How Bitcoin Works

George Vlahavas

November 11, 2023

What is Bitcoin?

Bitcoin is a decentralized digital currency. It operates on a peer-to-peer network that enables instant and secure transactions between users, without the need for intermediaries like banks or governments.

The Software Side

- Bitcoin is Free Software (MIT license)
- `https://bitcoincore.org`
- `https://github.com/bitcoin/bitcoin`

Software Details

- Bitcoin Core latest release: 25.1
- About 682000 lines of code
- Mostly written in C++
- 926 contributors so far
- Many more contribute research, peer review, testing...
- About 40000 commits
- 343 open issues (7316 closed)
- 291 pull requests (19593 closed)

History of Bitcoin

- **2008**: Whitepaper Publication by Satoshi Nakamoto
- **2009**: Genesis Block and Launch
- **Early Years**...
- **2010**: First Known Commercial Bitcoin Transaction
- **2011-2013**: Growth and Volatility
- **2013**: Price Surges and Regulatory Interest
- **2014-2016**: Maturing and Development
- **2017**: Price Boom and Mainstream Attention
- **2018-2021**: Market Corrections and Development
- **2022-Today**: New Growth

Bitcoin Main Attributes

- Decentralized
- Immutable
- Transparent
- Open
- Secure

How is it used?

- Monetary transactions (cross-border too)
- Investment and Store of Value
- Bank Services for the Unbanked
- Fundraising, Crowdfunding, Donations
- ...

Basics: Hashing

Basics: Data

- What is data?
  - Numbers
  - Text
  - Some combination of the two
- Text can be encoded as numbers (e.g. A=65, B=66, C=67...)
- So ultimately, everything is a number!

Basics: Number Systems

Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Basics: Numbers in Hex

Numbers in hex take less space... Examples:

Decimal	Hex
660475	A13FB
12965487	C5D66F

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71
- Sum them up:

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71
- Sum them up:
  - 71 + 114 + 101 + 101 + 107 + 76 + 85 + 71 = 726

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71
- Sum them up:
  - 71 + 114 + 101 + 101 + 107 + 76 + 85 + 71 = 726
- Let's give it a maximum size of 256 (8 bits)

Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71
- Sum them up:
  - 71 + 114 + 101 + 101 + 107 + 76 + 85 + 71 = 726
- Let's give it a maximum size of 256 (8 bits)
  - 726 % 256 = 214

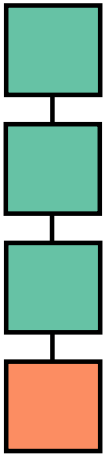
Basics: A Simple Hash Function

- Let's find a simple way to hash the string "GreekLUG"
- Assign each letter to a number (ASCII):
  - G: 71
  - r: 114
  - e: 101
  - e: 101
  - k: 107
  - L: 76
  - U: 85
  - G: 71
- Sum them up:
  - 71 + 114 + 101 + 101 + 107 + 76 + 85 + 71 = 726
- Let's give it a maximum size of 256 (8 bits)
  - 726 % 256 = 214
- "214", or in hex "d6" can be considered a digital signature for "GreekLUG"

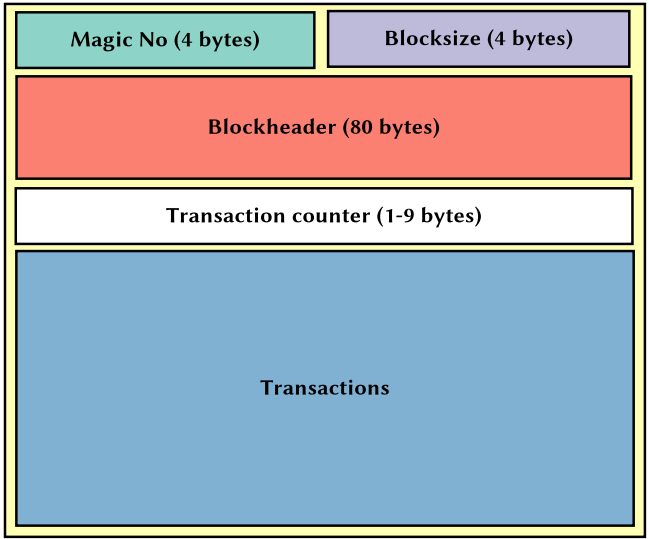
Basics: SHA256, A Much Better Hash Function

- Size: 256 bits (about 10^{77})
- Quick to calculate
- One way function
- Same input always results to same hash
- Any slight change in the input changes the output unpredictably
- Examples:
  - sha256("I have 2 apples") = `40a81c7a9d540081c7da5b5934c033a589a95657c13fd6eb99e286a3bfd0683c`
  - sha256("I have 3 apples") = `74d2c0de50110580f0e25fc22cf901d9ebda2006e1f0fd9c0216e79433c12c61`

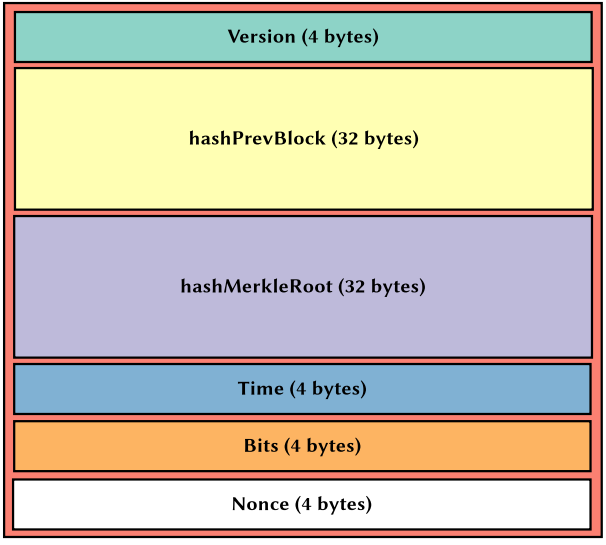
The Blockchain



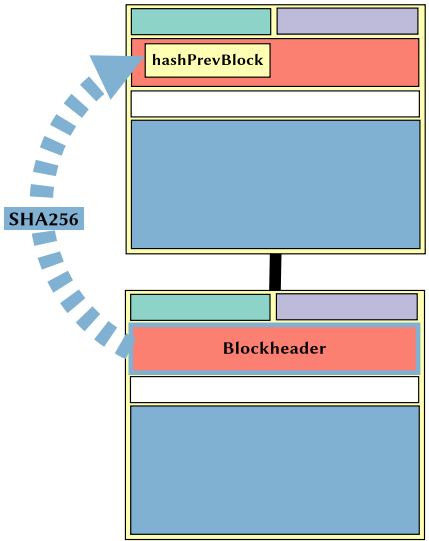
What's in a block?



What's in a block header?



Hashing the previous block



Transaction immutability

- The hash of the previous block is calculated only using the previous block's header
- But then how is it ensured that transactions in previous blocks won't be altered?

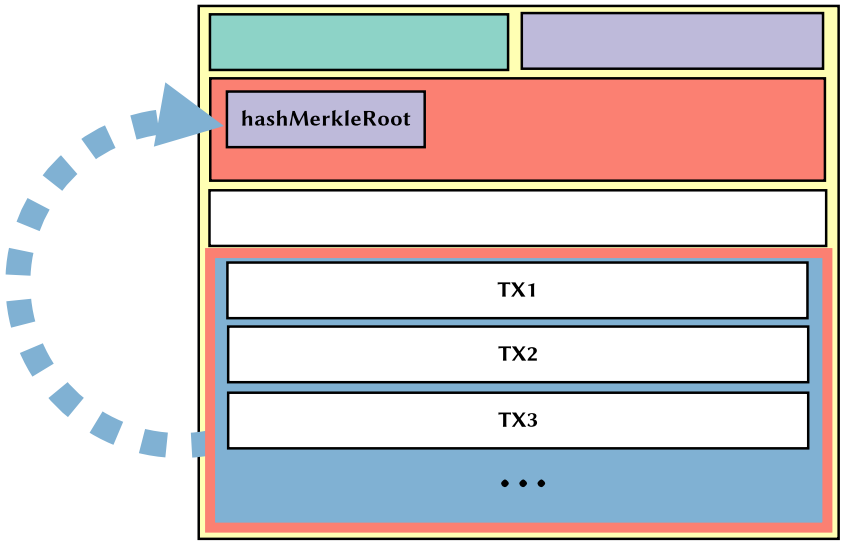
Transaction immutability

- The hash of the previous block is calculated only using the previous block's header
- But then how is it ensured that transactions in previous blocks won't be altered?
  - *The transactions themselves are hashed and their hash is stored in the current block's Merkle Root*

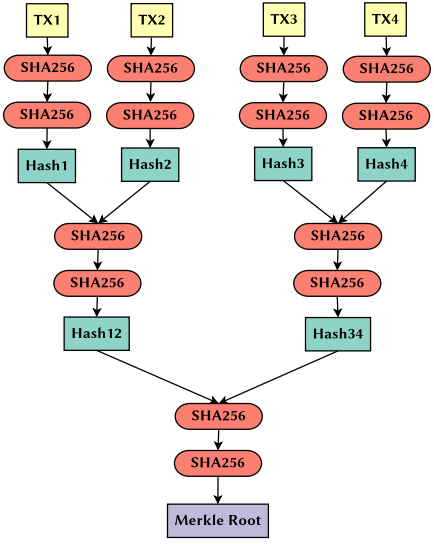
Transaction immutability

- So, if a transaction is altered, the Merkle Root would be changed.
- Since the Merkle Root is included in the block header, if the Merkle Root is changed, the block header contents would be changed.
- If the block header contents are changed the block header hash would be changed.

Calculating the Merkle Root

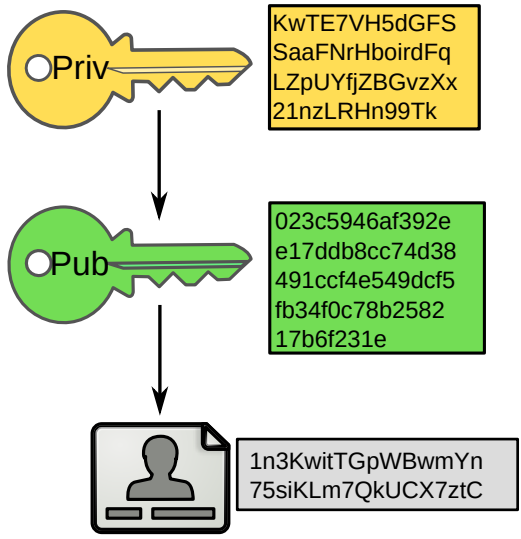


The Merkle Tree



Addresses: Overview

- Each user owns a pair of private/public keys
- A private key is just a very large random number (256 bits), base58 encoded
- The public key (also just a number) is calculated from the private key using Elliptic Curve Cryptography
  - One way function
  - Size 65 bytes (uncompressed), 33 bytes (compressed)
- An address (also just a number) is calculated from a public key (SHA256 and RIPEMD160 algorithms)


Priv
KwTE7VH5dGFS
SaaFNrHboirdFq
LZpUYfjZBGvzXx
21nzLRHn99Tk
Pub
023c5946af392e
e17ddb8cc74d38
491ccf4e549dcf5
fb34f0c78b2582
17b6f231e
1n3KwitTGpWBwmYn
75siKLm7QkUCX7ztC

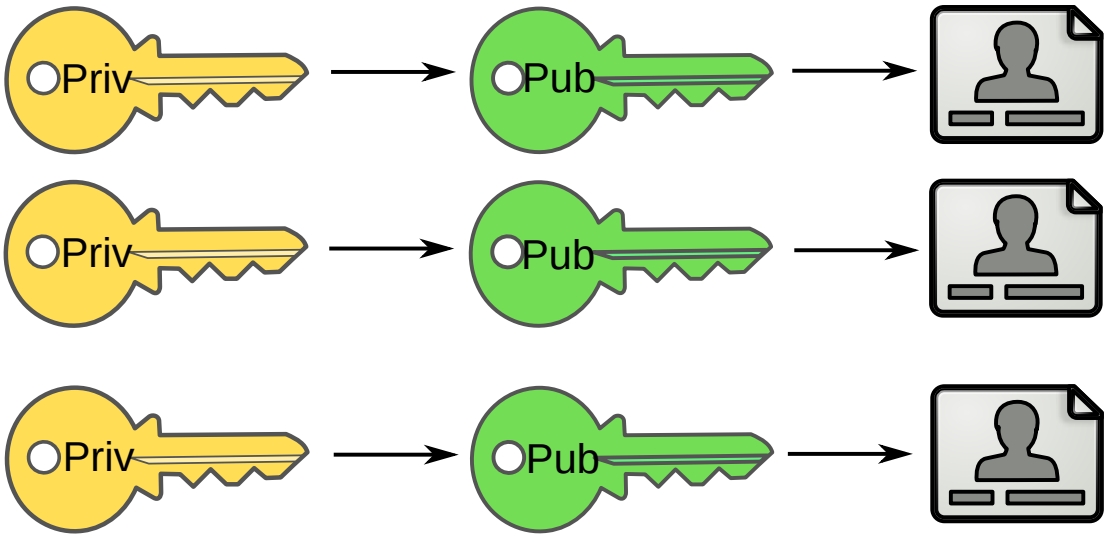
Keys and Addresses

- A private key should never be shared
- A public key may be shared but it's better to use an address for that
- An address represents the owner of a private/public pair
- An address is shared to receive bitcoins
- Ownership of the private key gives access to the respective address
- An address may represent complex scripts (P2SH)

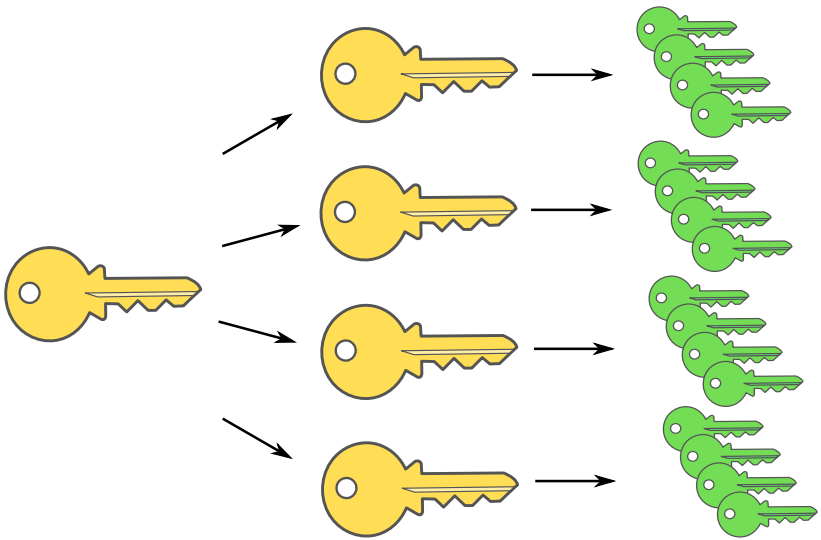
Bitcoin Wallets

- Bitcoin wallets do not hold Bitcoins. They hold private keys
- Allow management of keys and addresses
- Multiple keys can be managed by a single wallet
- A key is usually only used once
- Types of wallets:
  - Non-deterministic: Multiple private keys are pregenerated
  - Deterministic: One master key. All other keys are derived from that
- Can be used to send bitcoins, check balances (receive bitcoins?)

Non-deterministic Wallets



Deterministic Wallets



Addresses (mainnet)

- P2PKH Addresses: 34 characters in Base58 encoding. Start with 1.
- P2SH Addresses: 34 characters in Base58 encoding. Start with 3.
- SegWit Addresses: 42 characters in Bech32 encoding. Start with bc1.

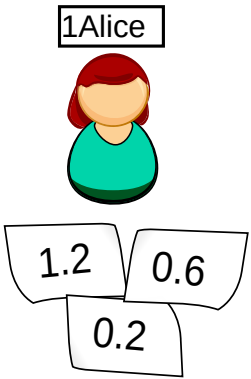
Transactions

- Transactions specify how bitcoins are transferred
  - 1Alice sends 1.3 BTC to 1Bob
- Alice has to prove that she owns that 1.3 BTC
- Bob doesn't need to do anything

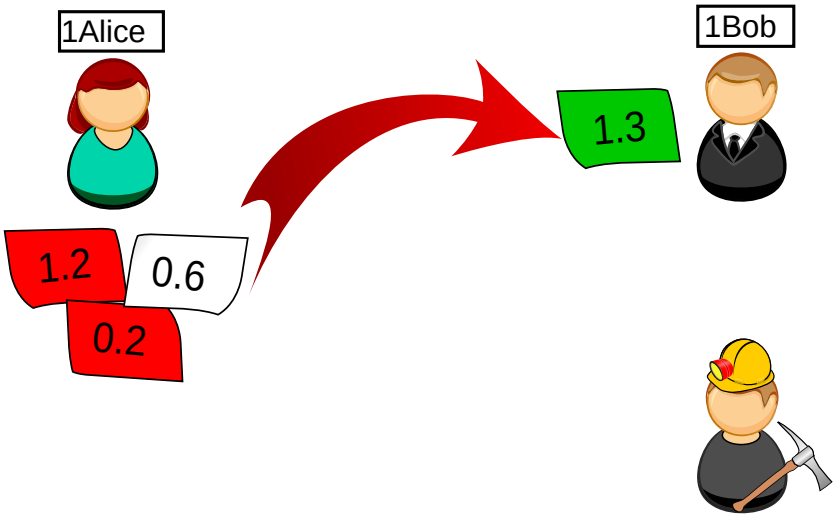
UTXOs

- Unspent Transaction Outputs
- Consider them as the Bitcoin notes
- Can hold any value of BTC
- They are created when received
- They are destroyed when they are spent

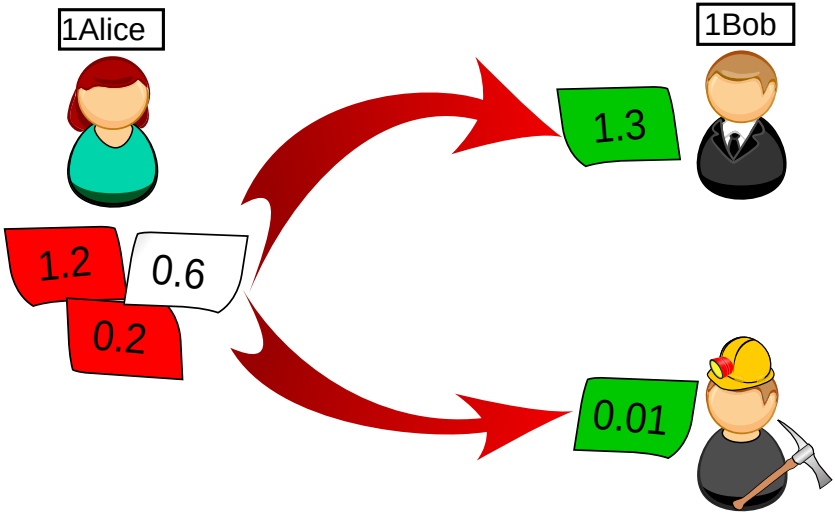
Transaction Example (1)



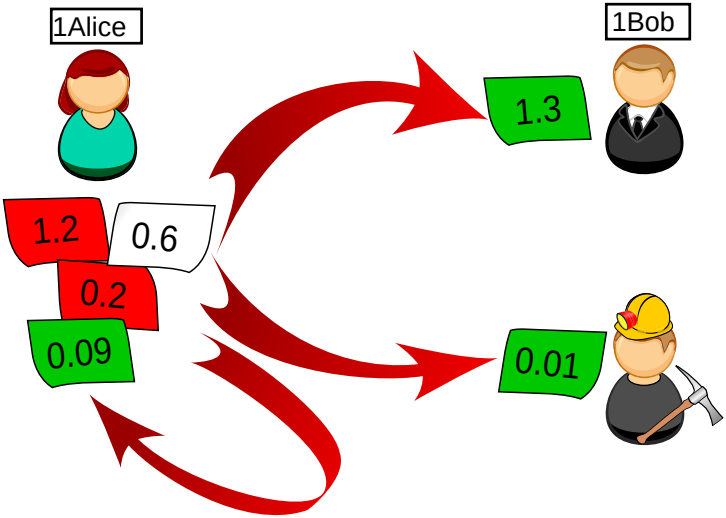
Transaction Example (2)



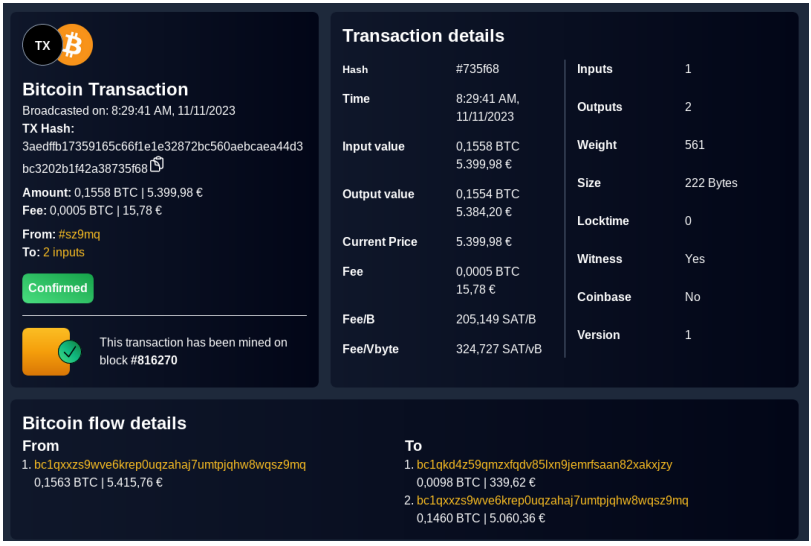
Transaction Example (3)



Transaction Example (4)



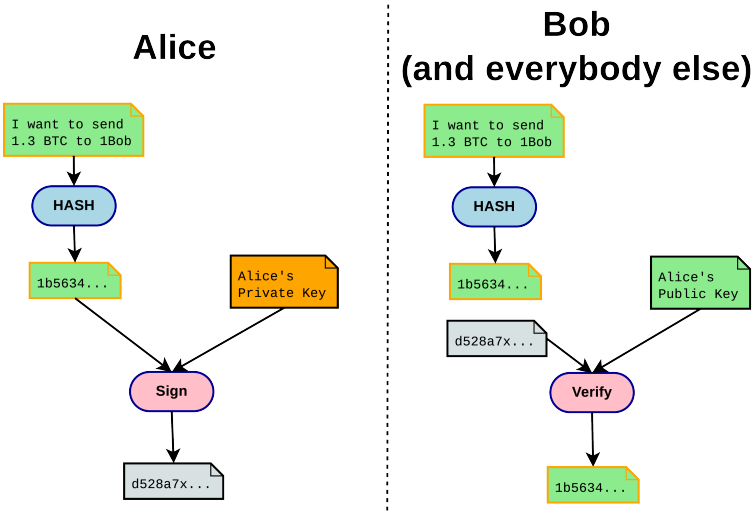
A Real Transaction



Proving Ownership

- You prove that you actually own the BTC you want to transfer using digital signatures

Digital Signatures



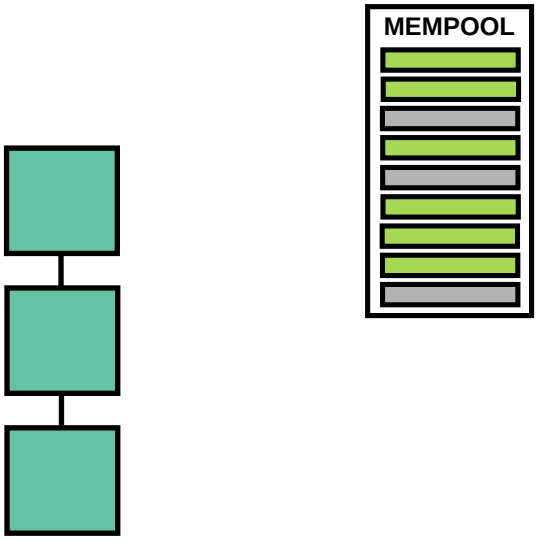
Sending a Transaction

- Every time someone wants to make a transaction, they sign it and broadcast it to all the other users in the network
- The transaction is stored in a data structure called the **Mempool**
- Miners pick transactions from the mempool to create a new block

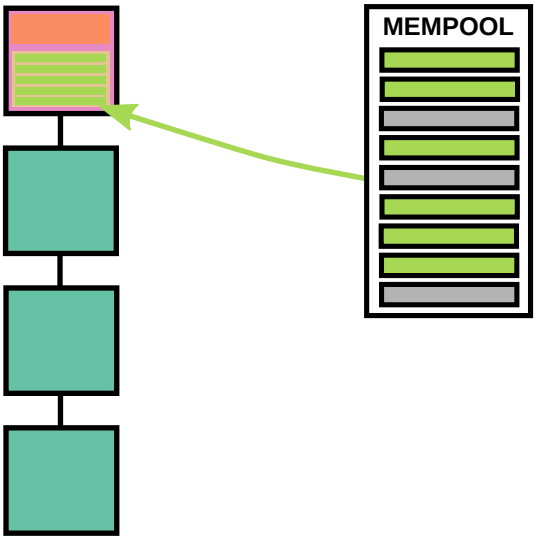
Mining

- The process of creating new blocks and adding new transactions
- Needs a lot of computing power, energy intensive process
- The only way new Bitcoins are created

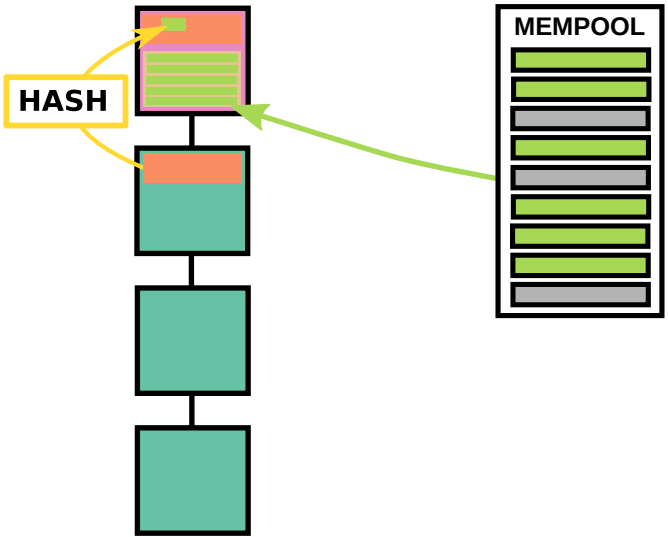
Including Transactions in a New Block (1)



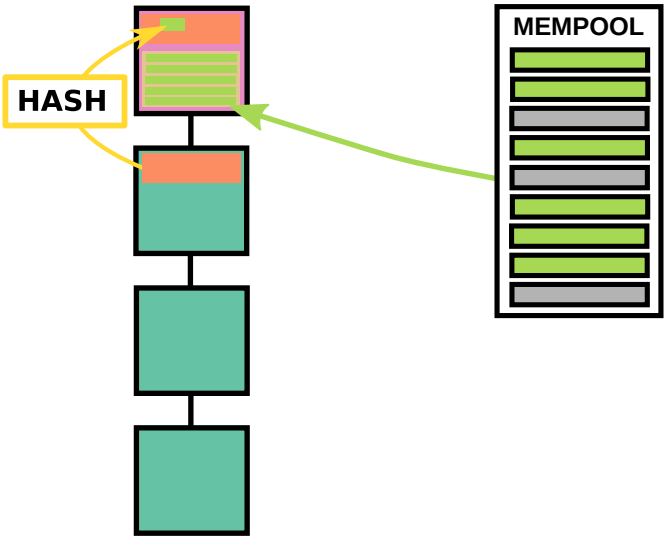
Including Transactions in a New Block (2)



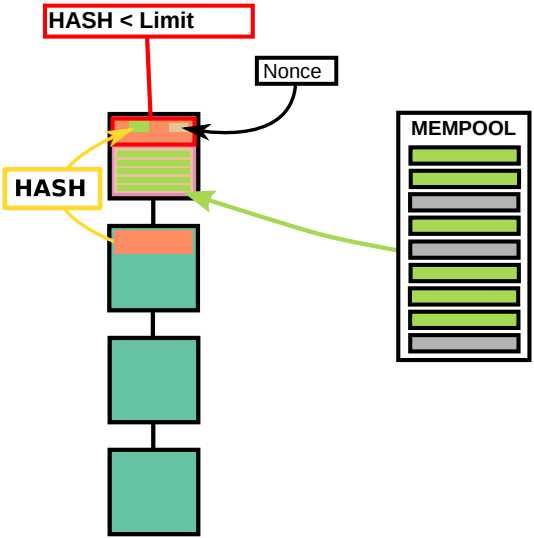
Including Transactions in a New Block (3)



Including Transactions in a New Block (4)



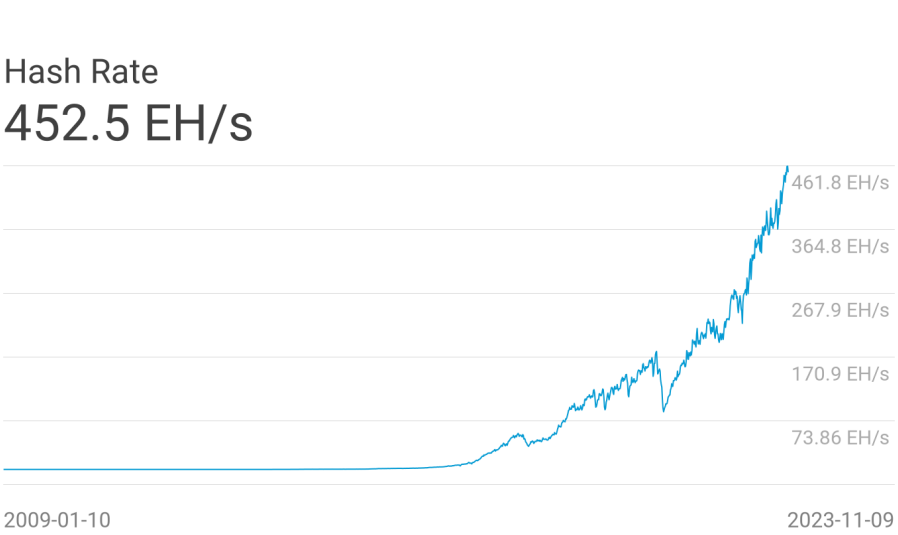
Including Transactions in a New Block (5)



Mining Difficulty

- The miner needs to create a block with a hash value smaller than a *Limit* value
- That is what is called the *Proof-of-Work (PoW)* algorithm
  - Extremely difficult to calculate
  - Extremely easy to validate
- The *Limit* value is determined in the previous block in the *Bits* section
- The miner can try to find a right hash by changing the nonce, rearranging the transactions, picking other transaction, changing the timestamp...
- Current hash rate is about 450000000 TH/s (1TH = 10^{12} hashes)
- Mining difficulty is adjusted every 2016 blocks (about two weeks)
- Mining difficulty is adjusted so that, on average, a new block is mined in about 10 minutes

Hash Rate Evolution



Why Miners Mine

- Miners earn a block reward with every block they mine
- They also receive fees from every transaction they include
- The block reward was originally 50 BTC
- Algorithmically set to halve every 210000 blocks (about four years)
- Currently set to 6.25 BTC (about 225000€)

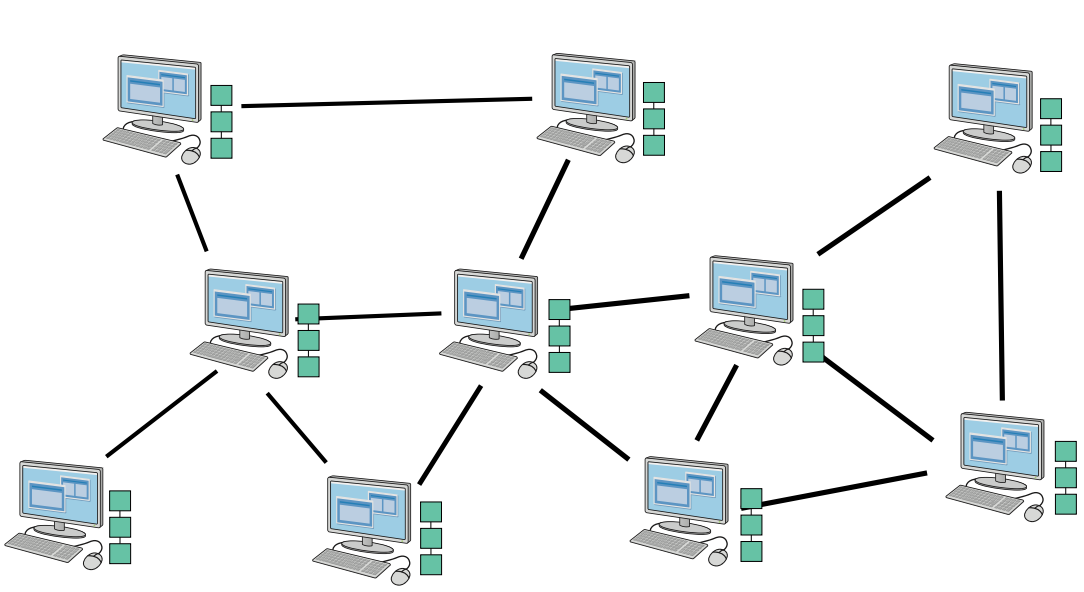
What Do Miners Use for Mining?

- These days mining is only possible with ASICs (Application Specific Integrated Circuits)
- Cost thousands of euros
- Capable of up to 230 TH/s
- Not cost effective to run on a home setup (unless you have free electricity?)

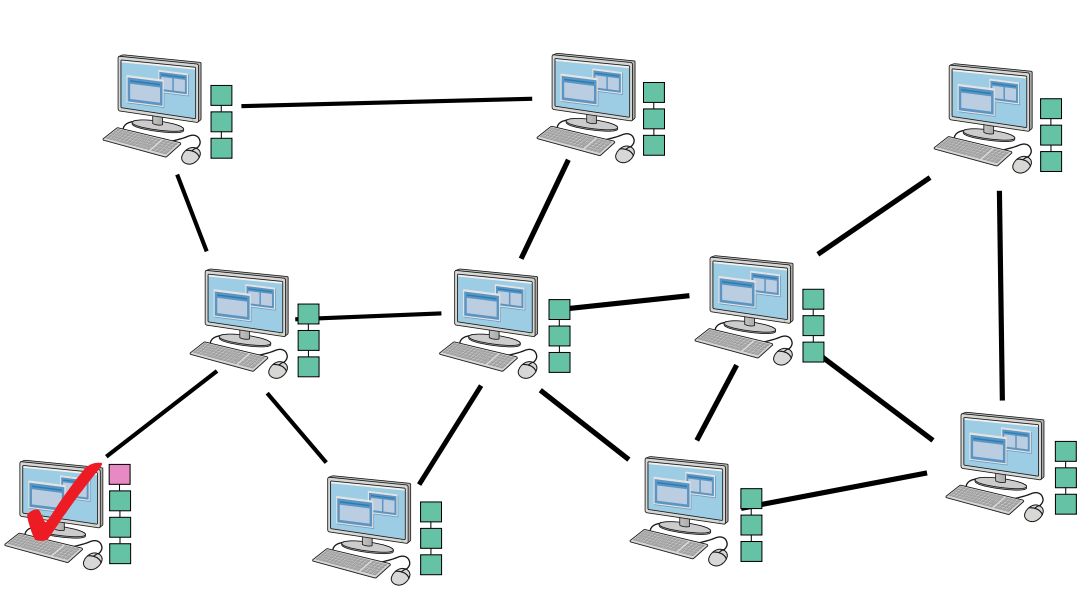
What do we get from PoW?

- Consensus mechanism
- Security against attacks
- Decentralization
- Trustlessness

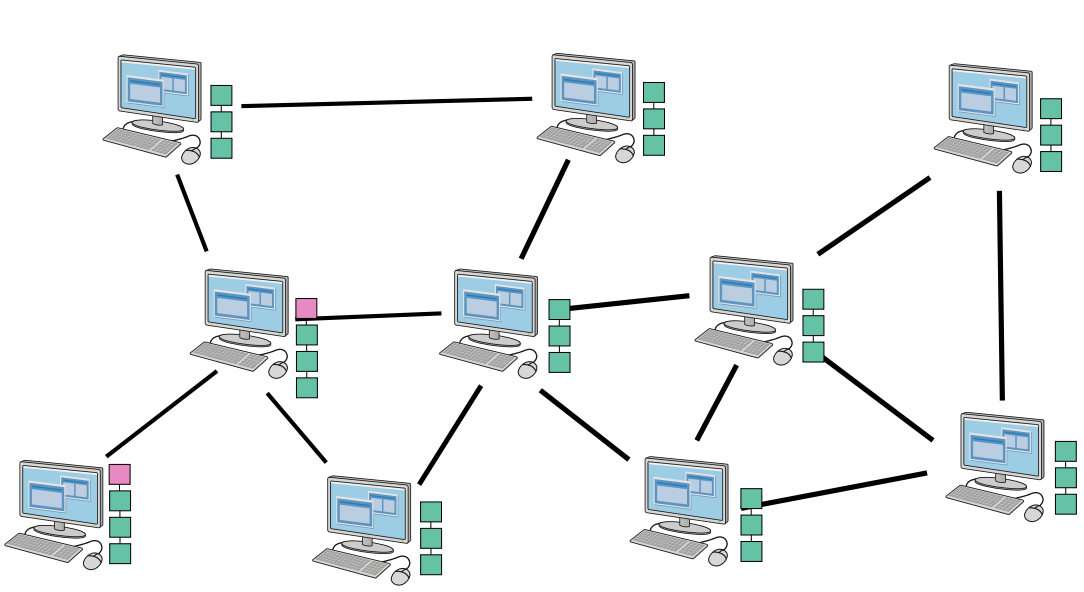
How Blocks Propagate (1)



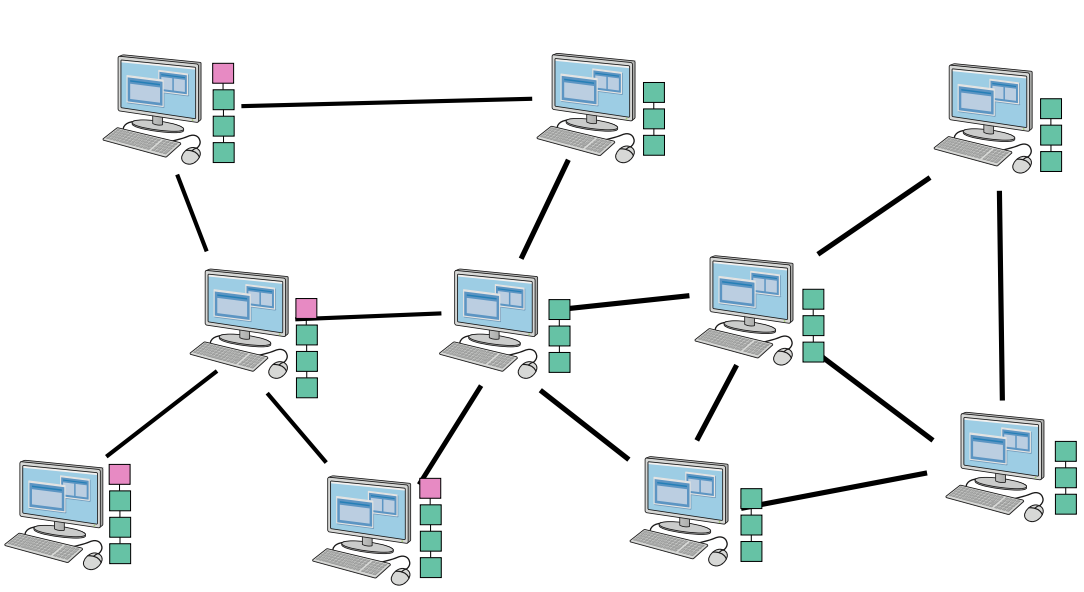
How Blocks Propagate (2)



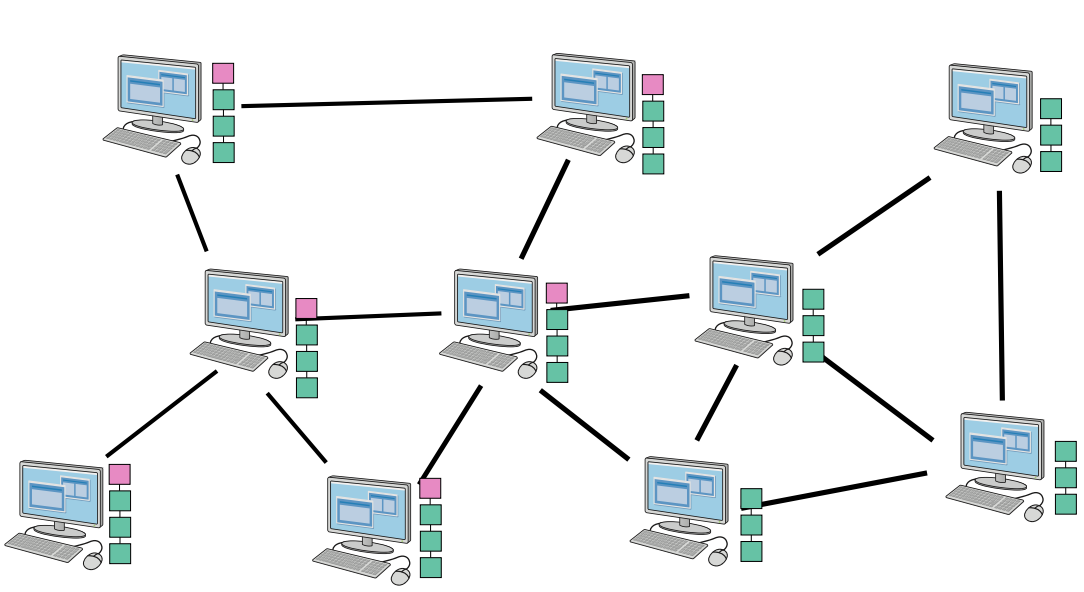
How Blocks Propagate (3)



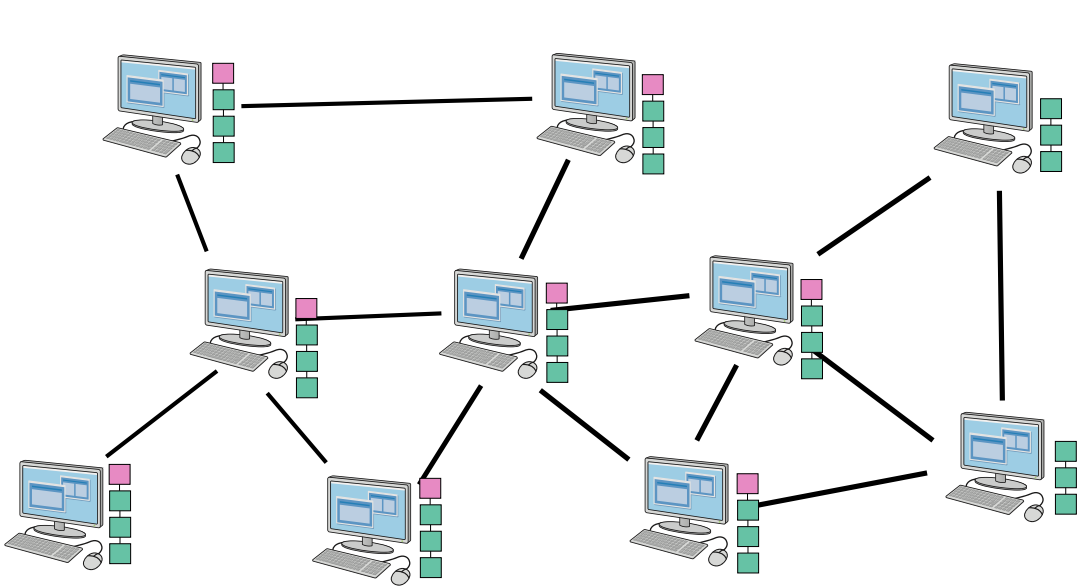
How Blocks Propagate (4)



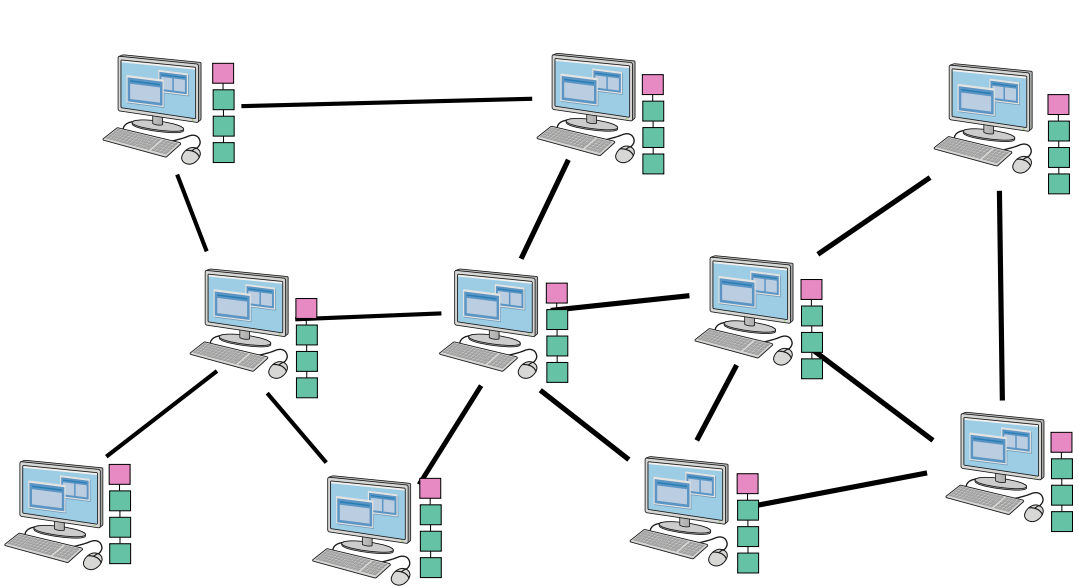
How Blocks Propagate (5)



How Blocks Propagate (6)



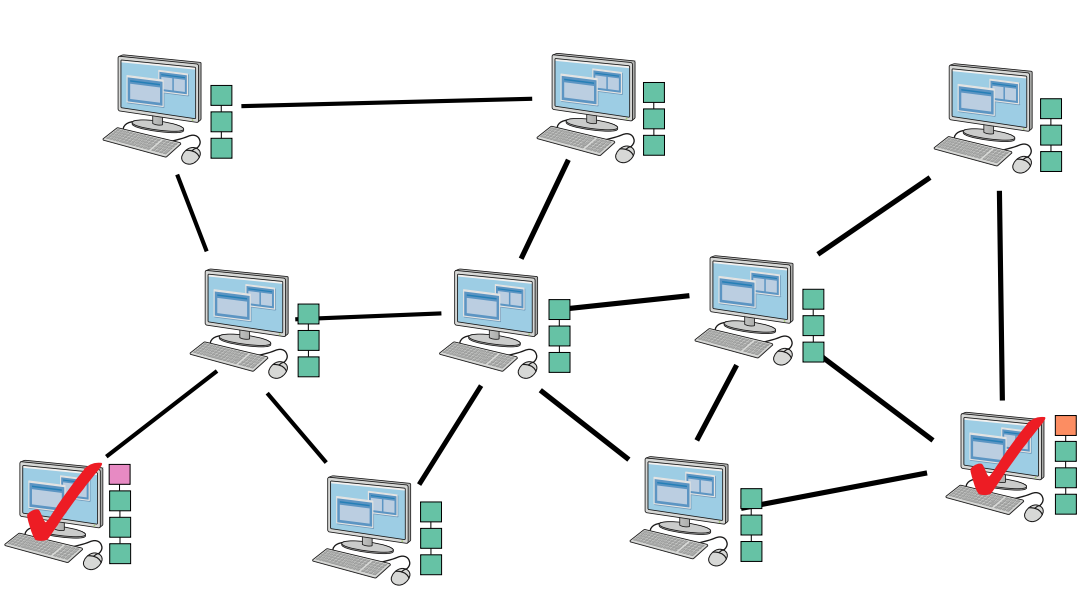
How Blocks Propagate (7)



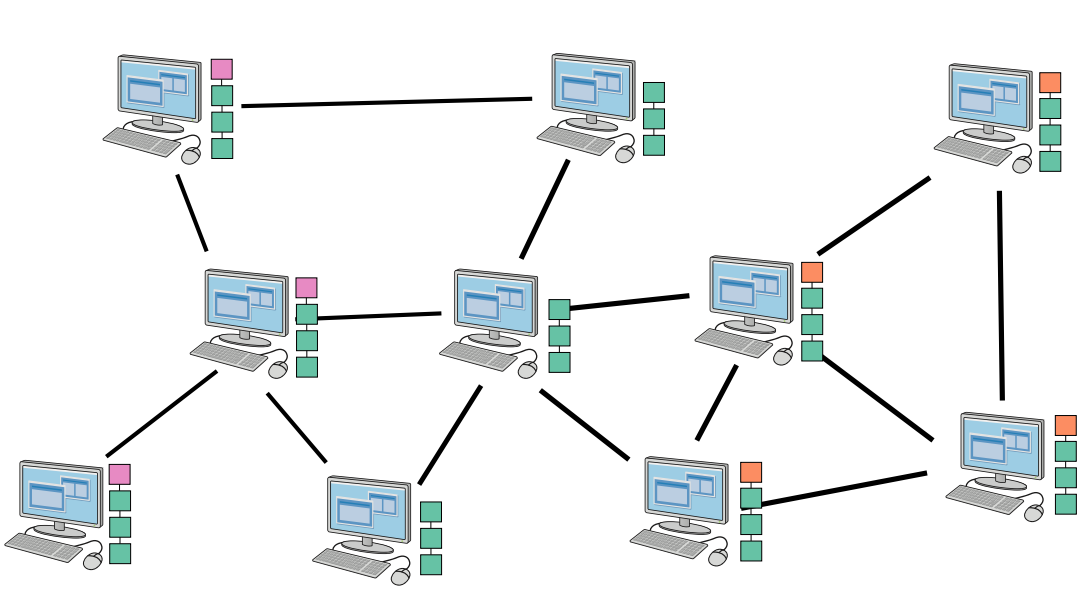
Multiple Miners

- There is not only a single solution to the mining problem
- More than one miners may mine new block at the same time
- The network is divided
- But eventually will get back into balance, probably by the next block
  - The longer chain wins (actually the one with the more work in it)

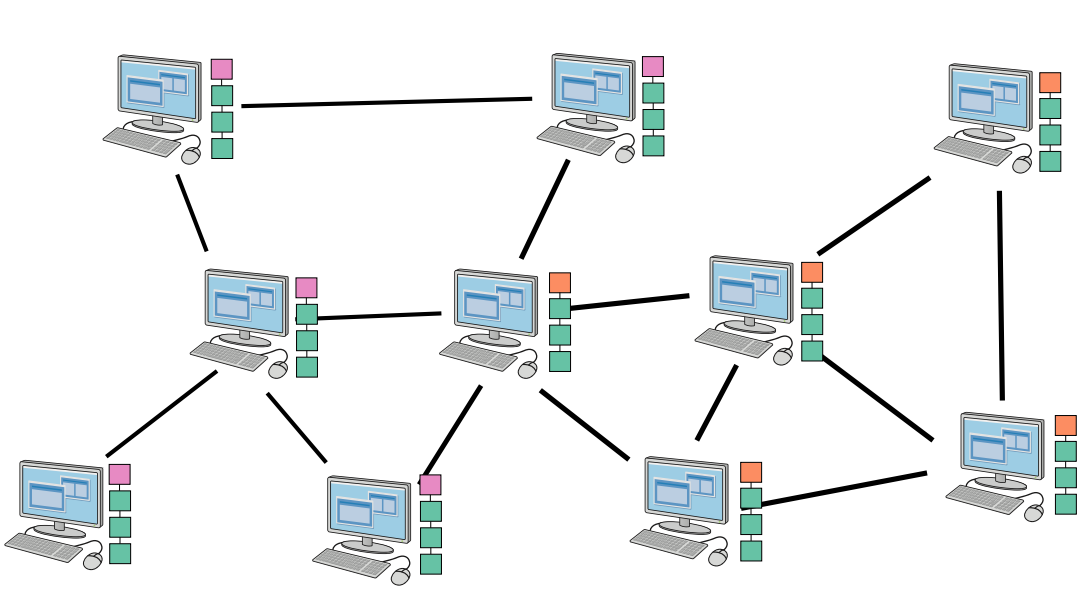
How Blocks Propagate (8)



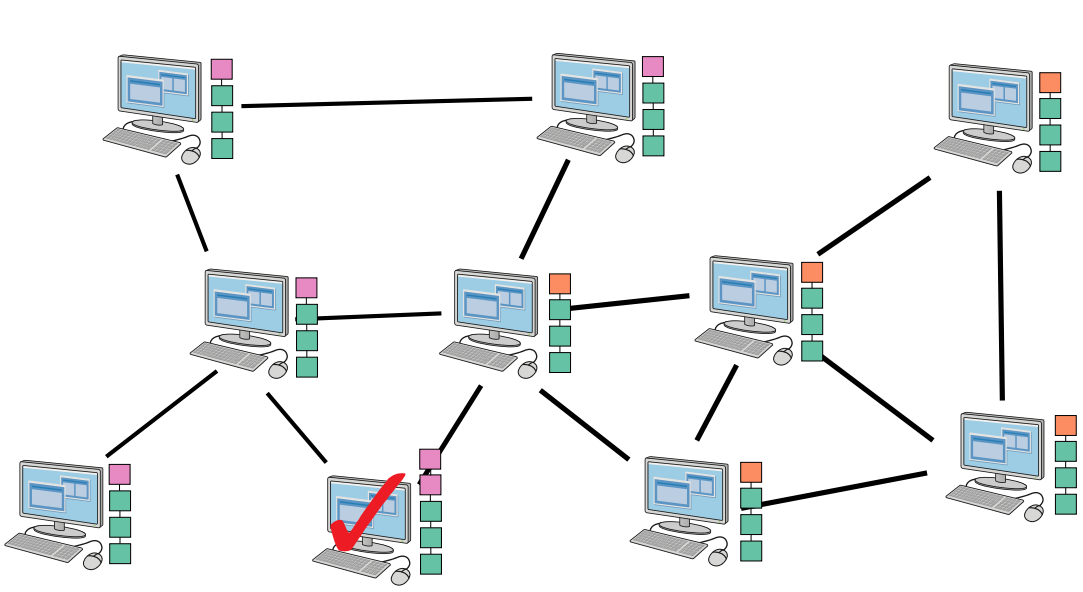
How Blocks Propagate (9)



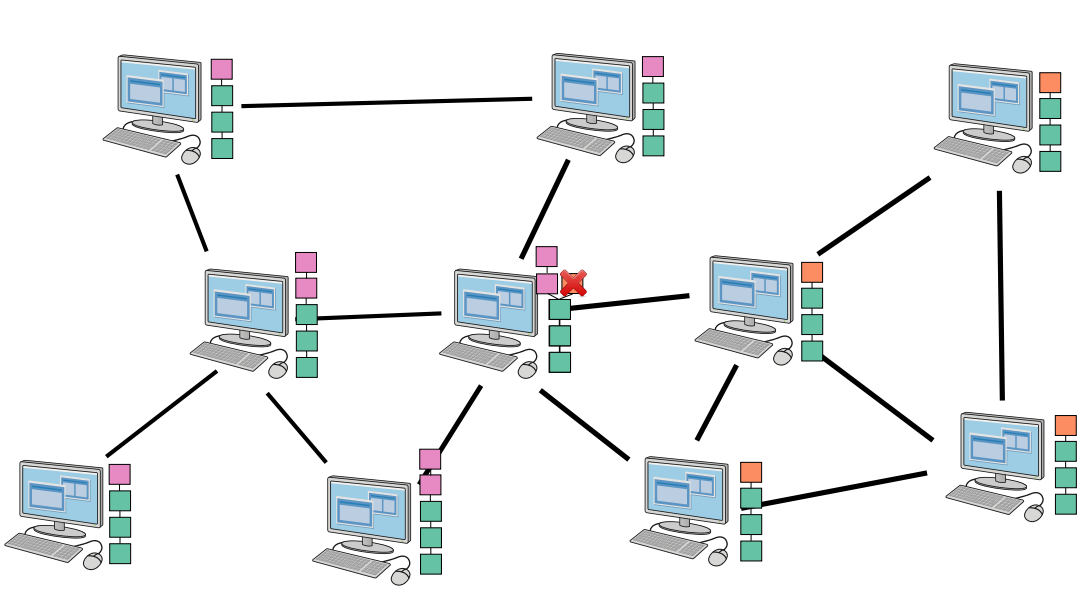
How Blocks Propagate (10)



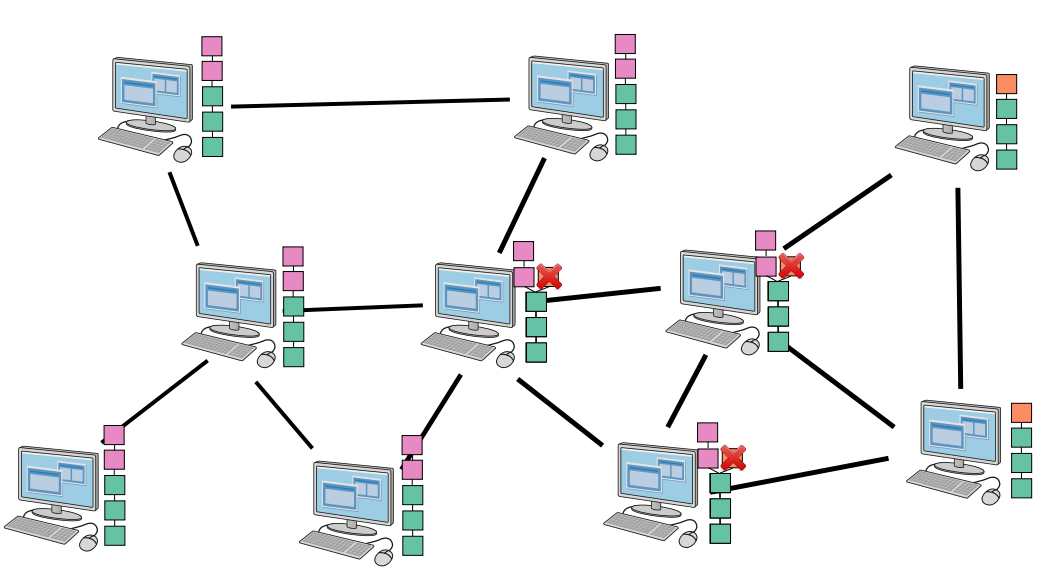
How Blocks Propagate (11)



How Blocks Propagate (12)



How Blocks Propagate (13)



How Do I Connect to the Bitcoin Network

- Pick a client (wallet): `https://bitcoin.org`
- Official client is BitcoinCore, only for Linux, Mac, Windows: `https://bitcoincore.org`
  - Supports full nodes, pruned nodes
  - Syncs the entire blockchain (currently about 560 GB)
  - Works from CLI, Qt GUI
- Lots of other clients for mobile devices
  - Lightweight nodes
  - Better pick an open source client
- Web clients. Rely on 3rd parties. Avoid.
- You may create new transactions, view balances and manage keys with any type of client

Other Kinds of Wallets

- Paper wallets: e.g. `https://www.bitaddress.org`
- Hardware wallets (Trezor, Ledger...)

Thank you!

Questions?