Consensus in Bitcoin

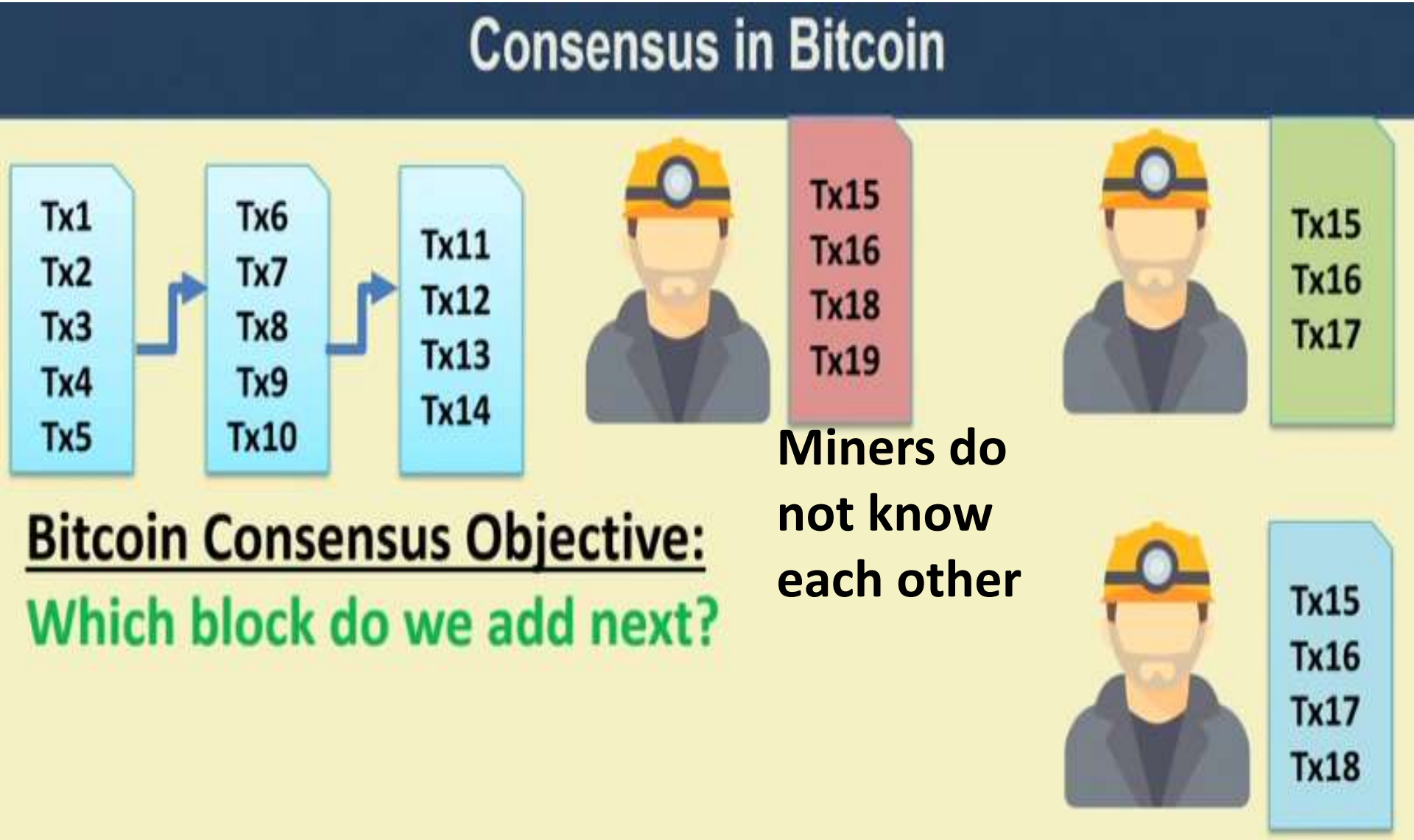
Agenda

- Need of Consensus
- Proof of Work-PoW
- Double Spending
- Attacks
- Proof of Stake-PoS
- Proof of Burn-PoB
- Proof of Elapsed Time-PoET

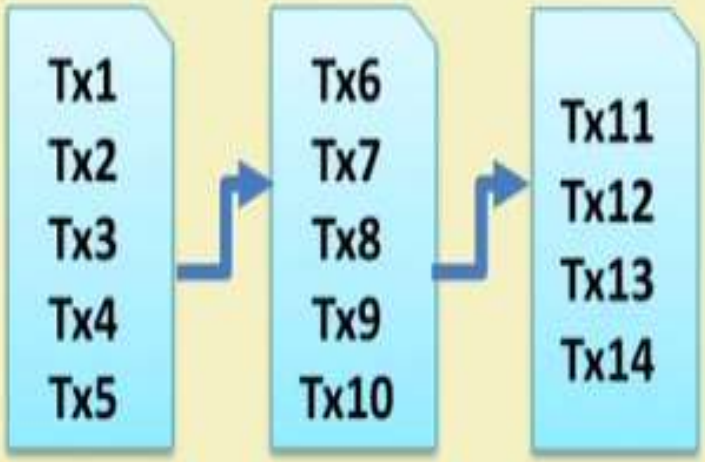
Need of Consensus

- It is not necessary that every miner will propose the same block
- The miners do not know each other
- One solution can be that all the miners can broadcast their information to the network and then apply a choice function
- This may not be feasible for bitcoin, because there is no global clock.
- Internet is asynchronous and we do not know exactly that how many miners are working and when they have added the block.
- It is not necessary that the block to be accepted must be proposed by multiple miners

Which block do we add next?



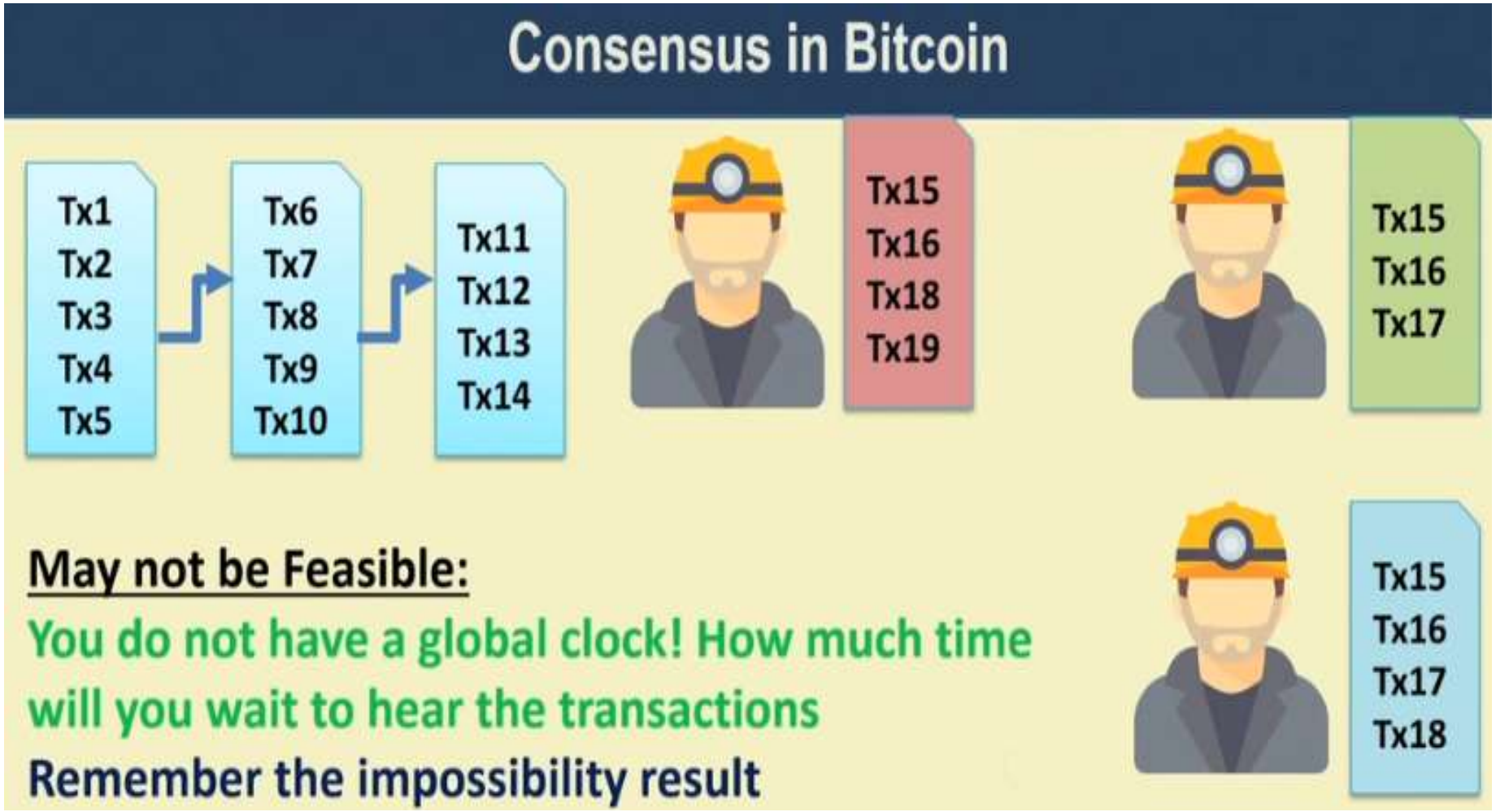
Consensus in Bitcoin



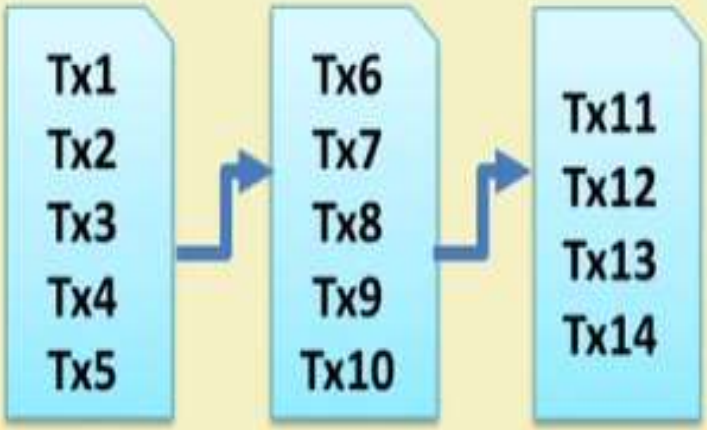
Possible Solution:

Broadcast the information and then apply a choice function – traditional distributed consensus

Impossibility Result



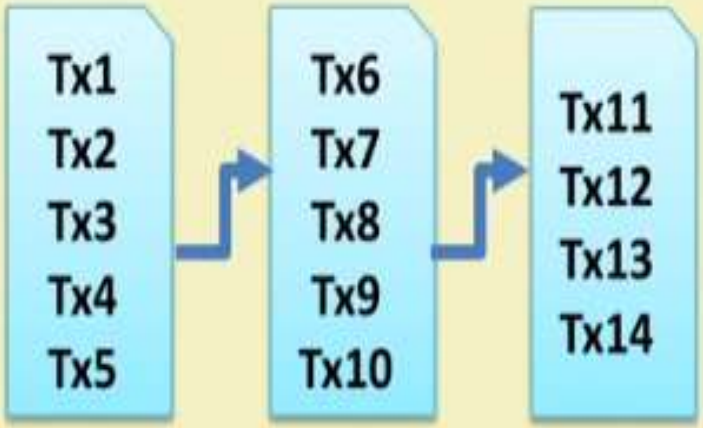
Consensus in Bitcoin



Observation - 1:

- Any valid block (a block with all valid transactions) can be accepted, even if it is proposed by only one miner

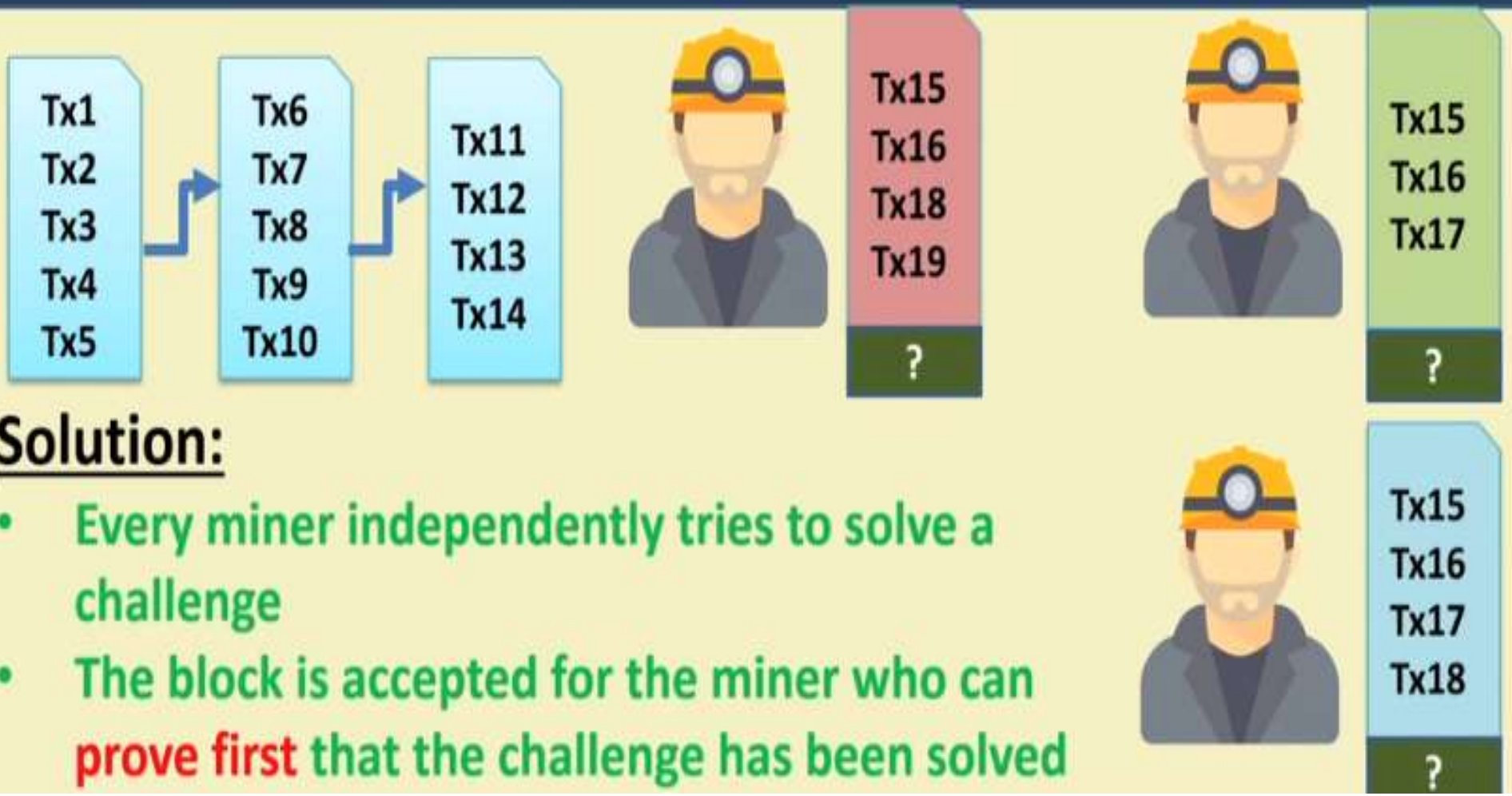
Consensus in Bitcoin



Observation - 2:

- The protocol can work in rounds
  - Broadcast the accepted block to the peers
  - Collect the next set of transactions

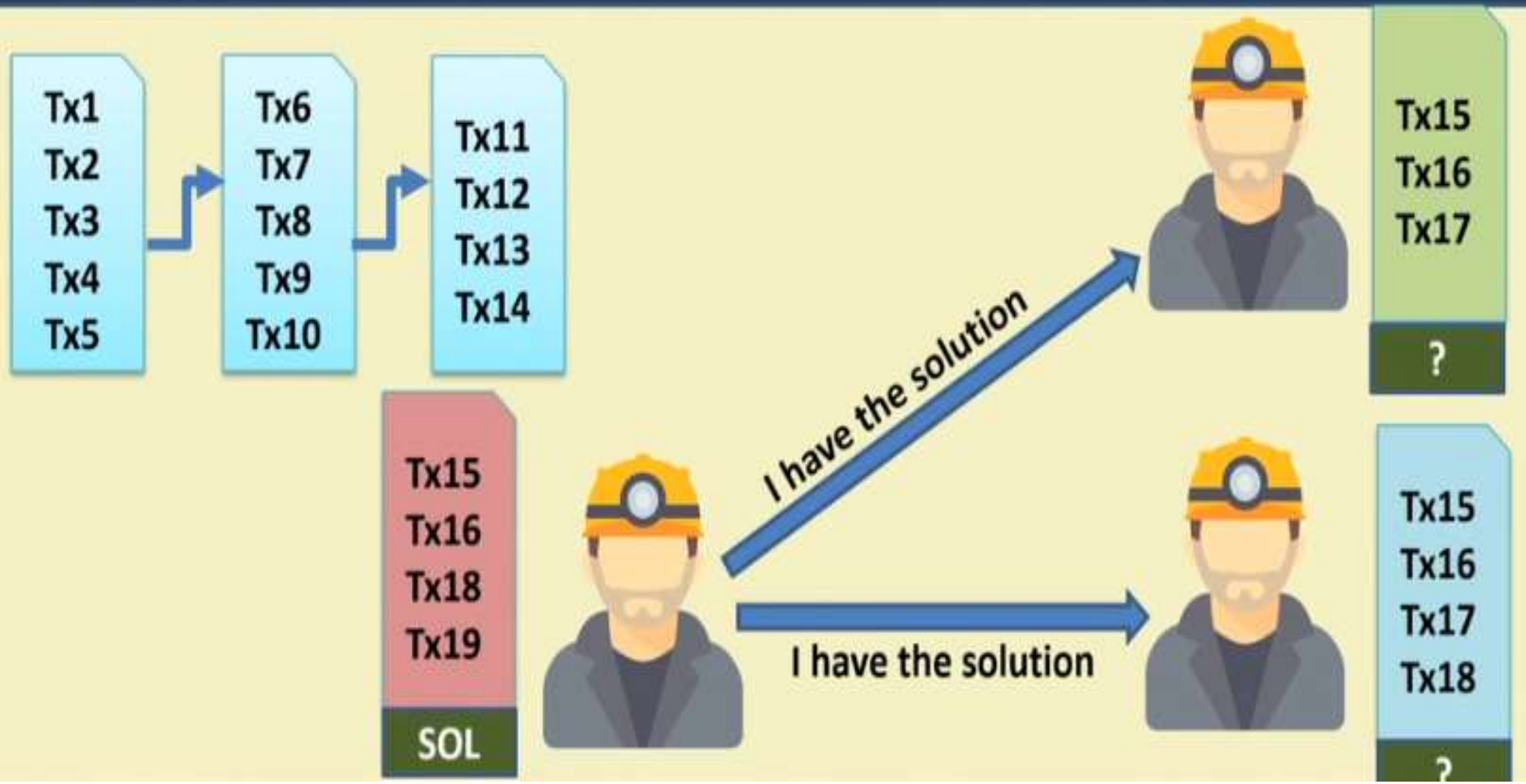
Consensus in Bitcoin



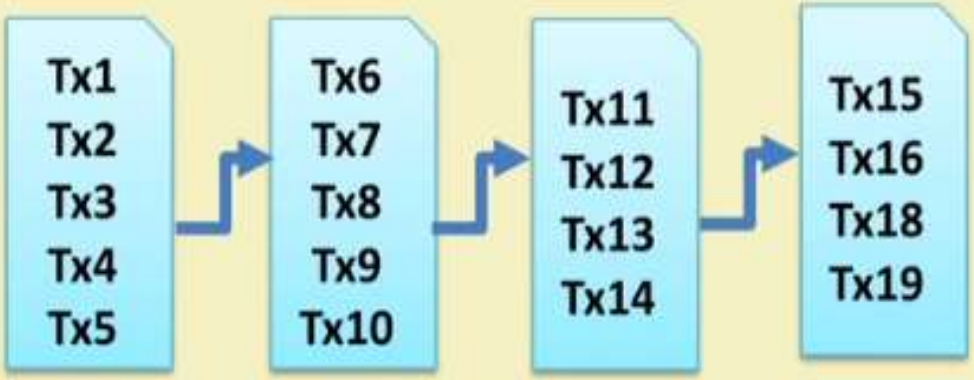
Solution:

- Every miner independently tries to solve a challenge
- The block is accepted for the miner who can prove first that the challenge has been solved

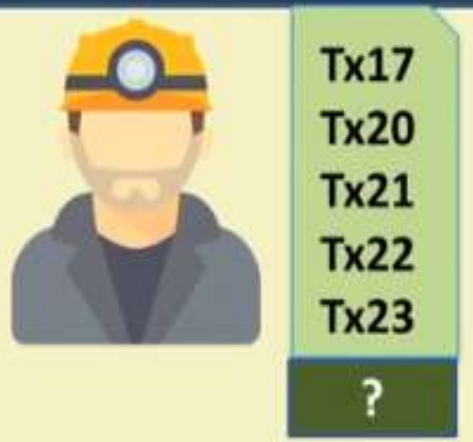
Consensus in Bitcoin



Consensus in Bitcoin



Note: Everyone can see that Tx18 and Tx19 have been committed, but Tx17 has not been committed. Include that in the next round



- As soon as some miner finds out the solution:
  - He tells all other miners that he has found out the solution
  - All other miners stop working on that block
  - Other miners can start mining new blocks coming from transactions coming from clients
- This solution can work asynchronously:
  - If I found that the transactions that I am mining are already present in the newly added block
  - Then I can stop working on those transactions and take another set of new transactions

Proof of Work (Pow)

- An economic measure to deter service abuses by requiring some work from the service requester (usually processing time by a computer)

- The idea came from Dwork and Naor (1992), to combat junk emails
  - You have to do some work to send a valid email
  - The attacker would be discouraged to send junk emails

Dwork, Cynthia; Naor, Moni (1993). "Pricing via Processing, Or, Combatting Junk Mail, Advances Cryptology". *CRYPTO'92: Lecture Notes in Computer Science No. 740*. Springer: 139–147.

- Every service requester must spend some time on the service he is asking for before actually asking for the service

- This shows that the requester has some interest in the service

Proof of Work (PoW) Features

- **Asymmetry**
  - The work must be moderately hard, but feasible for the service requester
  - The work must be easy check for the service provider

- Service requesters will get discouraged to forge the work, but service providers can easily check the validity of the work

Cryptographic Hash as the PoW

- Use the **puzzle friendliness** property of cryptographic hash function as the work
  - Given X and Y, find out k, such that $Y = Hash(X||k)$
  - It is difficult (but not infeasible) to find such k
  - However, once you have a k, you can easily verify the challenge

- Used in **Hashcash**, a proof of work that can be added with an email as a "*good-will*" token

Adam Back, "Hashcash - A Denial of Service Counter-Measure", technic
report, August 2002

Hashcash PoW

- A textual encoding of a hashcash stamp is included in an email header
  - Proof that the sender has expended a modest amount of CPU time calculating the stamp before sending the email
  - It is unlikely that the sender is a spammer
- The receiver can verify the hashcash stamp very easily
- Any change in the header requires a change in the hashcash
  - Brute force is the only way to find a hashcash

- For an email sender all the values in hash are fixed except the counter value.

- He has to check with different counter values, to have a hashcash which fulfils the requirement of first 20 digits to be zero

- Generating hash cash is tough and validating it is easy

Hashcash PoW – Recipient Side

- Recipient checks
  - The date – should be within two days
  - Email address
  - The random string – should not be used repeatedly within a certain duration (prevent replay)

- Compute the 160 bit SHA-1 hash of the entire received string

```
1:20:180401:sandipc@cse.iitkgp.ac.in::0000000267674b591257b87:6078
```

  - If the first 20 bits are not zero then it is invalid

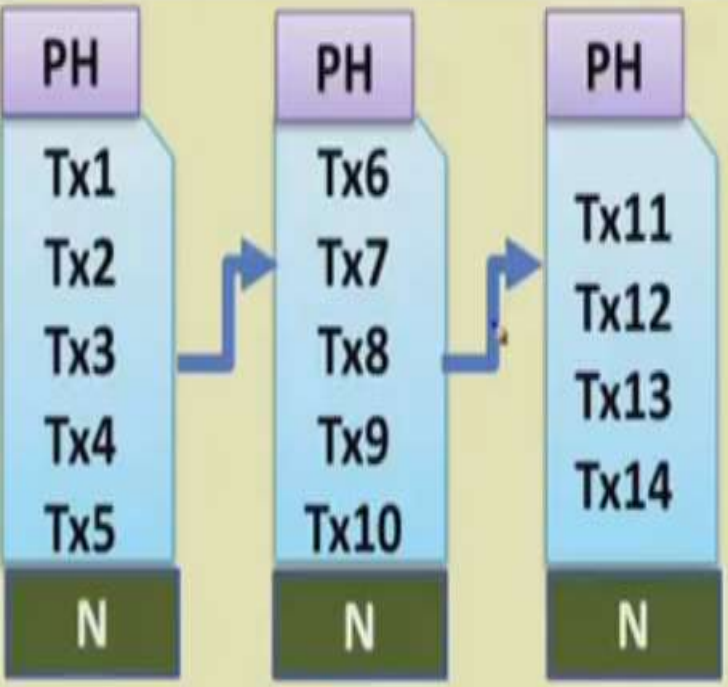
Hashcash PoW

- On average, the sender will have to try 2^{20} hash values to find a valid header (takes about a few seconds in a general purpose computer)
  - There are 2^{160} possible hash values
  - 20 zero bits at the beginning – 2^{140} possible hash values that satisfy this criteria
  - Chance of randomly selecting a header with 20 zero bits at the prefix is 1 in 2^{20}
- The recipient requires around 2 microsecond to validate

Bitcoin Proof of Work (PoW)

- Based on Hashcash PoW system
  - The miners need to give a proof that they have done some work, before proposing a new block
  - The attackers will be discouraged to propose a new block, or make a change in the existing blocks

Bitcoin Proof of Work System



BH: Block Hash

PH: Previous Block Hash

MR: Merkle Root

N: Nonce

BH should have certain zeros (difficulty) at the beginning

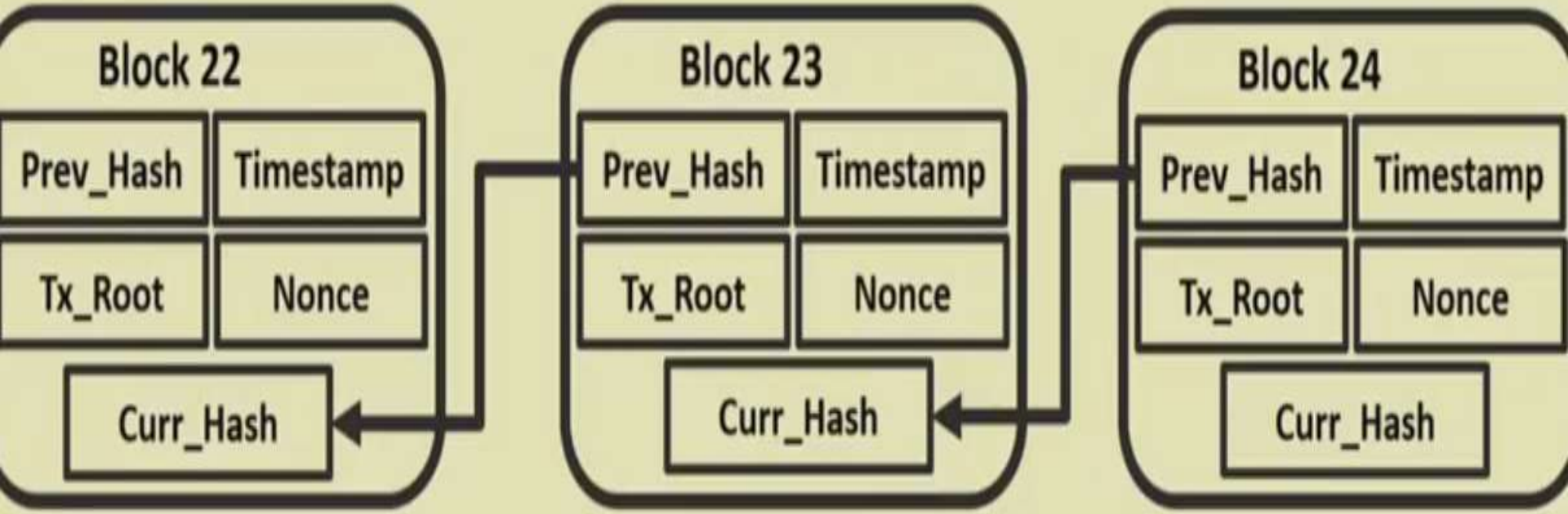
BH = Hash(PH:MR:N)

N = ?

Bitcoin Proof of Work (PoW) System

- Most implementations of Bitcoin PoW use double SHA256 hash function
- The miners collect the transactions for 10 minutes (default setup) and starts mining the PoW
- The probability of getting a PoW is low – it is difficult to say which miner will be able to generate the block
  - No miner will be able to control the bitcoin network single handedly

Tampering PoW Blockchain

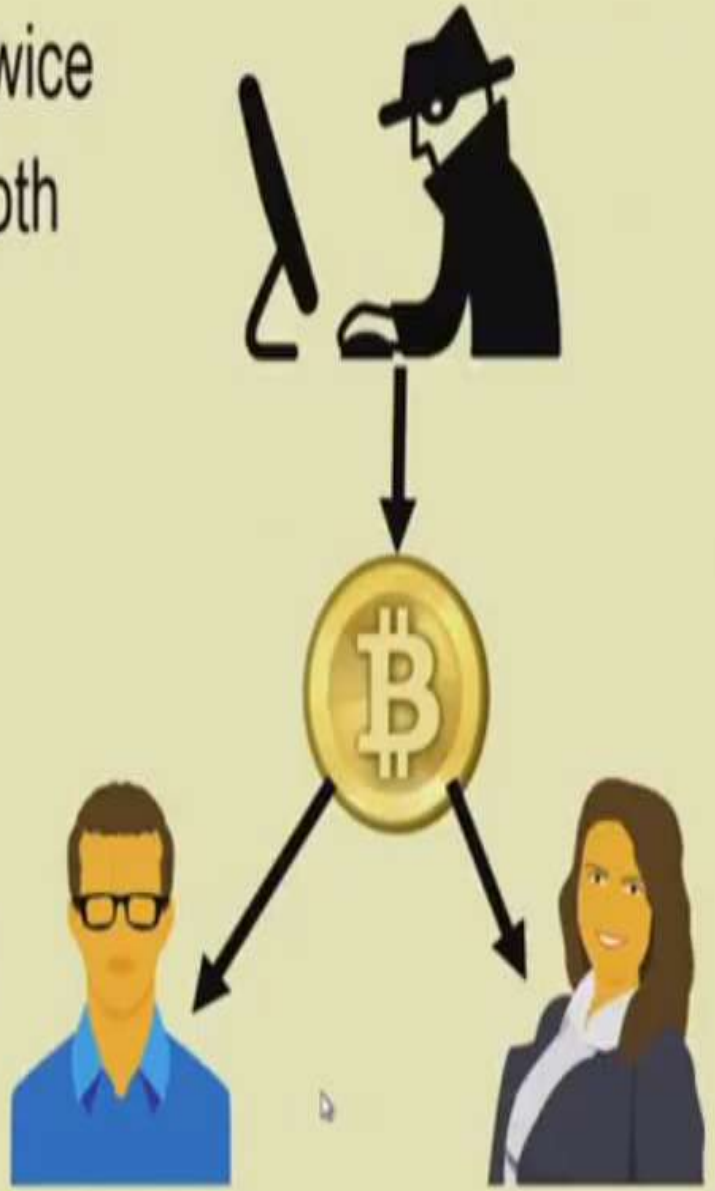


- The blockchain together contain a large amount of work
  - The attacker needs to perform more work to tamper the blockchain
  - This is **difficult** with the current hardware

Solving the Double Spending Problem

The attack: Successful use of the same fund twice

- A transaction is generated with BTC10 to both Bob and Carol at the same time



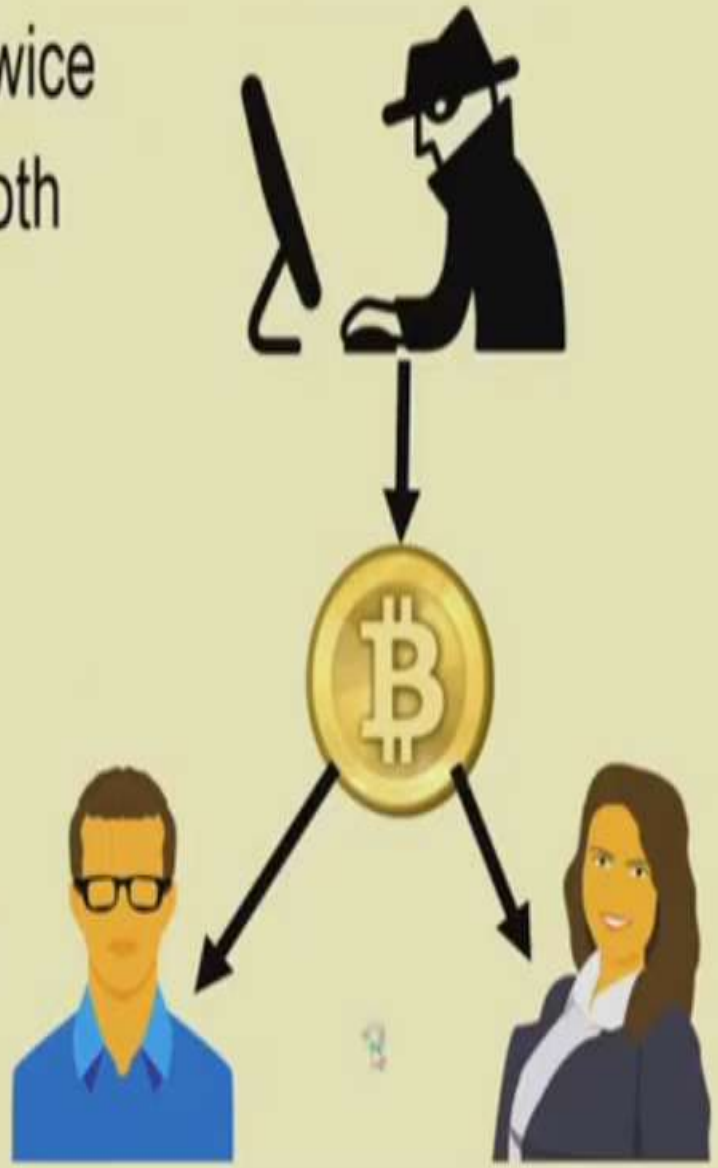
The solution:

- The transactions are irreversible (computationally impractical to modify)
- Every transaction can be validated against the existing blockchain

Solving the Double Spending Problem

The attack: Successful use of the same fund twice

- A transaction is generated with BTC10 to both Bob and Carol at the same time



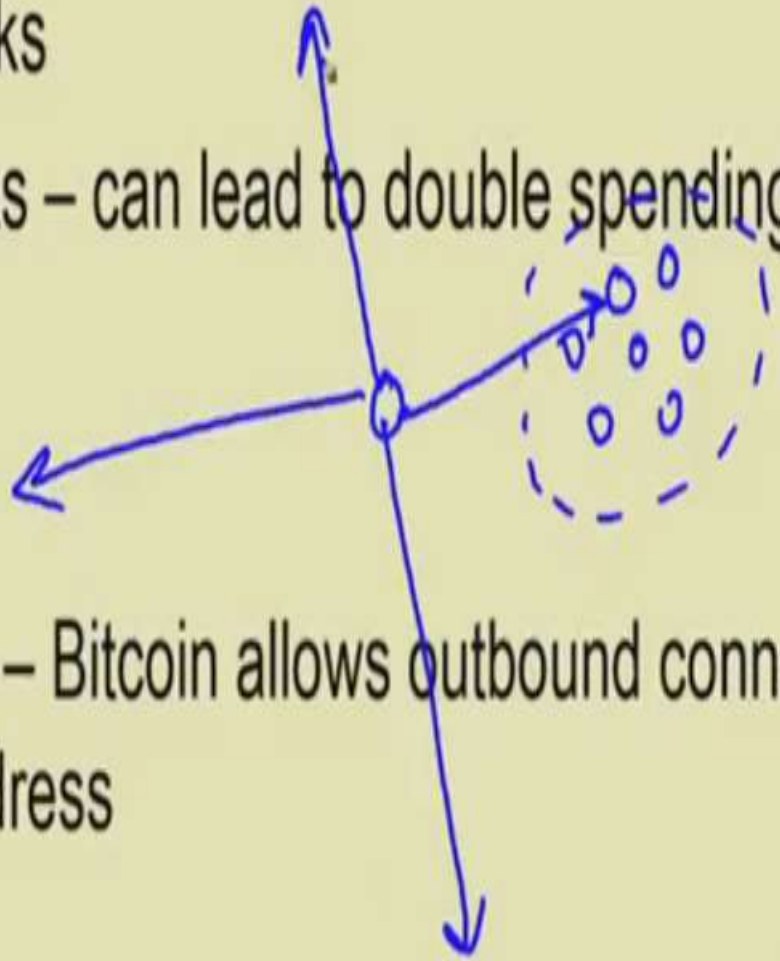
The solution:

- The transactions are irreversible (computationally impractical to modify)
- Every transaction can be validated against the existing blockchain

Sybil Attacks

Attacker attempts to fill the network with the clients under its control

- Refuse to relay valid blocks
- Relay only attacked blocks – can lead to double spending



Solution:

- Diversify the connections – Bitcoin allows outbound connection to one IP per /16 (a.b.0.0) IP address

Denial of Service (DoS) Attacks

- Send lot of data to a node – they will not be able to process normal Bitcoin transactions

- **Solutions:**
  - No forwarding of orphaned blocks
  - No forwarding of double-spend transactions
  - No forwarding of same block or transactions
  - Disconnect a peer that sends *too many* messages
  - Restrict the block size to 1 MB
  - Limit the size of each script up to 10000 bytes
  - …

Breaking Bitcoin PoW

- Bitcoin PoW is **computationally difficult** to break, but not **impossible**
- Attackers can deploy high power servers to do more work than the total work of the blockchain
- A known case of successful double-spending
  - (November 2013) "it was discovered that the GHash.io mining pool appeared to be engaging in repeated payment fraud against *BetCoin Dice*, a gambling site" [Source: https://en.bitcoin.it/]

The Monopoly Problem

- PoW depends on the computing resources available to a miner
  - Miners having more resources have more probability to complete the work
- Monopoly can increase over time (*Tragedy of the Commons*)
  - Miners will get less reward over time
  - Users will get discouraged to join as the miner
  - Few miners with large computing resources may get control over the network

Handling Monopoly and Power Consumption - Proof of Stake (PoS)

Possibly proposed in 2011 by a Member in Bitcoin Forum - https://bitcointalk.org/index.php?topic=27787.0

- Make a transition from PoW to PoS when bitcoins are widely distributed

PoW vs PoS

- PoW: Probability of mining a block depends on the work done by the miner
- PoS: Amount of bitcoin that the miner holds – Miner holding 1% of the Bitcoin can mine 1% of the PoS blocks.

Proof of Stake (PoS)

Provides increased protection

- Executing an attack is expensive, you need more Bitcoins
- Reduced incentive for attack – the attacker needs to own a majority of bitcoins – an attack will have more affect on the attacker

Variants of "stake"

- Randomization in combination of the stake (*used in Nxt and BlackCoin*)
- Coin-age: Number of coins multiplied by the number of days the coins have been held (*used in Peercoin*)

Proof of Burn (PoB)

Miners should show proof that they have *burned* some coins

- Sent them to a verifiably un-spendable address
- Expensive just like PoW, but no external resources are used other than the burned coins

PoW vs PoB – Real resource vs virtual/digital resource

PoB works by burning PoW mined cryptocurrencies

PoW vs PoS vs PoB

PoW

- Do some work to mine a new block
- Consumes physical resources, like CPU power and time
- Power hungry

PoS

- Acquire sufficient stake to mine a new block
- Consumes no external resource, but participate in transactions
- Power efficient

PoB

- Burn some wealth to mine a new block
- Consumes virtual or digital resources, like the coins
- Power efficient

Proof of Elapsed Time (PoET)

- Proposed by Intel, as a part of Hyperledger Sawtooth – a blockchain platform for building distributed ledger applications

- **Basic idea:**
  - Each participant in the blockchain network waits a random amount of time
  - The first participant to finish becomes the leader for the new block

PoET over Trusted Environments

- How will one verify that the proposer has **really waited** for a **random amount of time**?
  - Utilize special CPU instruction set – *Intel Software Guard Extension* (SGX) – a trusted execution platform
  - The trusted code is private to the rest of the application
  - The specialized hardware provides an attestation that the trusted code has been set up correctly

THANK YOU