

# **How Bitcoin Achieves Decentralization**



ScroogeCoin

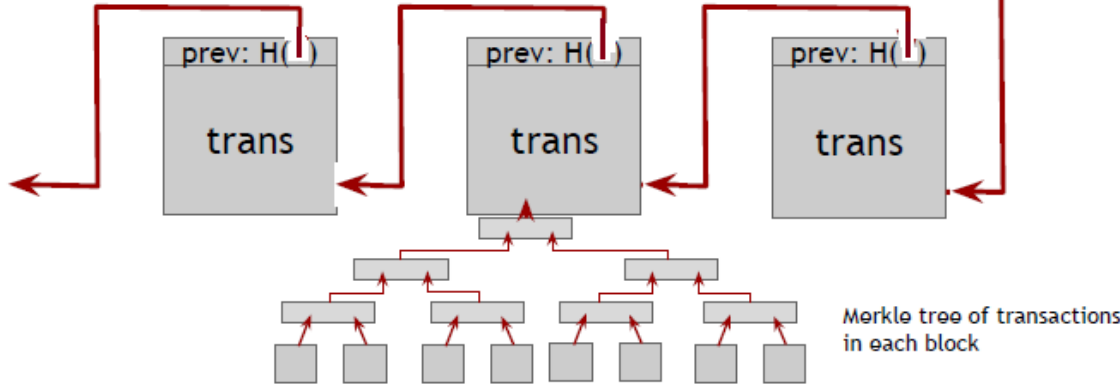
Don't worry, I'm honest.

Scrooge publishes ledger of all transactions  
(a blockchain, signed by Scrooge)



signed by  $pk_{\text{Scrooge}}$

$H(\quad)$



## Crucial question:

Can we descroogify the currency, and operate without any central, trusted party?

# Centralization vs. decentralization

# Centralization vs. decentralization

Competing paradigms that underlie many digital technologies

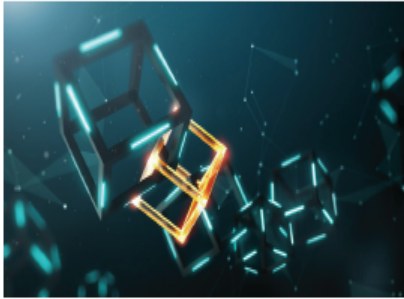
# Decentralization is not all-or-nothing

E-mail:

decentralized protocol, but dominated by  
centralized webmail services

# The path to decentralization

- technology & incentive design



Who determines the  
validity of transactions to  
be included in the ledger?



Who maintains the ledger  
of transactions? (and how?)

All  
Participants

Consensus

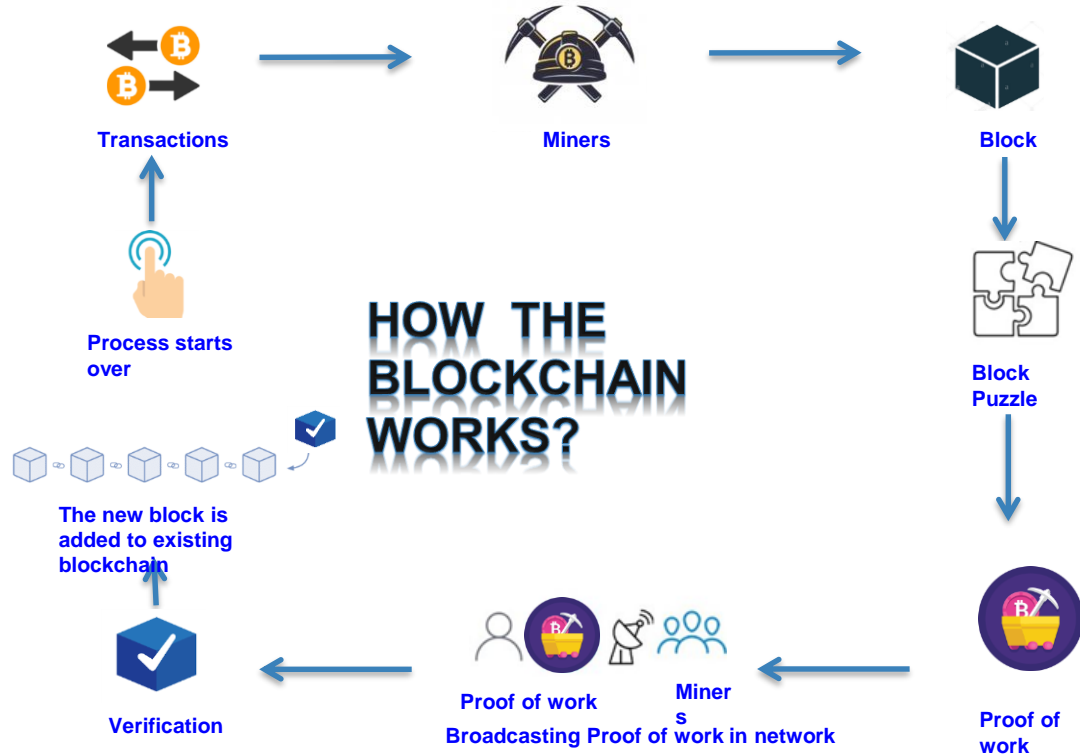
All  
Participants

Bitcoin  
Script

Who creates new Bitcoins?

Reward  
for Mining

# BLOCKCHAIN WORKING PRINCIPLE



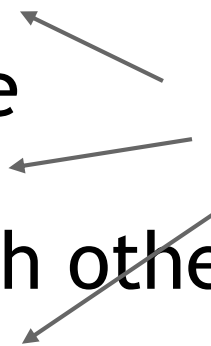


# Bitcoin blocks

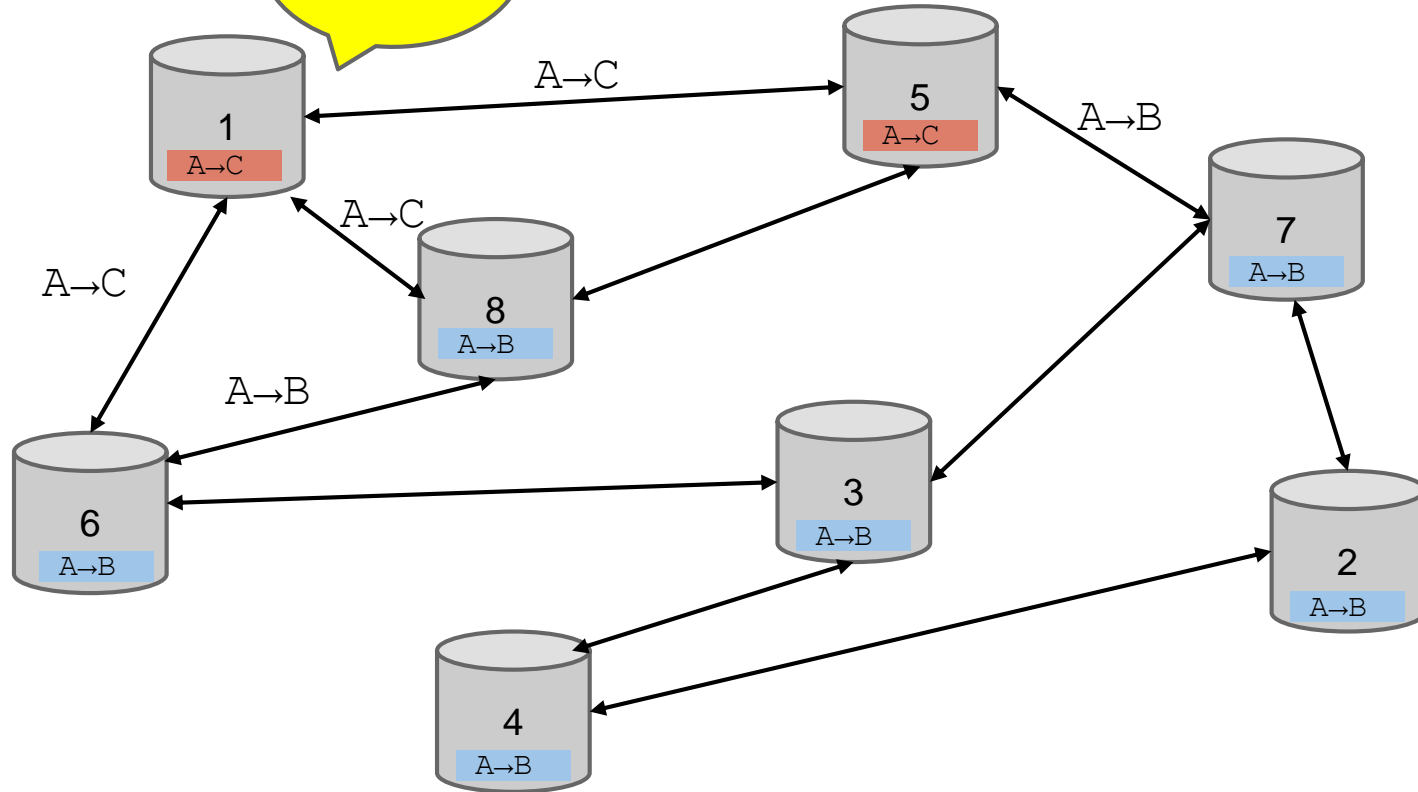
# Should I relay a proposed transaction?

- Transaction valid with current block chain
- (default) script matches a whitelist
  - Avoid unusual scripts
- Haven't seen before
  - Avoid infinite loops
- Doesn't conflict with others I've relayed
  - Avoid double-spends

Sanity checks only...  
Some nodes may ignore them!



# Nodes may differ on transaction pool



# Bitcoin blocks

Why bundle transactions together?

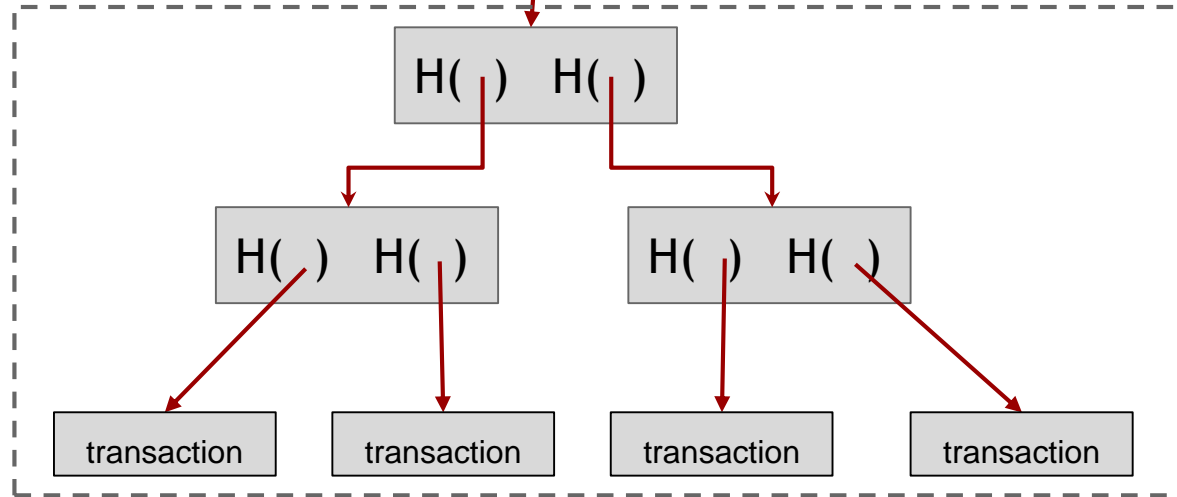
- Single unit of work for miners
- Limit length of hash-chain of blocks
  - Faster to verify history

# Bitcoin block structure

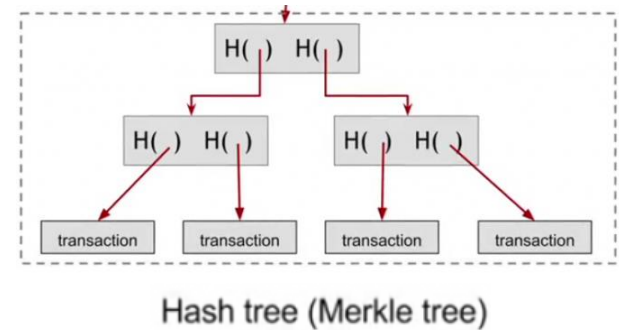
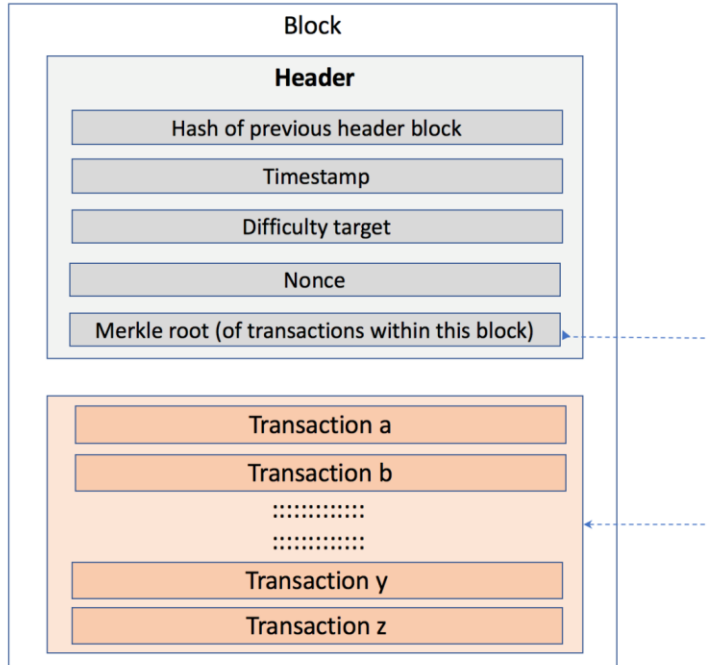
Hash chain of blocks



Hash tree (Merkle tree) of transactions in each block



# Bitcoin block structure

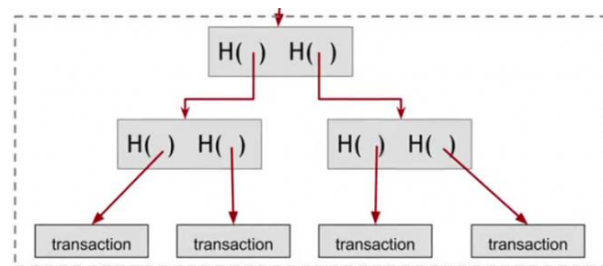


# The real deal: a Bitcoin block

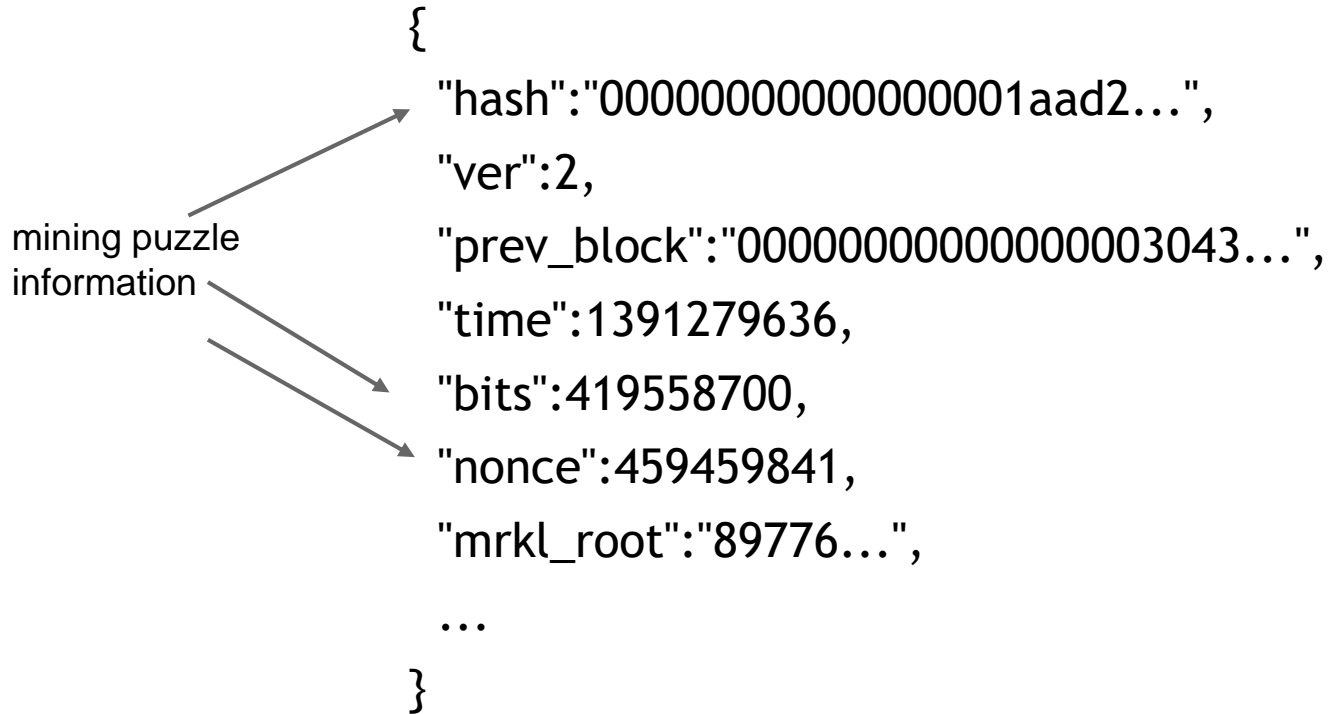
block header

```
{  
  "hash": "0000000000000001aad2...",  
  "ver": 2,  
  "prev_block": "0000000000000003043...",  
  "time": 1391279636,  
  "bits": 419558700,  
  "nonce": 459459841,  
  "mrkl_root": "89776...",  
  "n_tx": 354,  
  "size": 181520,  
  "tx": [  
    ...  
  ],  
  "mrkl_tree": [  
    "6bd5eb25...",  
    ...  
    "89776cdb..."  
  ]  
}
```

transaction data



# The real deal: a Bitcoin block header





# Distributed consensus

# Bitcoin's key challenge

Key technical challenge of decentralized e-cash: distributed consensus

or: how to decentralize ScroogeCoin

# Defining distributed consensus

The protocol terminates and all correct nodes decide on the same value

This value must have been proposed by some correct node

*Distributed consensus protocol.* There are  $n$  nodes that each have an input value. Some of these nodes are faulty or malicious. A distributed consensus protocol has the following two properties:

- It must terminate with all honest nodes in agreement on the value
- The value must have been generated by an honest node

# Why consensus is hard

Nodes may crash

Nodes may be malicious

Network is imperfect

- Not all pairs of nodes connected
- Faults in network
- Latency



No notion of global time

# Many impossibility results

- Fischer-Lynch-Paterson (deterministic nodes): consensus impossible with a single faulty node
- These results say more about the model than about the problem
- The models were developed to study systems like distributed databases

# Some well-known protocols

Example: Paxos, Raft

Makes certain compromises

Never produces inconsistent result, but can (rarely) get stuck

# Bitcoin consensus: theory & practice

Bitcoin consensus works better in practice than in theory

Theory is still catching up

BUT theory is important, can help predict unforeseen attacks

# Some things Bitcoin does differently

## Introduces incentives

- Possible only because it's a currency!

## Embraces randomness

- Does away with the notion of a specific start and end-point
- Consensus happens over long time scales — about 1 hour



# Bitcoin Consensus

# Consensus algorithm (simplified)

1. New transactions are broadcast to all nodes
2. Follow Flooding/Gossip Protocol to broadcast
3. Some nodes collect new transactions into a block
4. In each round a random node gets to broadcast its block
5. Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)
6. Nodes express their acceptance of the block by including its hash in the next block they create



Mining  
Process



Consensus  
Agreement

# Hashcash: Proof of work

- Based on the idea of HashCash, a Proof of Work concept invented by Adam Back in 1997  
(<http://www.hashcash.org/papers/hashcash.pdf>)
- Originally proposed as an anti-spam throttling mechanism
- The core idea is that before accepting a transaction, the sender must first demonstrate a “cost” via a computationally “hard” problem that can simultaneously be easily verified.
- This generally referred to as a “Proof of Work”

# Hashcash: Proof of work

- HashCash Cost Function: Interactive Vs. Non-interactive
- $s$ : service name

$$\begin{cases} \mathcal{T} \leftarrow \text{MINT}(s, w) & \text{mint token} \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) & \text{token evaluation function} \end{cases}$$

$$\left\{ \begin{array}{ll} \text{PUBLIC:} & \text{hash function } \mathcal{H}(\cdot) \text{ with output size } k \text{ bits} \\ \mathcal{T} \leftarrow \text{MINT}(s, w) & \textbf{find } x \in_R \{0, 1\}^* \textbf{ st } \mathcal{H}(s||x) \stackrel{\text{left}}{=}_w 0^k \\ & \textbf{return } (s, x) \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) & \mathcal{H}(s||x) \stackrel{\text{left}}{=}_v 0^k \\ & \textbf{return } v \end{array} \right.$$

# Hashcash: Proof of work

- Hashcash Cost Function: Interactive Vs. Non-interactive

$$\left\{ \begin{array}{ll} \mathcal{C} \leftarrow \text{CHAL}(s, w) & \text{server challenge function} \\ \mathcal{T} \leftarrow \text{MINT}(\mathcal{C}) & \text{mint token based on challenge} \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) & \text{token evaluation function} \end{array} \right.$$

$$\left\{ \begin{array}{ll} \mathcal{C} \leftarrow \text{CHAL}(s, w) & \textbf{choose } c \in_R \{0, 1\}^k \\ & \textbf{return } (s, w, c) \\ \mathcal{T} \leftarrow \text{MINT}(\mathcal{C}) & \textbf{find } x \in_R \{0, 1\}^* \textbf{ st } \mathcal{H}(s||c||x) \stackrel{\text{left}}{=}_w 0^k \\ & \textbf{return } (s, x) \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) & \mathcal{H}(s||c||x) \stackrel{\text{left}}{=}_v 0^k \\ & \textbf{return } v \end{array} \right.$$

Prevents **DOS motivated attack** or pre-computation attacks

# The real deal: a Bitcoin block

block header

transaction data

```
{
  "hash": "0000000000000001aad2...",
  "ver": 2,
  "prev_block": "0000000000000003043...",
  "time": 1391279636,
  "bits": 419558700,
  "nonce": 459459841,
  "mrkl_root": "89776...",
  "n_tx": 354,
  "size": 181520,
  "tx": [
    ...
  ],
  "mrkl_tree": [
    "6bd5eb25...",
    ...
    "89776cdb..."
  ]
}
```

# Proof of Work

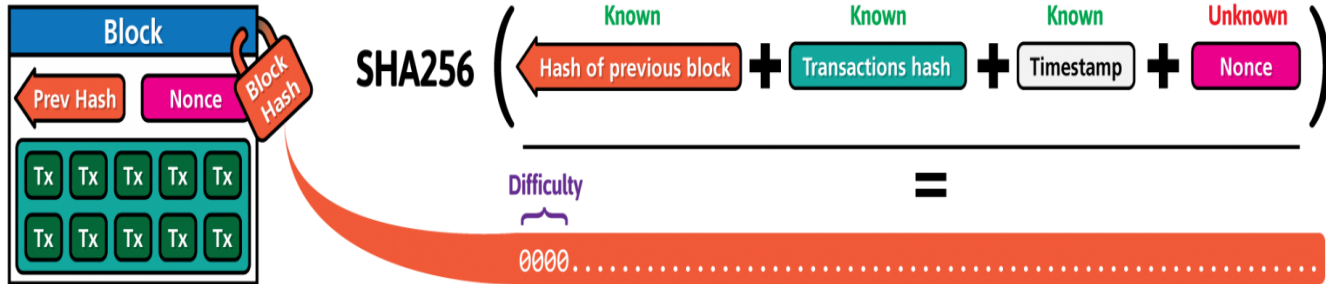
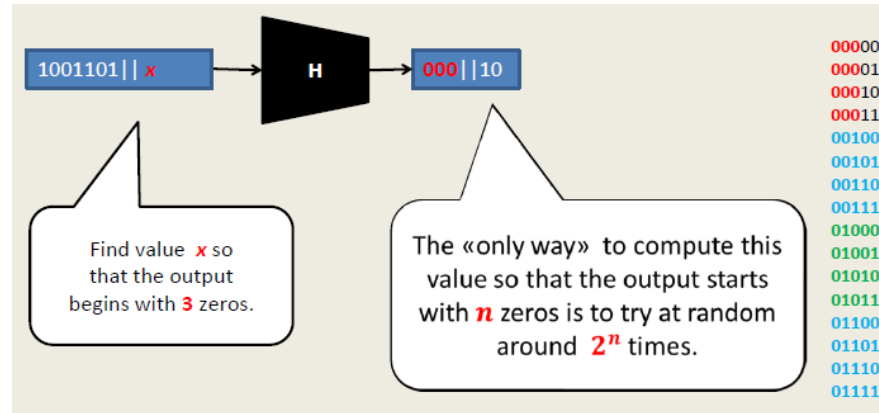
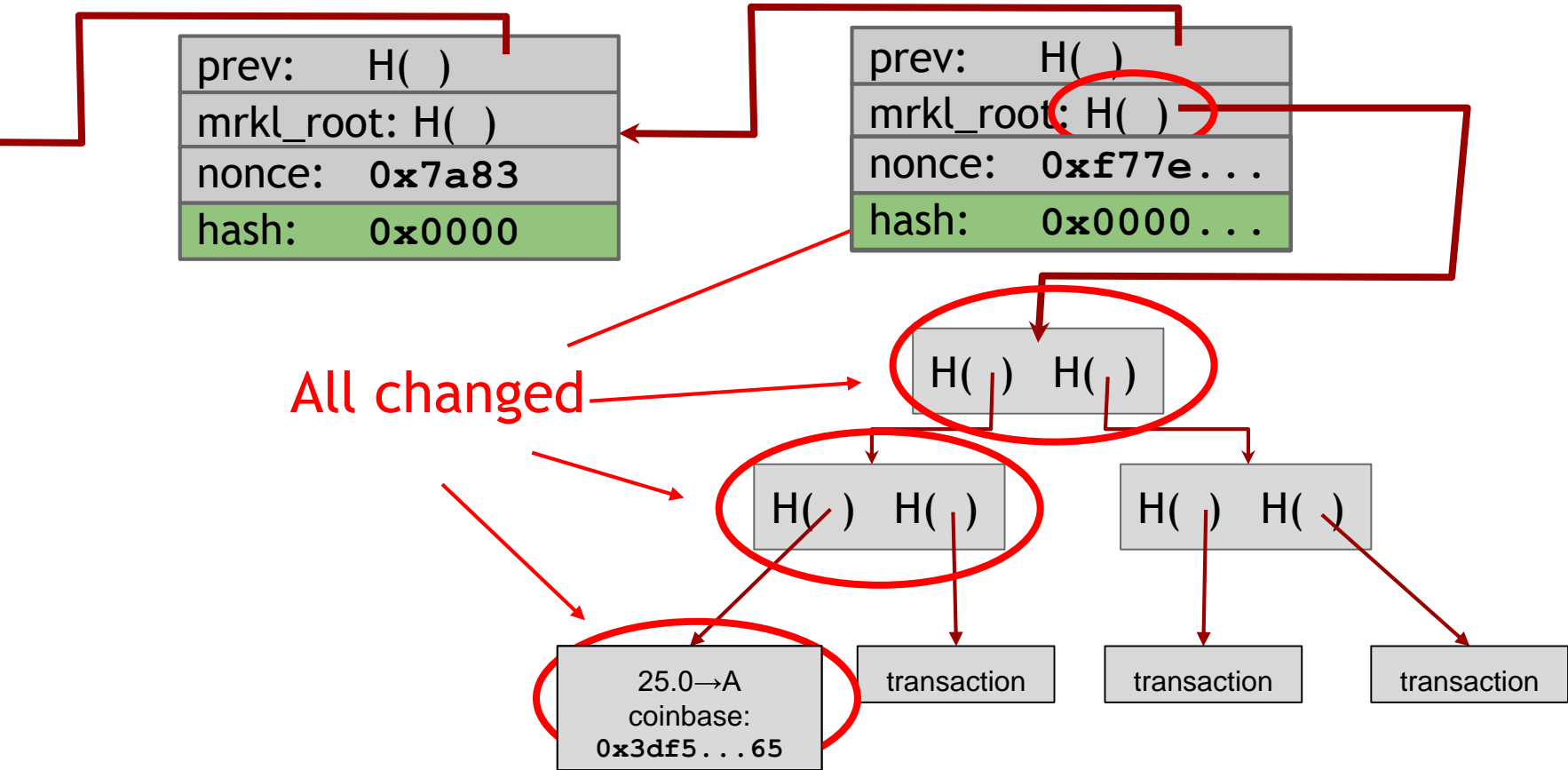


Illustration by CryptoGraphics.info



Proof of Work [Back2002]

# Finding a valid block: Proof-of-Work





# Coinbase

redeeming nothing

arbitrary

```
"in":[
  {
    "prev_out":{
      "hash":"000000.....0000000",
      "n":4294967295
    },
    "coinbase":"..."
  },
  "out":[
    {
      "value":"25.03371419",
      "scriptPubKey":"OPDUP OPHASH160 ... "
    }
  ]
}
```

Null hash pointer

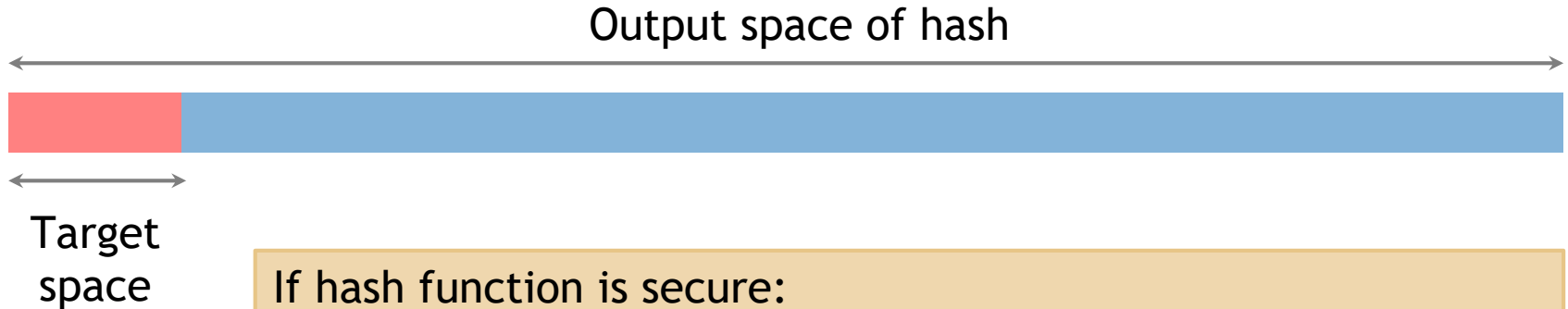
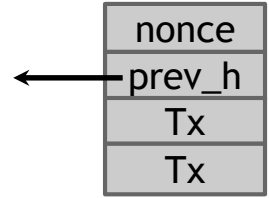
block reward

transaction fees

# Hash puzzles: Bitcoin Proof-of-Work

To create block, find nonce s.t.

$H(\text{nonce} \parallel \text{prev\_hash} \parallel \text{tx} \parallel \dots \parallel \text{tx})$  is very small



If hash function is secure:  
only way to succeed is to try enough nonces until you get lucky

Target space is 1% of overall output space, You would have to try 100 nonces before you are likely to get valid result.

# Race conditions

Transactions or blocks may *conflict*

- Default behavior: accept what you hear first
- Network position matters
- Miners may implement other logic!


# Block propagation nearly identical

Relay a new block when you hear it if:

- Block meets the hash target
- Block has all valid transactions
  - Run *all* scripts, even if you wouldn't relay
- Block builds on current longest chain
  - Avoid forks



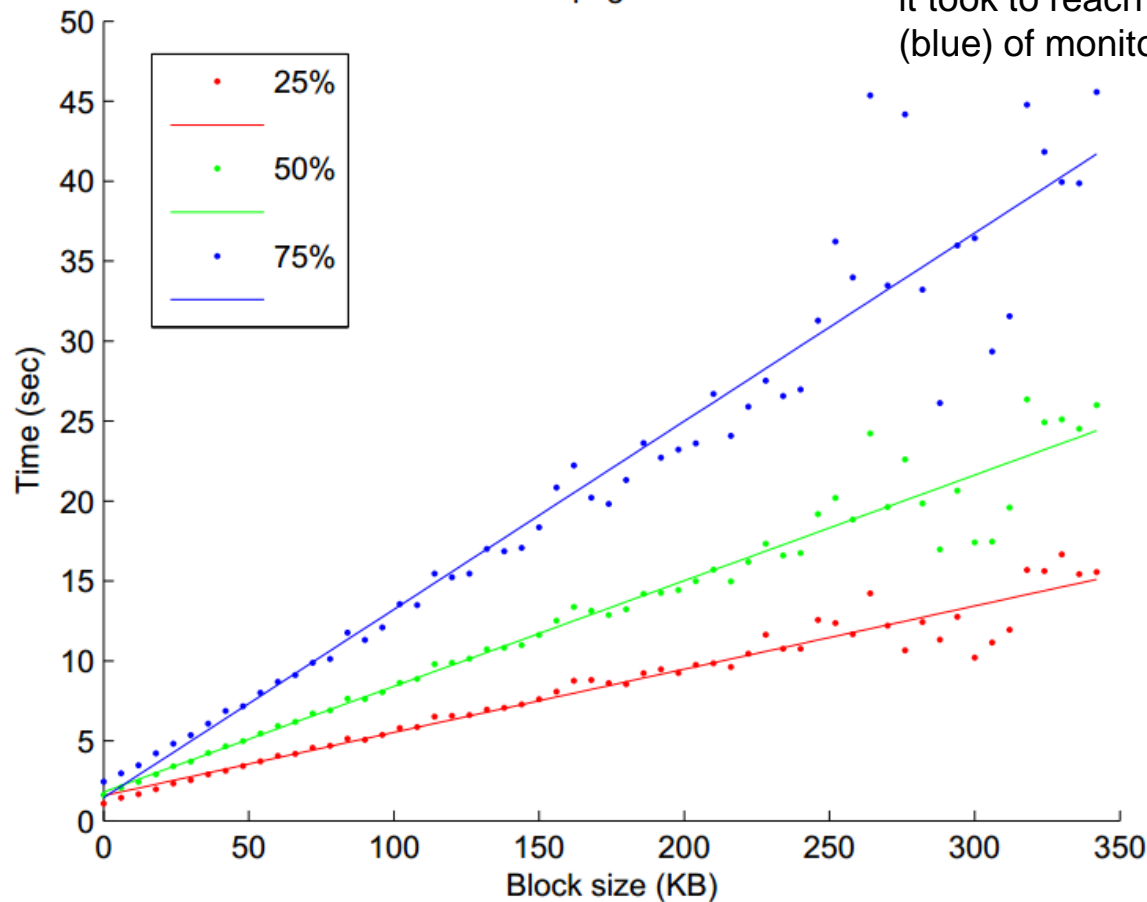
Bitcoin Mining



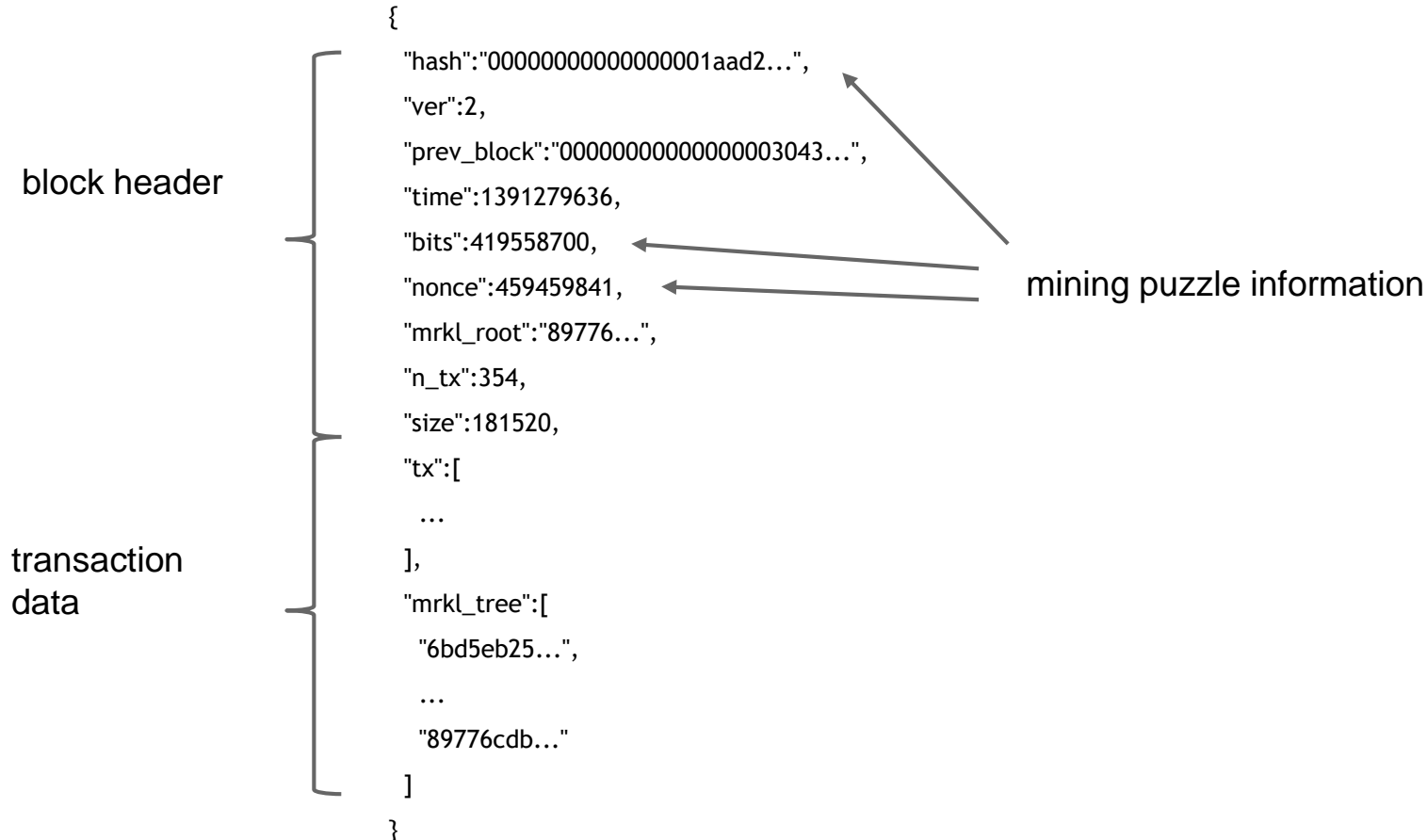
Sanity check  
Also may be ignored...

## Block Propagation Times

The relation between the block size and the time it took to reach 25% (red), 50% (green), and 75% (blue) of monitored nodes



# The real deal: a Bitcoin block



# Proof of Work

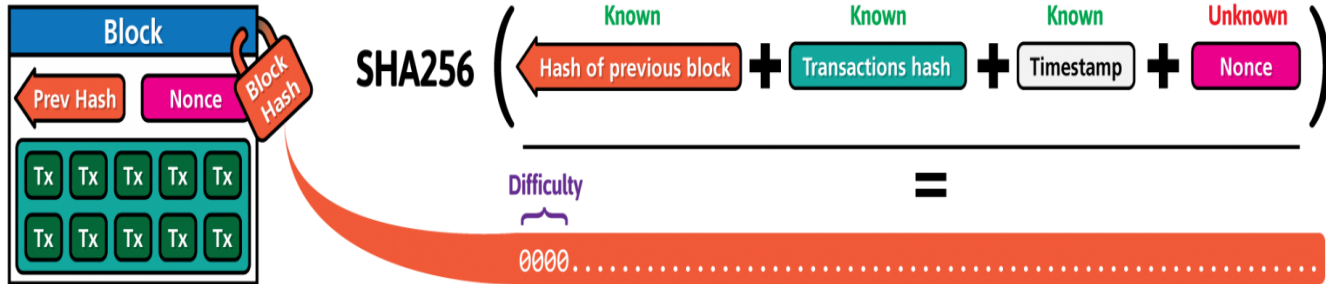
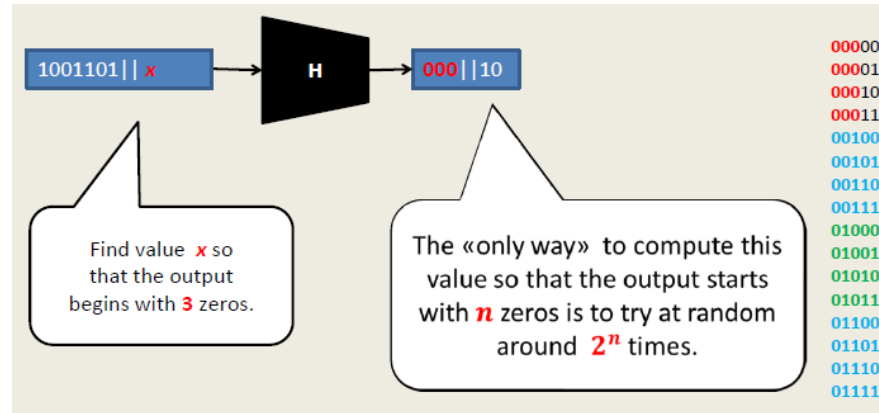


Illustration by CryptoGraphics.info



Proof of Work [Back2002]

# Mining Difficulty (Max Target)

**Bits:** 486604799 = 1D00FFFF

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

```
target = 00FFFF * 2^(8 * (1D - 3))
```

```
target = 00FFFF * 2 ^ (8*1A)
```

```
target = 00FFFF * 2 ^ D0
```

```
|0x00ffff * 2** (8*(0x1d - 3)) = 0x00000000FFFF000000000000000000000000000000000000000000000000000
```

## Block #0

Summary	
Number Of Transactions	1
Output Total	50 BTC
Estimated Transaction Volume	0 BTC
Transaction Fees	0 BTC
Height	0 (Main Chain)
Timestamp	2009-01-03 18:15:05
Difficulty	1
Bits	486604799
Size	0.285 kB
Weight	0.896 kWU
Version	1
Nonce	2083236893
Block Reward	50 BTC



# Mining Pseudocode

More Info: <https://en.bitcoin.it/wiki/Difficulty>

```
TARGET = (65535 << 208) / DIFFICULTY;
coinbase_nonce = 0;
while (1) {
    header = makeBlockHeader(transactions, coinbase_nonce);
    for (header_nonce = 0; header_nonce < (1 << 32); header_nonce++){
        if (SHA256(SHA256(makeBlock(header, header_nonce))) <
            TARGET)
            break; //block found!
    }
    coinbase_nonce++;
}
```

Max Target: An application-defined constant, which sets the target hash corresponding to the lowest possible difficulty, 1

# CPU mining

```
while (1) {  
    HDR[kNoncePos]++;  
    IF (SHA256 (SHA256 (HDR) ) < (65535 << 208) / DIFFICULTY)  
        return;  
}
```

- Hashes are 256-bit integers. So, TOTAL output size:  $2^{256}$ .
- The current TARGET is “max\_target/difficulty”, where max\_target is  $(65535 * 2^{208})$ .
- Therefore, fraction of output space is TARGET/TOTAL. Therefore, TOTAL/TARGET= $(2^{256} * \text{difficulty} / \text{max\_target})$  no. of hashes are needed on average to find a block.
- This is done over 600 sec, considering previous 2016 blocks.
- Global Hashrate:  
$$\begin{aligned} & (2^{256} * \text{difficulty} / \text{max\_target}) / 600 \\ &= (2^{256} * \text{difficulty} / 65535 * 2^{208}) / 600 \\ &= (2^{48} * \text{difficulty} / 65535) / 600 \\ &= \text{difficulty} * 7158388.055 \end{aligned}$$



[BitcoinWisdom](#) | **[Bitcoin Difficulty](#)** | [Litecoin Difficulty](#) | [Bitcoin Calculator](#) | [Litecoin Calculator](#) | [Markets](#)

Bitcoin Difficulty: 6,068,891,541,676  
Estimated Next Difficulty: 5,933,343,057,673 (-2.23%)  
Adjust time: After 1529 Blocks, About 11.0 days  
Hashrate(?): 40,575,440,550 GH/s  
1 block: 10.3 minutes  
Block Generation Time(?): 3 blocks: 31.0 minutes  
6 blocks: 1.0 hours  
Updated: 13:10 (2.6 hours ago)

1

1

1

0

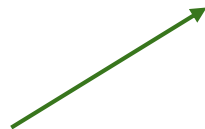
### Bitcoin Hash Rate (2 Months)



# Setting the mining difficulty

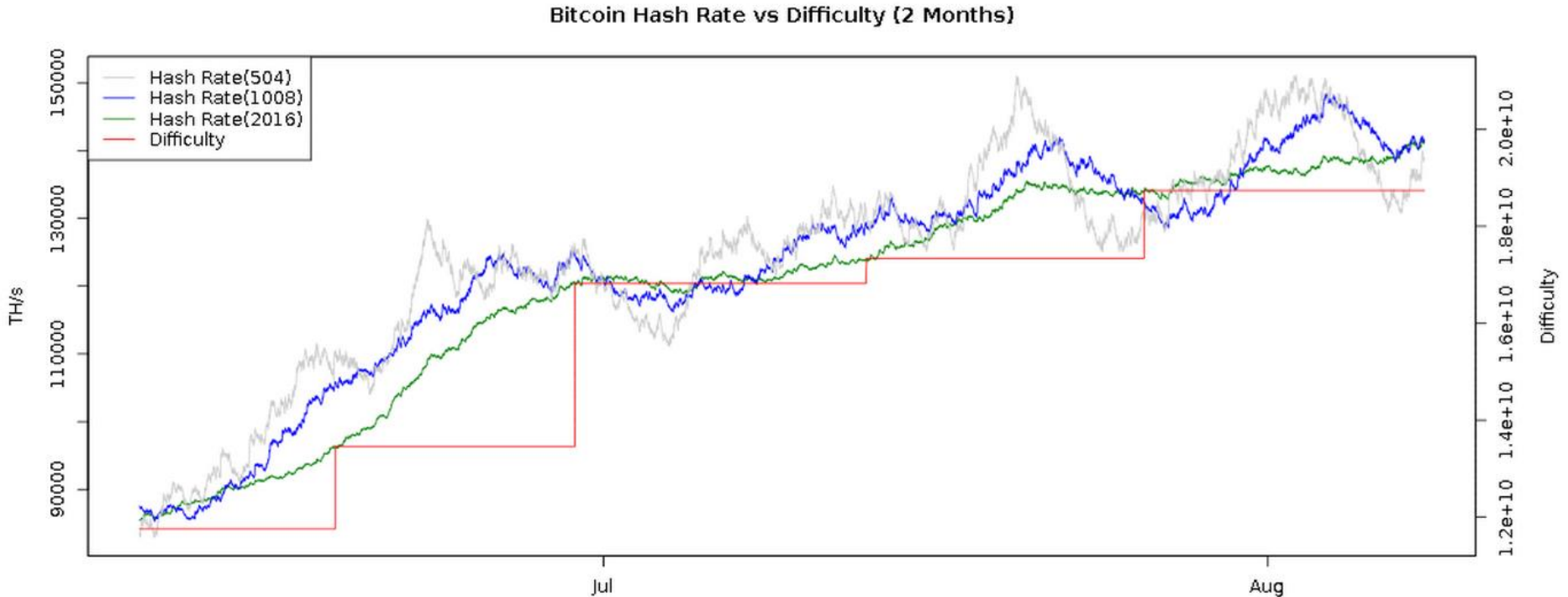
Every two weeks, compute:

```
next_difficulty= current_difficulty * (2 weeks)/(time to mine last 2016 blocks)
```



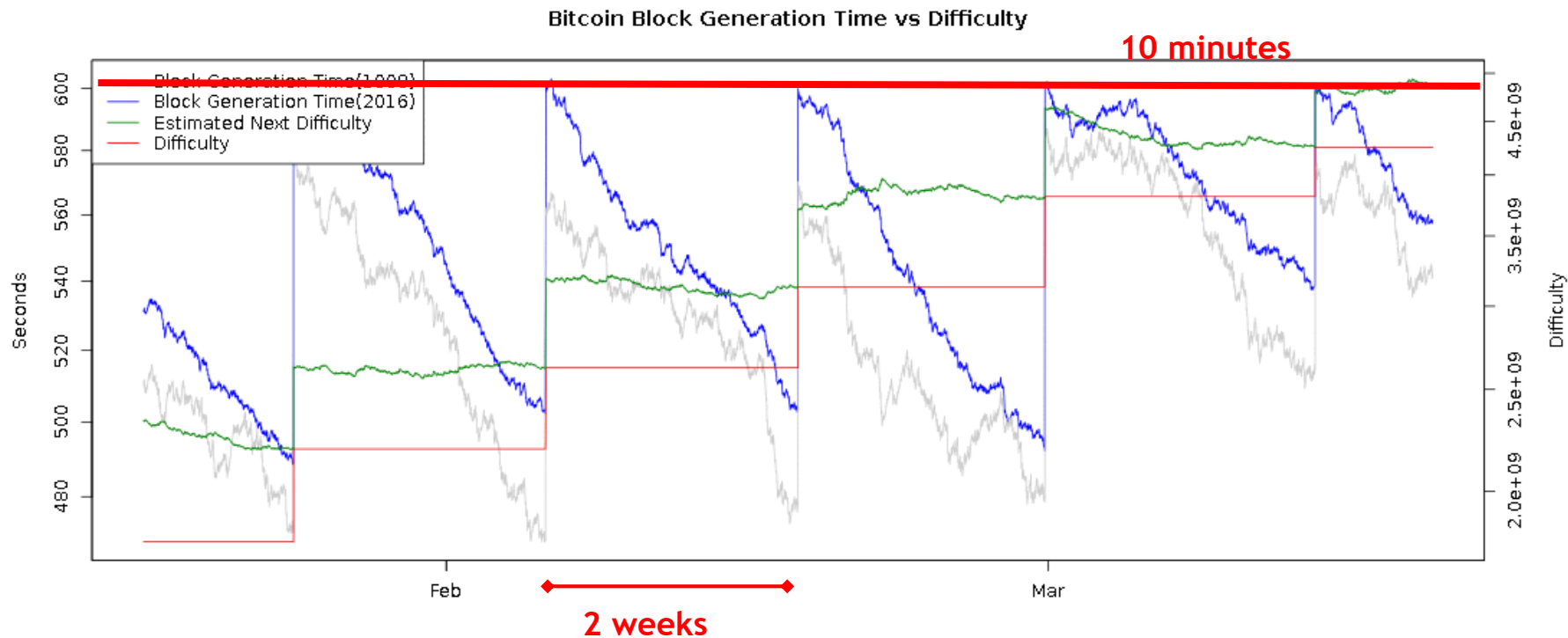
Expected number of blocks in 2 weeks at 10 minutes/block

# Mining difficulty over time



Note that the y-axis begins at 80,000 TeraHashes/s. The hash rate is averaged over 2016/1008/504 blocks

# Time to find a block



# Mining hardware

# Evolution of mining



CPU



GPU



FPGA



ASIC



gold pan



sluice box



placer mining



pit mining



# CPU mining

```
while (1){  
    HDR[kNoncePos]++;  
    IF (SHA256(SHA256(HDR)) < (65535 << 208) / DIFFICULTY)  
        return;  
}
```

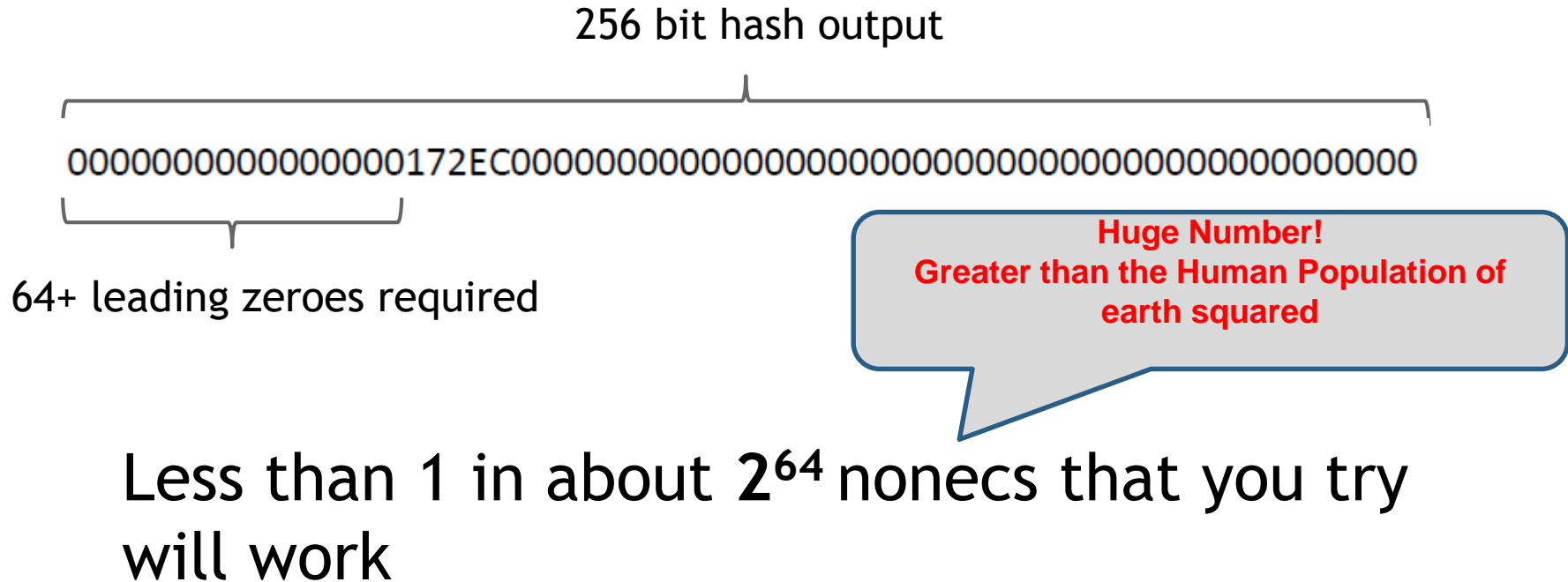
↑  
two hashes

Throughput on a high-end PC = 10-20 MHz  $\approx 2^{24}$  Hashes/Sec

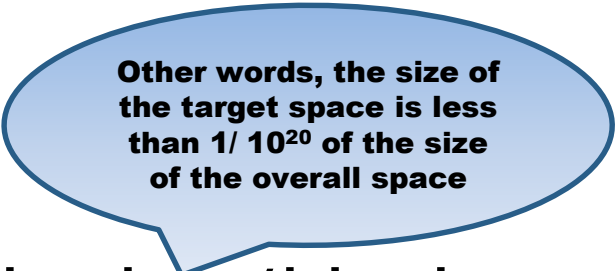
**139,461 years** to find a block today!

# Mining difficulty “target”

(as of March 2015)



# PoW property 1: difficult to compute



**Other words, the size of the target space is less than  $1/10^{20}$  of the size of the overall space**

As of Aug 2014: about  $10^{20}$  hashes/block

Only some nodes bother to compete — miners

## PoW property 2: parameterizable cost

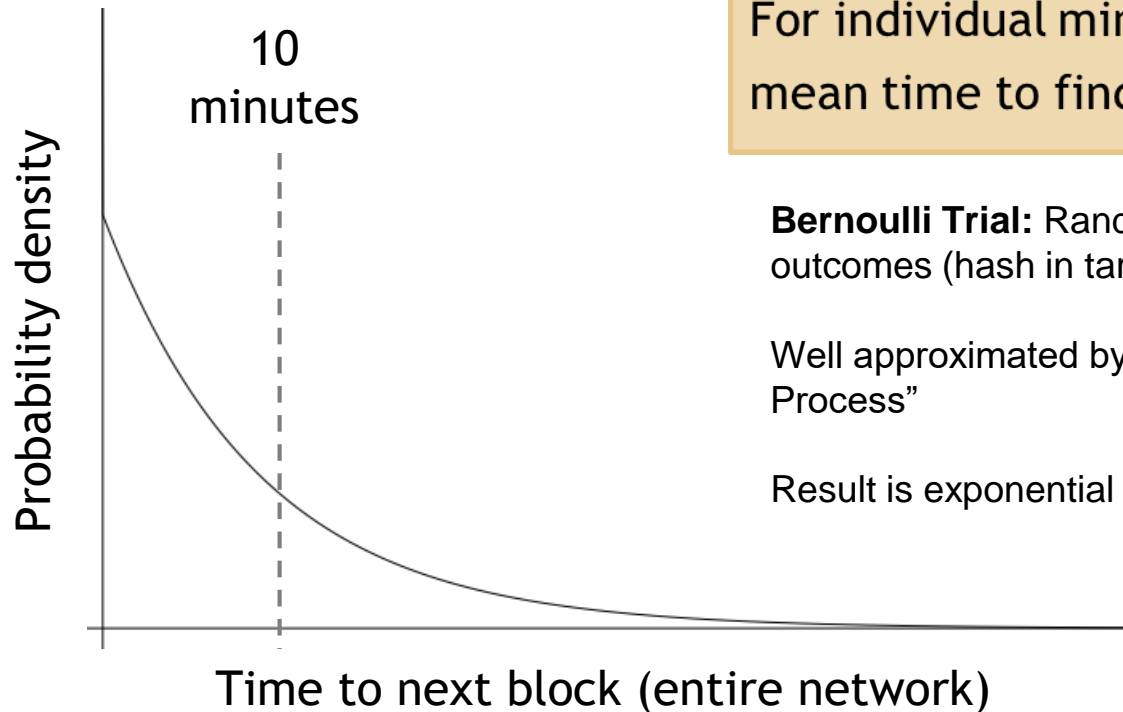
Nodes automatically re-calculate the target every two weeks

Goal: average time between blocks = 10 minutes

Prob (Alice wins next block) =  
fraction of global hash power she controls

Alice with 0.1% of total hash power will find roughly one in every 1000 blocks.

# Solving hash puzzles is probabilistic



For individual miner:

$$\text{mean time to find block} = \frac{10 \text{ minutes}}{\text{fraction of hash power}}$$

**Bernoulli Trial:** Random experiments with two possible outcomes (hash in target space or not) with equal probability.

Well approximated by continuous probability process "Poisson Process"

Result is exponential distribution.

Alice with 0.1% of total hash power will find roughly one in every 1000 minutes.

# PoW property 3: trivial to verify

Nonce must be published as part of block



# Mining Bitcoins in 6 easy steps

1. Join the network, listen for transactions
  - a. Validate all proposed transactions
2. Listen for new blocks, maintain block chain
  - a. When a new block is proposed, validate it
3. Assemble a new valid block
4. Find the nonce to make your block valid
5. Hope everybody accepts your new block
6. Profit!

Useful to  
Bitcoin  
network

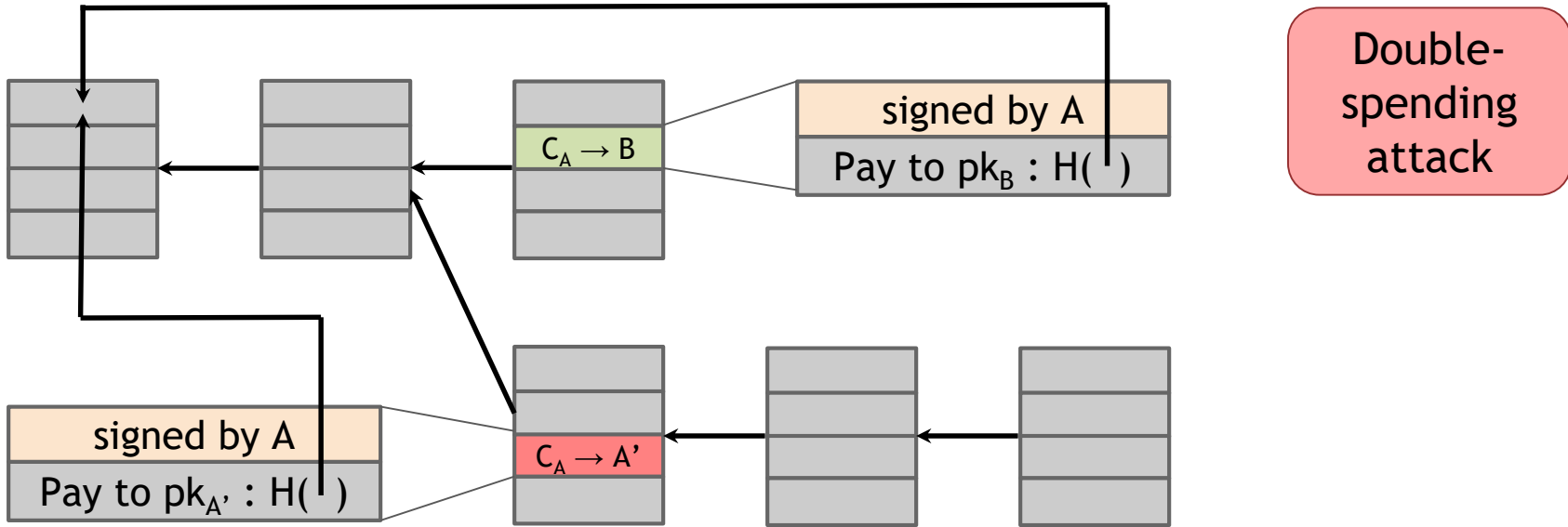


The diagram consists of two green arrows. One arrow starts at the end of step 2 ('a. When a new block is proposed, validate it') and points to the start of step 3 ('Assemble a new valid block'). The second arrow starts at the end of step 1 ('a. Validate all proposed transactions') and also points to the start of step 3.

# Incentives in proof of work

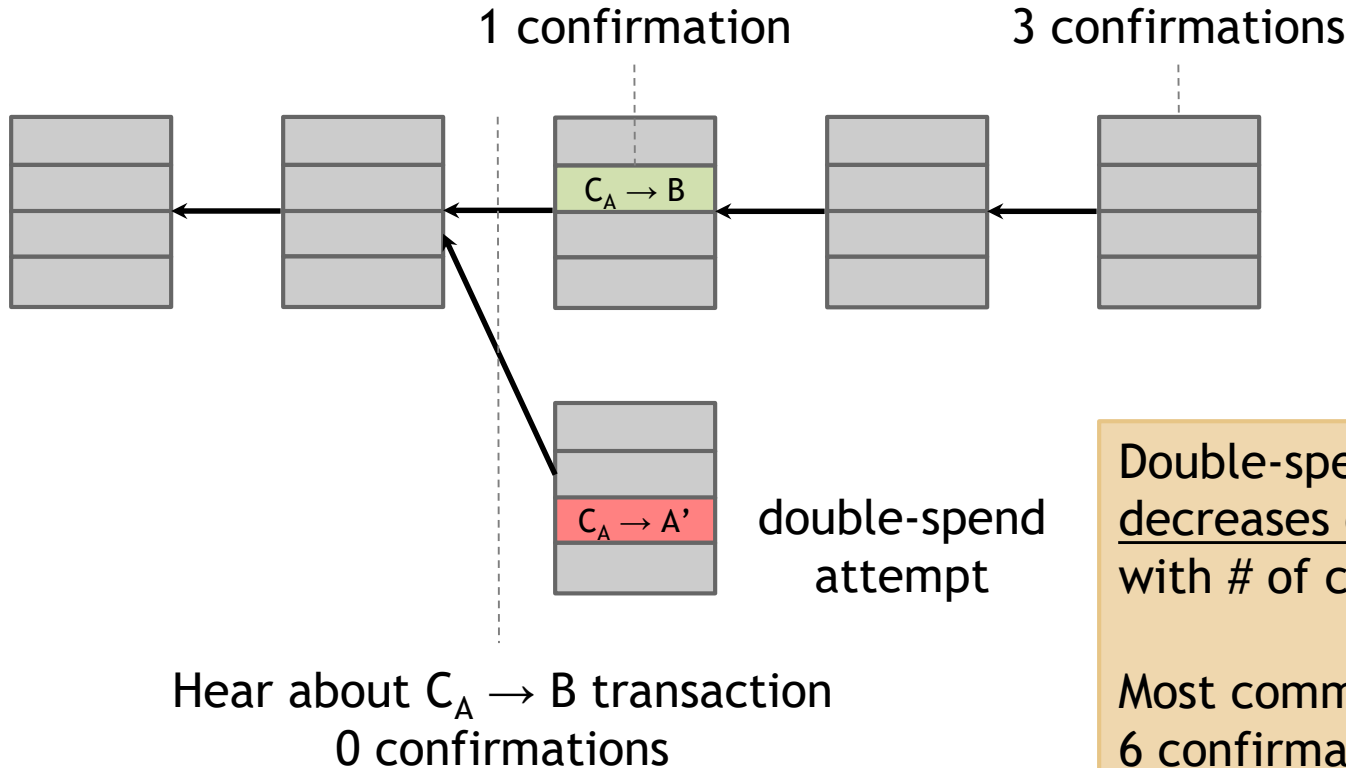


# What can a malicious node do?



Honest nodes will extend the longest valid branch

# From Bob the merchant's point of view



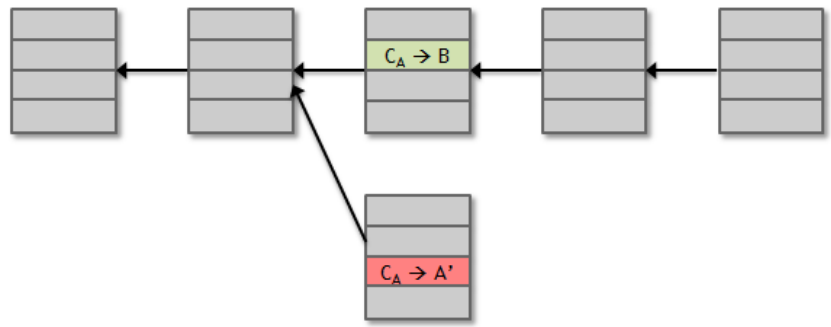
Double-spend probability  
decreases exponentially  
with # of confirmations

Most common heuristic:  
6 confirmations

# Key security assumption

Attacks infeasible if majority of miners  
weighted by hash power follow the protocol

# Security



Protection against invalid transactions is cryptographic, but enforced by consensus

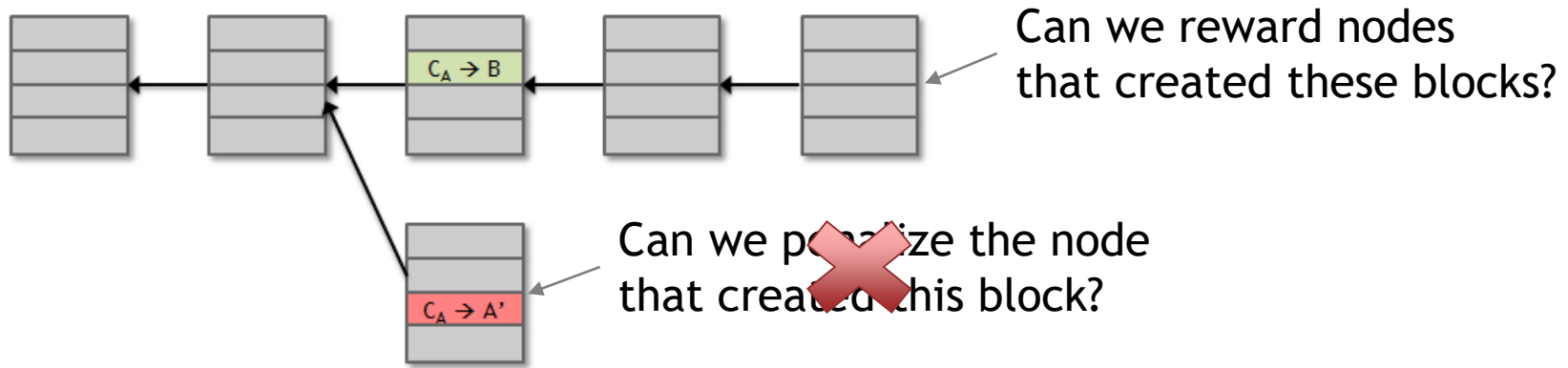
Denial of Service

Protection against double-spending is purely by consensus

You're never 100% sure a transaction is in consensus branch.  
Guarantee is probabilistic

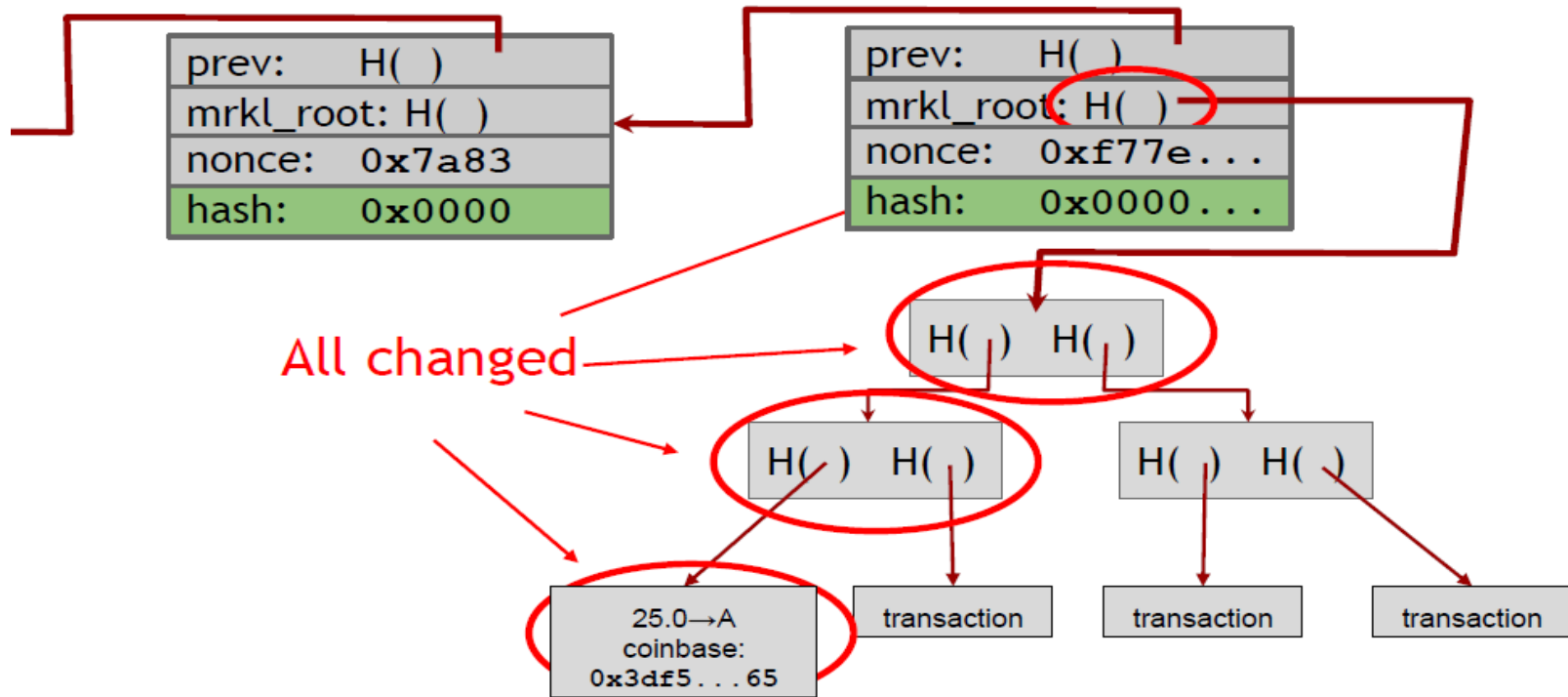
# Assumption of honesty is problematic

Can we give nodes incentives for behaving honestly?



Everything so far is just a distributed consensus protocol  
But now we utilize the fact that the currency has value

# Incentive in Coinbase Transaction



# Incentive 1: block reward

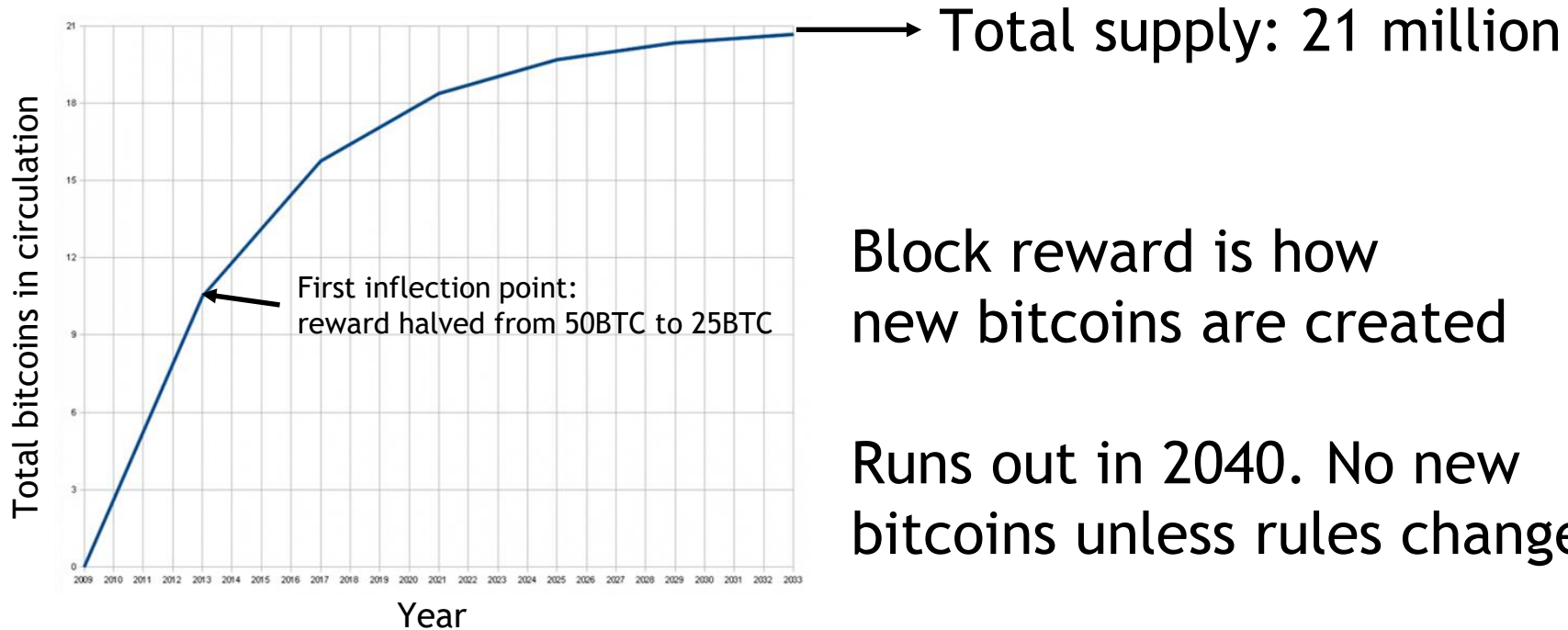
Creator of block gets to

- include special coin-creation transaction in the block
- choose recipient address of this transaction

Value is fixed: currently 12.5 BTC, halves every 4 years

Block creator gets to “collect” the reward only if the block ends up on long-term consensus branch!

# There's a finite supply of bitcoins



Block reward is how  
new bitcoins are created

Runs out in 2040. No new  
bitcoins unless rules change



## Incentive 2: transaction fees

Creator of transaction can choose to make output value less than input value

Remainder is a transaction fee and goes to block creator

Purely voluntary, like a tip

# Transaction

View information about a bitcoin transaction

9cd03f530b83b67eee52bbbd2e9067e79e31513cffb5535c7463d96a8c5d96ae

Transaction ID (TX ID)

1J29P1ceAfJHpG2jPQN1QxdHgCGEnLHd3u

Input Address

34auLDAG8skCooDAPpWFm69JuDz3rYnaDG  
16XafbSNEkkkwshkcusFJS4JxyHs74nudp  
1AW2YoNvhAwatTjUcnzYWPETb3WSonZUD8  
1L5a3gfb8FNJQn2MexVEjSzvXkXCp7mEBU

Output Addresses

0.1 BTC  
0.77 BTC  
0.58 BTC  
2.87094476 BTC

1 Confirmations4.32094476 BTC

Block Information:	
Summary	
Size	292 (bytes)
Weight	1168
Received Time	2018-02-02 07:45:17
Included In Blocks	507234 ( 2018-02-02 08:12:38 + 27 minutes )
Confirmations	1 Confirmations
Visualize	<a href="#">View Tree Chart</a>

Transaction information:	
Inputs and Outputs	
Total Input	4.32123876 BTC
Total Output	4.32094476 BTC
Fees	0.000294 BTC
Fee per byte	100.685 sat/B
Fee per weight unit	25.171 sat/WU
Estimated BTC Transacted	0.1 BTC
Scripts	<a href="#">Show scripts &amp; coinbase</a>

Recall:

transaction fee = value of inputs - value of outputs  
fee goes to miner who records the transaction

Costs resources for

peers to relay your transaction

miner to record your transaction

Transaction fee compensates for (some of) these costs

How are transaction fees set today?

No fee if

tx less than 1000 bytes in size,  
all outputs are 0.01 BTC or larger, and  
priority is large enough

Priority = (sum of inputAge\*inputValue) / (trans size)

Otherwise fee is 0.0001 BTC per 1000 bytes

Approx transaction size:  $148 N_{\text{inputs}} + 34 N_{\text{outputs}} + 10$

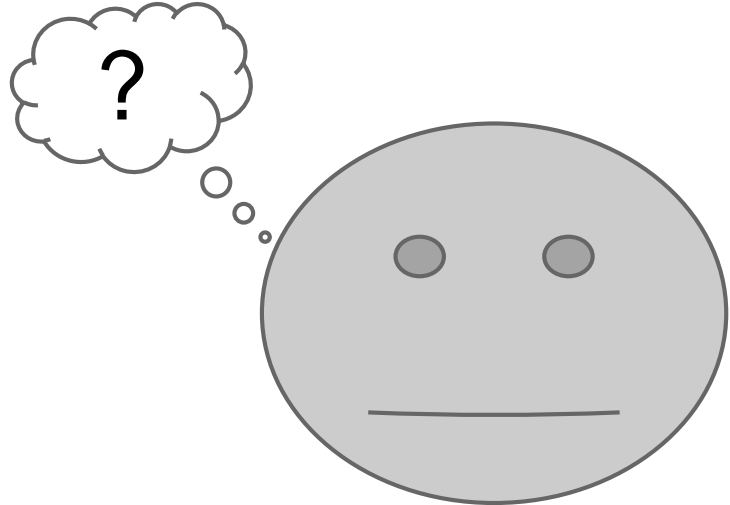
# A second look at transaction fees

Default policy:

```
priority = sum(input_value * input_age) / size_in_bytes
```

Accept without fees if:

```
priority > 0.576
```



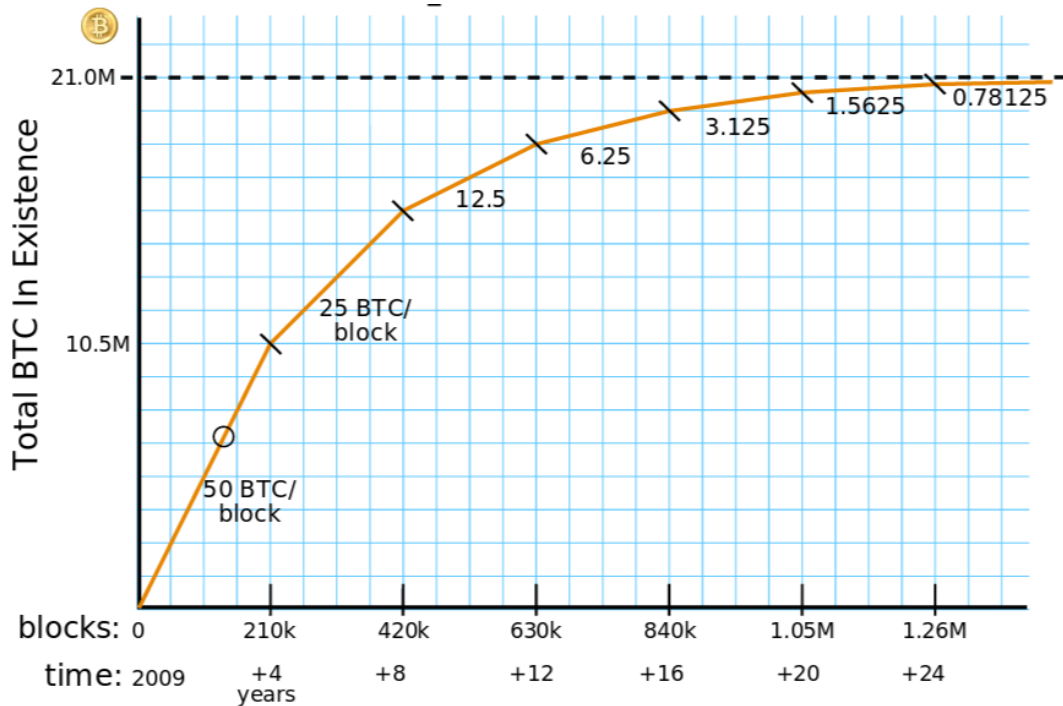
Generally, higher fee means transaction will be forwarded and recorded faster.

If you don't pay the consensus fee, your transaction will take longer to be recorded.

Miners prioritize transactions based on fees and the priority formula.

# Transaction fees will matter more

Currently, block rewards are > 99% of miner revenue. But:



Eventually,  
transaction fees  
will dominate

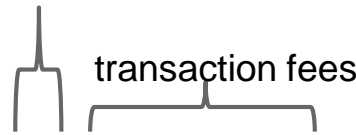
Courtesy:  
Brian Warner

# The real deal: coinbase transaction

```
"in":[
  {
    "prev_out":{
      "hash":"000000.....0000000",
      "n":4294967295
    },
    "coinbase":"..."
  },
  "out":[
    {
      "value":"25.03371419",
      "scriptPubKey":"OPDUP OPHASH160 ... "
    }
  ]
}
```

block reward

transaction fees





# Mining economics

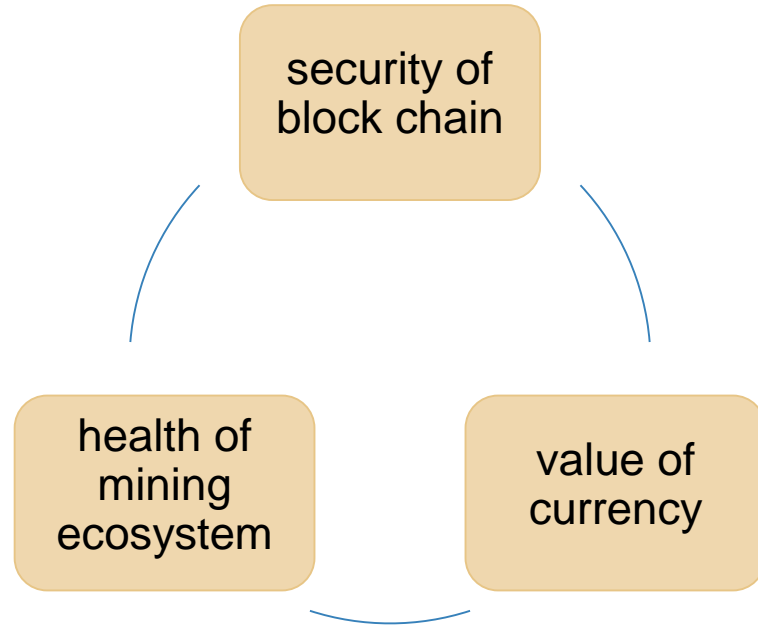
If mining reward  
(block reward + Tx fees) > hardware +  
electricity cost → Profit

## Complications:

- fixed vs. variable costs
- reward depends on global hash rate

Putting it all together

# Bitcoin is bootstrapped



# What can a “51% attacker” do?

Steal coins from existing address? X

Suppress some transactions?

- From the block chain ✓
- From the P2P network X

Change the block reward? X

Destroy confidence in Bitcoin? ✓✓

# How big is the network?

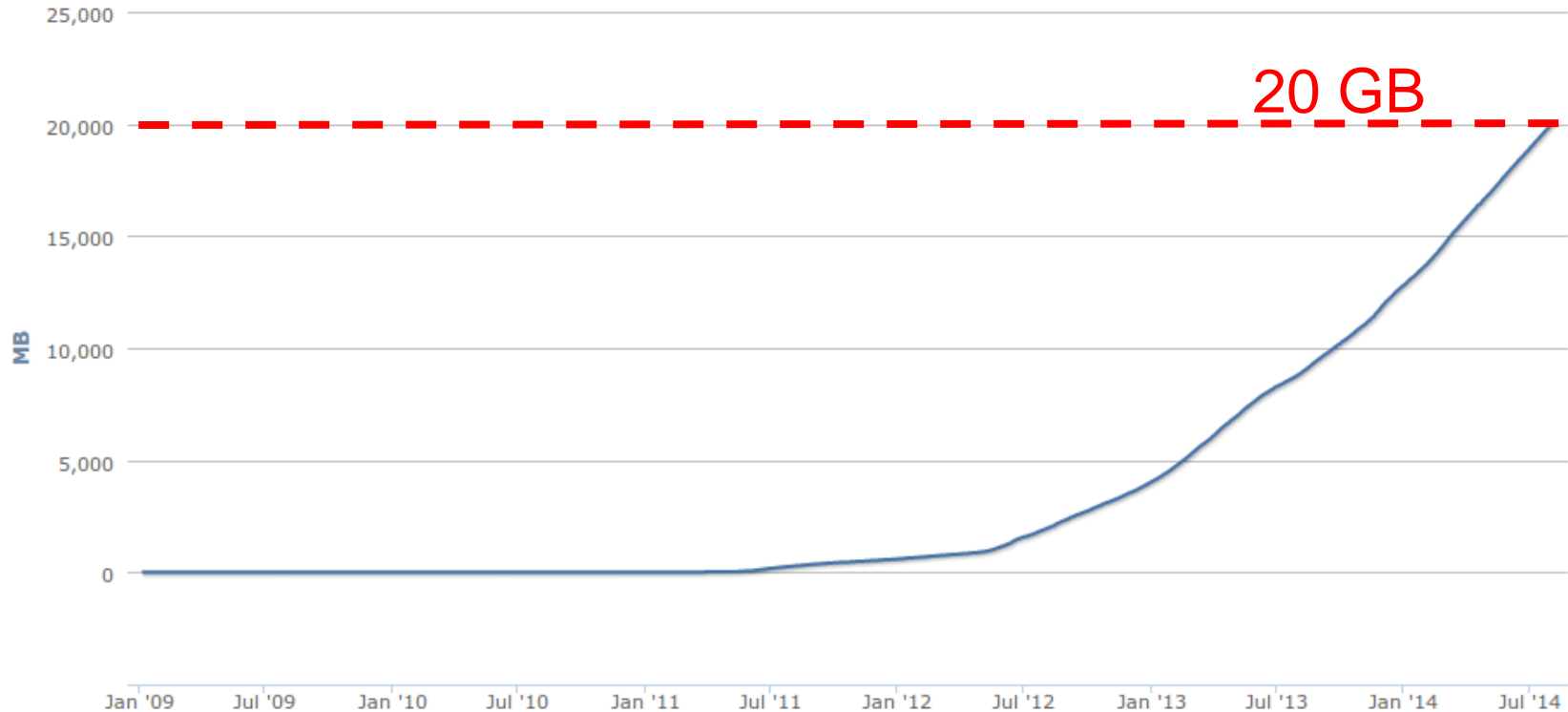
- Impossible to measure exactly
- Estimates-up to 1M IP addresses/month
- Only about 5-10k “full nodes”
  - Permanently connected
  - Fully-validate
- This number may be dropping!

# Fully-validating nodes

- Permanently connected
- Store entire block chain
- Hear and forward every node/transaction

# Storage costs (Size of Blockchain)

Blockchain Size  
Source: blockchain.info



# Tracking the UTXO set

- **U**nspent **T**ransaction **O**utput
  - Everything else can be stored on disk
- Currently ~12 M UTXOs
  - Out of 44 M transactions
- Can easily fit into RAM



# Thin/SPV clients (not fully-validating)

Idea: don't store everything

- Store block headers only
- Request transactions as needed
  - To verify incoming payment
- Trust fully-validating nodes

1000x cost savings! (20 GB -> 23MB)

# Software diversity

- About 90% of nodes run “Core Bitcoin” (C++)
  - Some are out of date versions
- Other implementations running successfully
  - BitcoinJ (Java)
  - Libbitcoin (C++)
  - btcd (Go)
- “Original Satoshi client”

## Limitations & improvements

# Hard-coded limits in Bitcoin

- 10 min. average creation time per block
- 1 M bytes in a block
- 20,000 signature operations per block
- 100 M *satoshis* per bitcoin
- 21M total bitcoins maximum
- 50,25,12.5... bitcoin mining reward

} These affect  
economic  
balance of  
power too much  
to change now

# Throughput limits in Bitcoin

- 1 M bytes/block (10 min)
- >250 bytes/transaction
- 7 transactions/sec 😞

Compare to:

- VISA: 2,000-10,000 transactions/sec
- PayPal: 50-100 transaction/sec

# Cryptographic limits in Bitcoin

- Only 1 signature algorithm (ECDSA/P256)
- Hard-coded hash functions (SHA-256, RIPEMD-160)

Crypto primitives might break by 2040...