of bitcoins to be mined halves ("reward era")



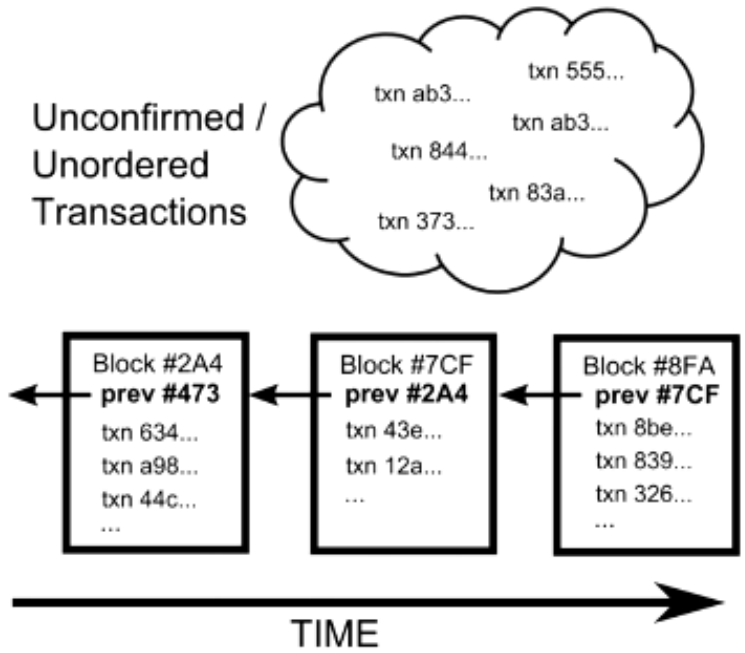
Bitcoin Schedule

- every 210000 blocks the amount of bitcoins to be mined halves ("reward era")



Order of Transactions

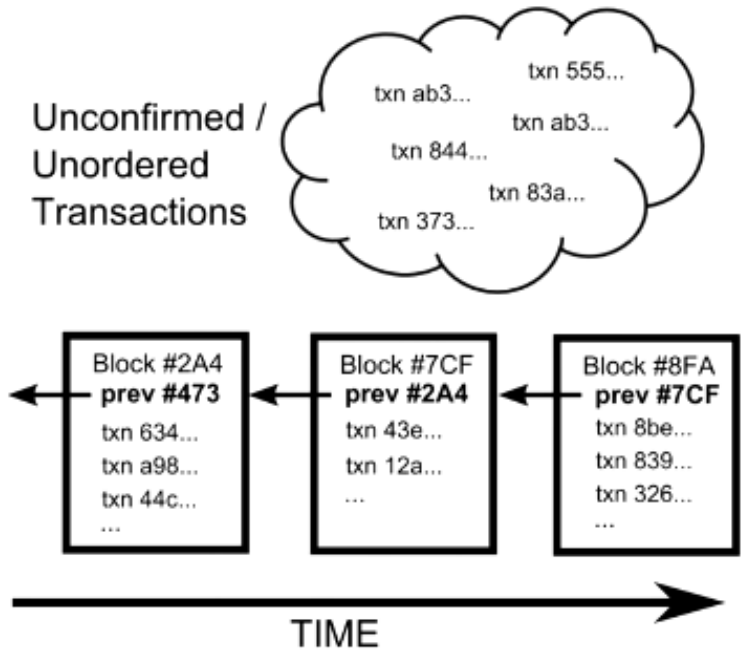
If we don't have such an ordering at any given moment then it may not be clear who owns which Bitcoins.



Say, miner David is lucky and finds a suitable salt to confirm the transactions. Celebration!

Order of Transactions

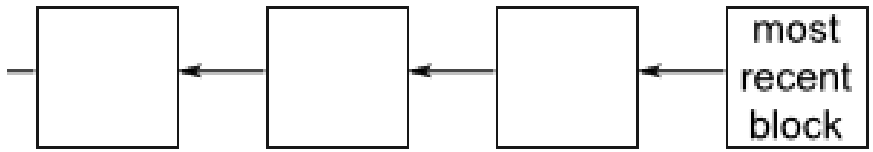
If we don't have such an ordering at any given moment then it may not be clear who owns which Bitcoins.



Say, miner David is lucky and finds a suitable salt to confirm the transactions. Celebration! ??

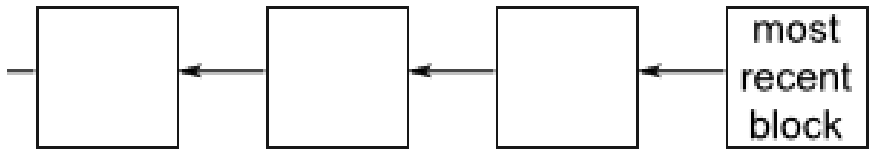
Forks

Typically the blockchain will look as follows

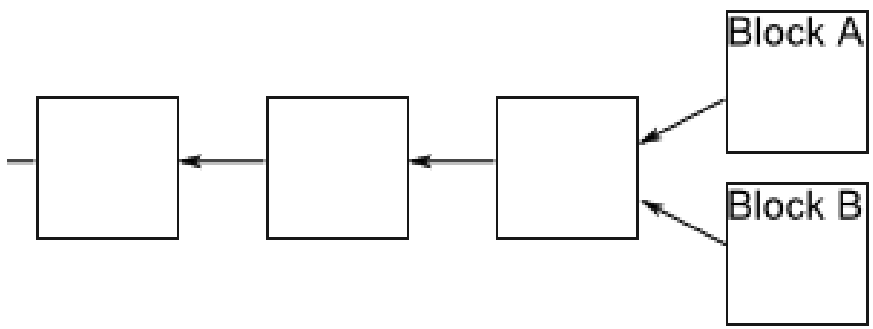


Forks

Typically the blockchain will look as follows

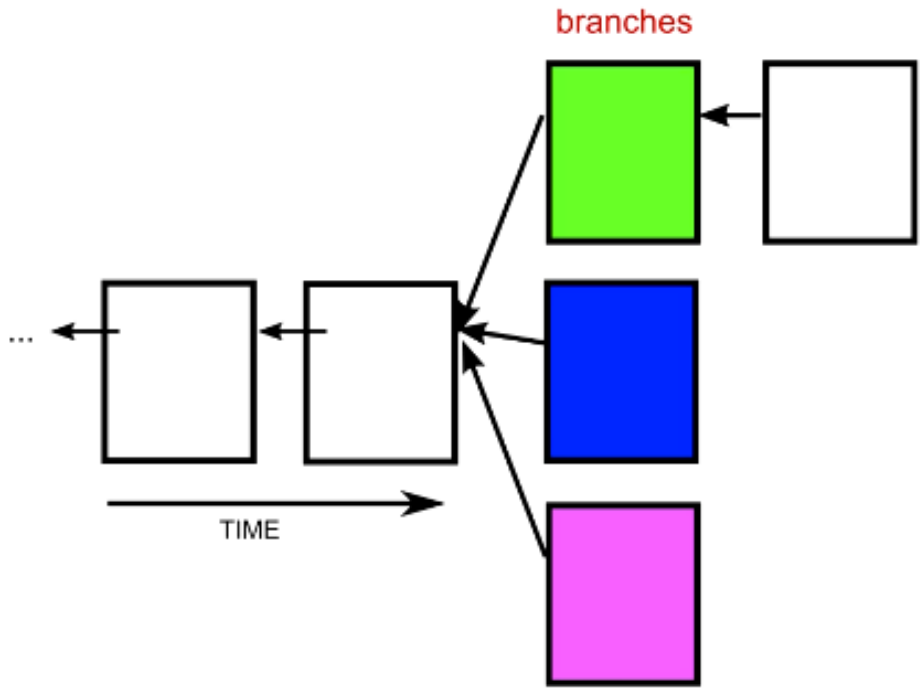


But every so often there is a fork



...bugger this is exactly what we are trying to avoid

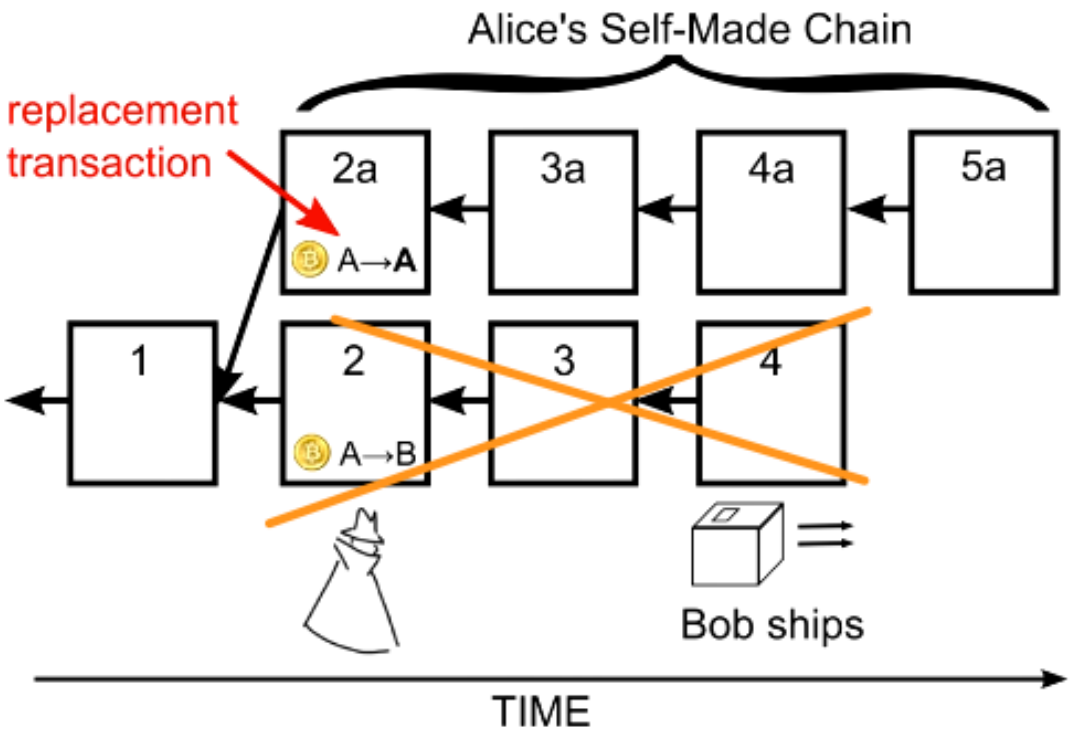
The tie is broken if another block is solved



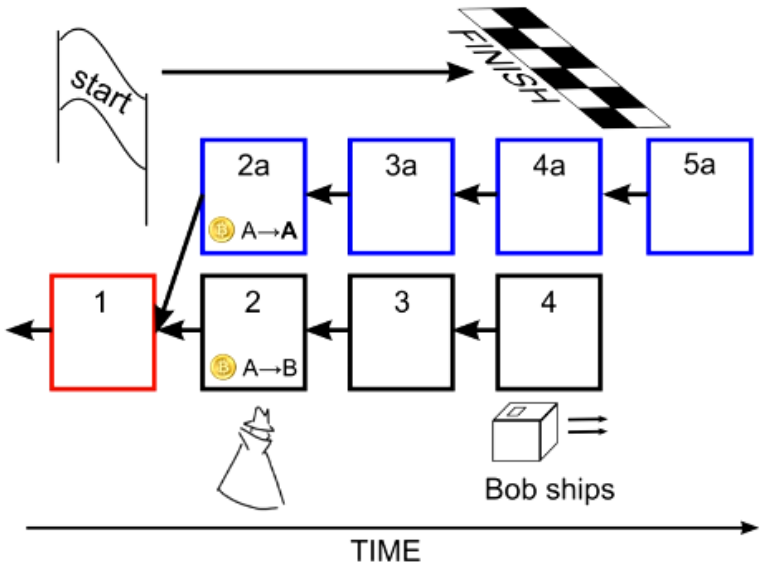
The rule is: if a fork occurs, people on the network keep track of all forks. But at any given time, miners only work to extend whichever fork is longest in their copy of the block chain.

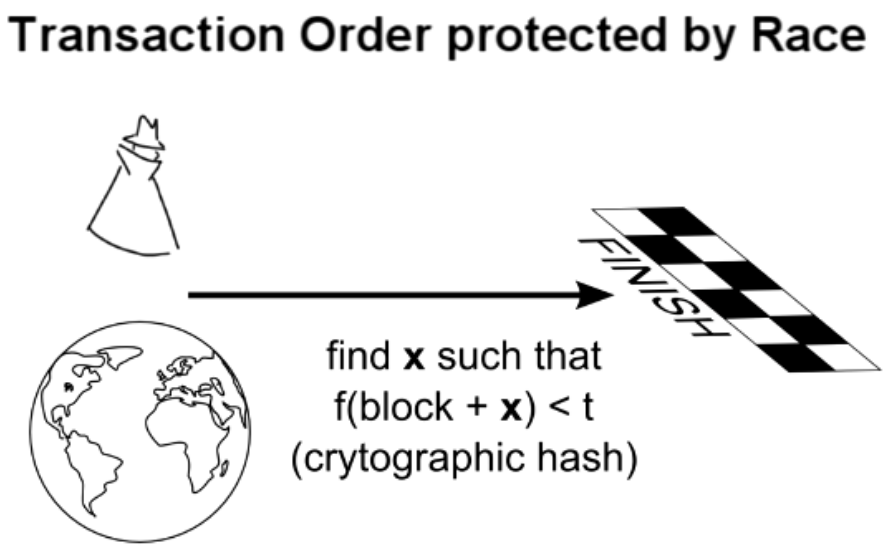
Double Spending

So if Alice wants to fake it, she needs to produce a longer chain:

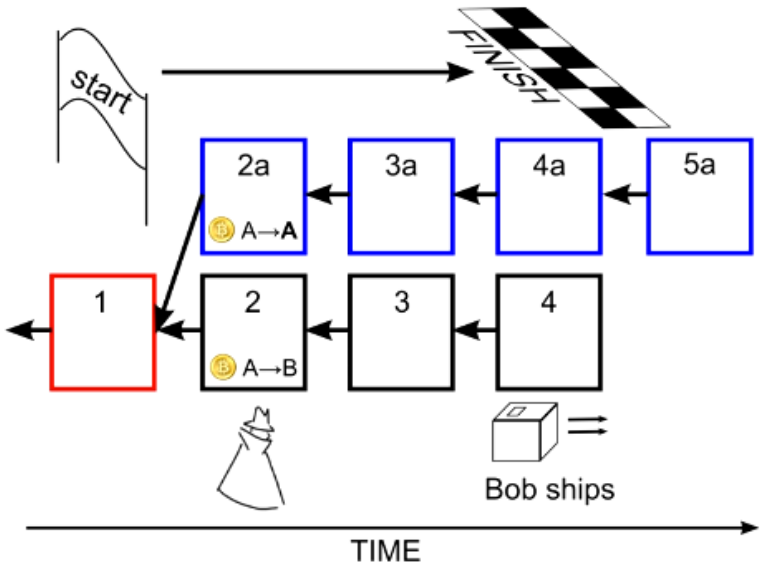


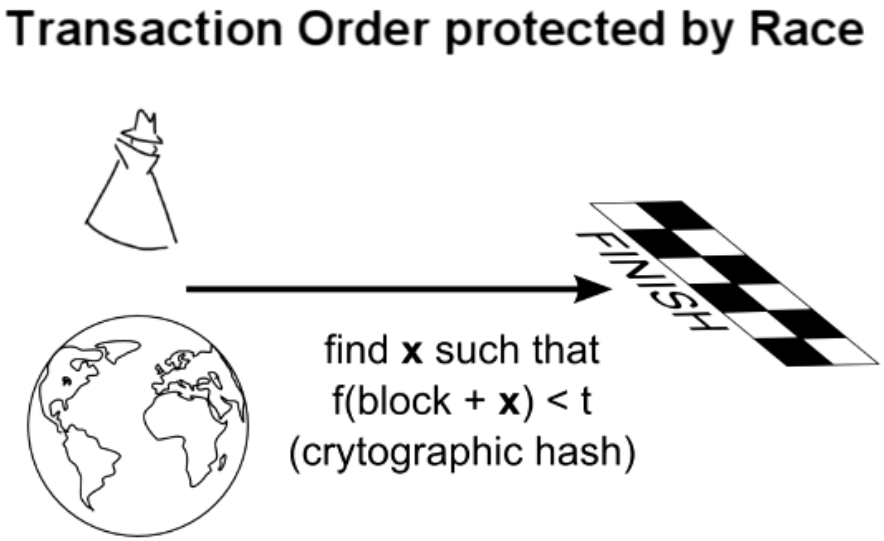
Racing Against the World





Racing Against the World





A transaction is "confirmed" if:

(1) it is part of a block in the longest fork, and (2) at least 5 blocks follow it in the longest fork. In this case we say that the transaction has "6 confirmations".

(might take 1h+...but for creditcards you have 6 months chargeback)

Mining Pools

On average, it would take several years for a typical computer to solve a block, so an individual's chance of ever solving one before the rest of the network, which typically takes 10 minutes, is negligibly low.

Mining Pools

On average, it would take several years for a typical computer to solve a block, so an individual's chance of ever solving one before the rest of the network, which typically takes 10 minutes, is negligibly low.

Many people join groups called mining pools that collectively work to solve blocks, and distribute rewards based on work contributed. These act somewhat like lottery pools among co-workers, except that some of these pools are quite large, and comprise more than 20% of all the computers in the network.

BTCC, the largest mining pool, has limited its members to not solve more than 6 blocks in a row. https://blockchain.info/pools

Bitcoins for Real

- you need a public-private key (the hash of the public key to determines your bitcoin address)
- if you want to receive bitcoins, you publicise this address
- there are 2^{160} possibilities
  (no check for duplicates)

Bitcoins for Real

- you need a public-private key (the hash of the public key to determines your bitcoin address)
- if you want to receive bitcoins, you publicise this address
- there are 2^{160} possibilities
  (no check for duplicates)

- transactions contain "payment scripts"
  (non-Turing-complete scripting language)

  simplest script: pay-to-public-key

Multi-Signature Addresses

- ...Bitcoin Improvement Proposal

Multi-Signature Addresses

- ...Bitcoin Improvement Proposal

- pay-to-public-key (explained so far)
- pay-to-script-hash (since 2012)

Multi-Signature Addresses

- ...Bitcoin Improvement Proposal

- pay-to-public-key (explained so far)
- pay-to-script-hash (since 2012)

  can specify: requires M out of N signatures

  for example
  1-of-2: me and my wife, or

  2-of-2 in banking/companies

Dispute Mediation

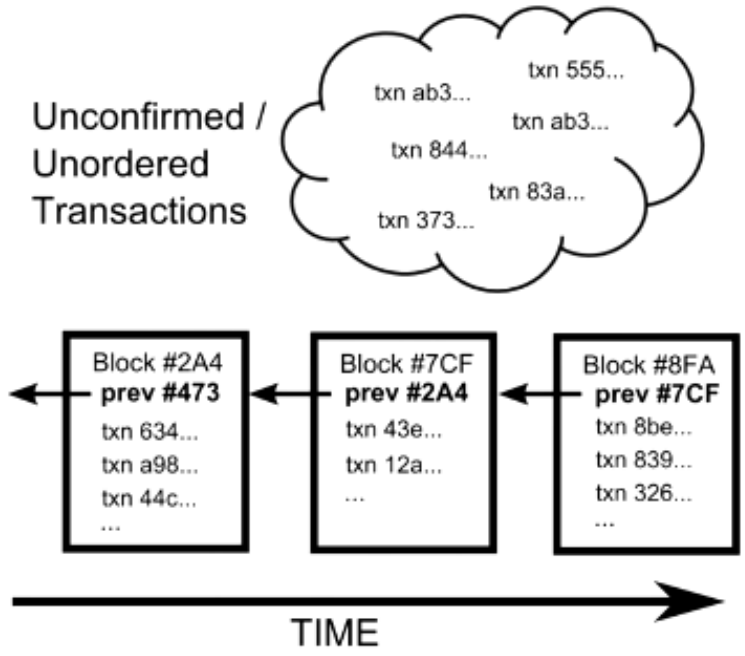
- say, client and (online) merchant do not trust each other

Dispute Mediation

- say, client and (online) merchant do not trust each other

- 2-of-3: mutually trusted escrow service
  1. client sends money to 2-of-3 transaction
  2. merchant sends out goods
  3. if goods are OK, client sends signed transaction to merchant, merchant can sign and receive the money (publish in blockchain)
  4. if goods are defective, merchant sends signed transaction to client, client can sign and receive the money back
  5. if client and merchant disagree, then they ask escrow servive who signs a transaction and sends it to "winning" party

A Block in the Blockchain



- each block is hashed and contains a reference to the earlier block
- contains the "salt" and address of whoever solved the puzzle

Transaction History

you can follow back the transaction history until you reach either

- the genesis block (a transaction without input of 50 bitcoins), or
- a coinbase transaction (this is the reward of the miner who validated a block of transactions in the blockchain)

Lost Bitcoins?

- somebody needs to be able to generate a key-pair for the signature (for this you need the private key)

- somebody spends your bitcoins fraudulently (you cannot charge them back)... bad luck
- you can send bitcoins to a "non-existing" address (Mt. Gox)

Good Points

An attacker can't:

- reverse other people's transactions
- change the number of coins generated per block
- create coins out of thin air
- send coins that never belonged to an attacker
- you cannot meddle with the "history"

The system can be scaled to all world transactions.

Take Home Points

- Don't gamble! I am not a first mover in such things.
- Cool idea, but I am sure there will be a Bitcoin 2.0.
- It still depends on a lot of old-fashioned security (e.g. keeping private-keys secret)

- Having now the knowledge how it works, go back and listen to what people/media make of it.