Bitcoin

→ Permission less, No one can control.

→ A decentralized digital currency enables instant payments to anyone, anywhere in the world

—// en.bitcoin.it

→ No central authority, uses peer to peer technology.

→ Two major operations

- Transaction Mangement – Transfer of bitcoins from one user to another
- Money Issuance – regulate the money base

→ Controlled Supply : Must be limited for the currency to have value – any maliciously generated currency needs to be rejected by the network.

→ Bitcoins are generated during the mining – each time a user discovers a new block.

→ The rate of block creation is ~~edy~~ adjusted every 2016 blocks to aim for a constant two week adjustment period.

→ The node or miners who invested their time and computing power, will be awarded with the generated money.

→ The number of bitcoins generated per block is set to decrease geometrically, with a 50% reduction for every 210,000 blocks or approx 4 years.

→ This reduces, with time, the amount of bitcoin generated per block —

— Theoritically limit for total bitcoin: Slightly less than 21 million.

— Miners will get less rewards as time progress.

— How to pay mining fee — increase the transaction fee.

Source: bitcoin.com

Sending Payments:

→ Need to ensure that Eve cannot spend Alice's bitcoins by creating transactions in her name.

→ Bitcoin uses public key cryptography to make and verify digital signatures.

→ Each person has one or more addresses each with an associated pair of public and private keys (may hold in the bitcoin wallet)

⇒ Alice wish to transfer some bitcoin to Bob

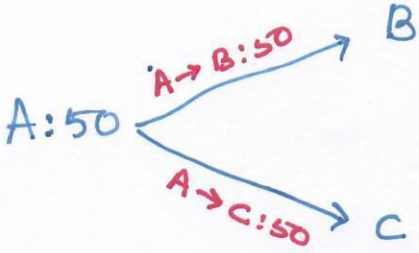
— Alice can sign a transaction with her private key

— Any one can validate the ~~the~~ transaction with Alice public key.

Alice $\xrightarrow{T_{(A\to B)}, S_A(T_{(A\to B)})}$ Bob

Double Spending Problem:

→ Same bitcoin is used for more than one transaction



→ In centralized system, the bank prevents double spending

→ How we can prevent it in decentralized n/w?

Solution

— Details about the transaction are ~~sent~~ sent and forwarded to all or as many other nodes as possible.

— Use Blockchain — a constantly growing chain of blocks that contain a records of all transactions.

— The blockchain is maintained by all peers in the ~~n/w~~ bitcoin n/w — everyone has a copy.

— To be accepted in the chain, transaction blocks must be valid must includ proof of work — a computationally difficult hash generated by the mining procedure.

— Blockchain ensures that, if any of the block is modified, all following blocks will have to be recomputed.

Bitcoin Anonymity:

- Bitcoin is permission-less, you do not need to setup any 'account', or required any email address, user name or password to login to the wallet.

- The public and private keys do not need to be registered, the wallet can generate them for the users.

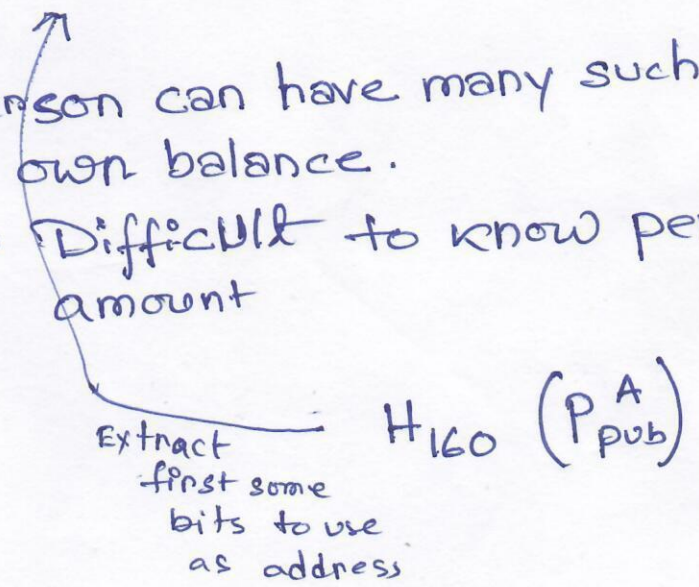
- The bitcoin address is used for transaction, not the username or identity.

- A bitcoin address mathematically corresponds to a public key based on ECDSA – the digital signature algorithm used in bitcoin.

- A sample bitcoin address:

  1PHYpmdJ33NKbUcvqb4MBMpVCKj ...

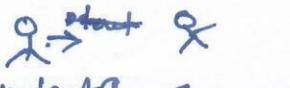
- Each person can have many such address, each with its own balance.
  - Difficult to know person owns what amount



Consensus

→ A procedure to reach in a common agreement in a distributed or decentratized multi-agent platform.

→ Important for msg passing.

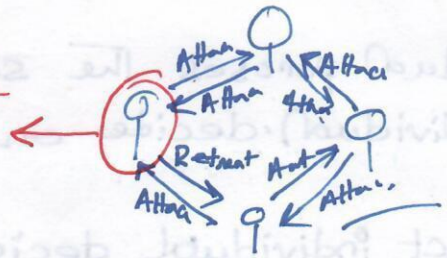


Why Consensus: Reliability and fault tolerance in a distributed System.

— Ensure correct opperations in the presence of faulty individuals.

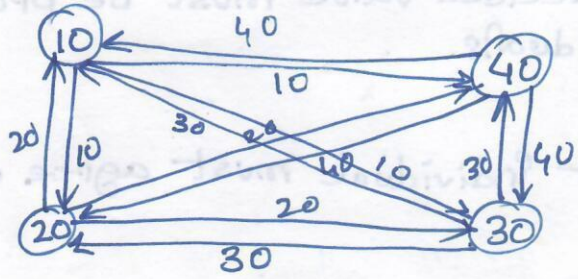
Ex:
- → Commit or transaction in a database
- → State machine peplication
- → Clock Synchronization

malicious node or byzantine node



→ Achiving consensus in DS is difficult when you have a malicious node.

Trivial



→ If no failiures

→ Broad cast personal choice to all

→ Apply a choice function that ϕ max of all chaices.

→ 4 nodes → max( ) → 40

⇒ System should be ~~faulttlo~~ faultless

⇒ System should behave in a sychronous way.

↳ it is expected that all nodes received all msg in predetermine time interval

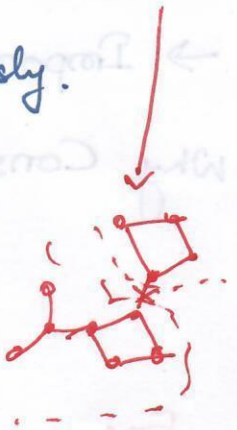
Non Trivial

There are 3 types of faults in DS.

① Crash-fault: A node suddenly crashes or becomes unavailable in the middle of the communication.

(ii) N/w or partitioned faults: A n/w fault occurs (say the link failure) and n/w gets partitioned.

(iii) Byzantine fault: A node become maliciously.
↳ difficult to handel in DS



Distributed Consensus – Properties

① Termination: Every Correct individual decides some value at the end of the consensus protocol.

② Validity: If all individual poroses the same value, the all correct individual decide on that value.

③ Integrity: Every correct individual decides atmost one value and the decided value must be proposed by some individuals.

④ Agreement: Every correct individual must agree on the same value.

Synchronaization → message passing

① Synchronous message passing: The msg must be received by the predefined time interval.

— Strong guarantee on message termination delay.

② Asynchronous message passing: There is no upper bound on the message termination or msg reception time.

— no timing constraint, msg can be delayed for arbitrary period of time.

Desinging DS in Sychronous way is easy insted of asynchronous. Consensus is difficult

→

FLP85 (Impossible Results): In a purely asynchronous distributed system, the consensus problem is impossible (with a diterministic solution) to solve if in the presence of single crash faliure.

— Results by Fischer, Lynch and Patterson (most influential paper Award in ACM PODC 2001)

— Randomized algorithms may exist. Probabilities solution

— This gives a bound that now we can design a distributed system.

Synchronous Consensus:

— Various consensus algorithm has been explored by the distributed system community

— Paxos

— Raft

— Byzantine fault tolerance (BFT)

Correctness of Distributed Consensus Protocol

- Safety: Correct individuals must not agree on an incorrect value.
  — Nothing bad happend.
- ~~Livi~~ Liveliness or Liveness: Every correct value must be accepted eventually
  — Something good eventually happens.

Consensus in a Open System:

→ The traditional distributed consensus protocol are based on
  — Message passing (when individuals are connected over the Internet)
  — Shared memory (When a common memory place is available to read and write the shared variables that every one can access)

→ Massages Passing requires a close environment — everyone need to know the identity of others.

→ But blockchain or bitcoin environment which is a open environment ~~the~~ traditional consensus protocol Paxos, Raft, BFT implimation is difficult
  ↳ msg passing is difficult, we don't know who are the in the System (identity)

→ Consensus in an Open System:
- Shared memory is not suitable for Internet grade computing
  — Where do we put the shared memory?
- Bitcoin is an open environment
  — Anyone can ~~join~~ join the n/w anytime

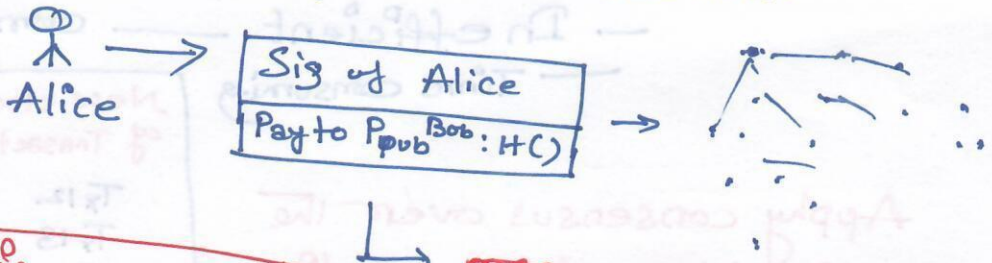
Challenge — How do we ensure the consensus in such environmet?

Why do we need consensus for Bitcoin or Open System

— Bitcoin is a peer-to-peer n/w

— Alice broadcast transaction in this p-2-p n/w

— All nodes in this n/w need to agree on the correctness of this transaction.



Flooding is more costly than of broadcast

msg not broadcast it basically flooding.

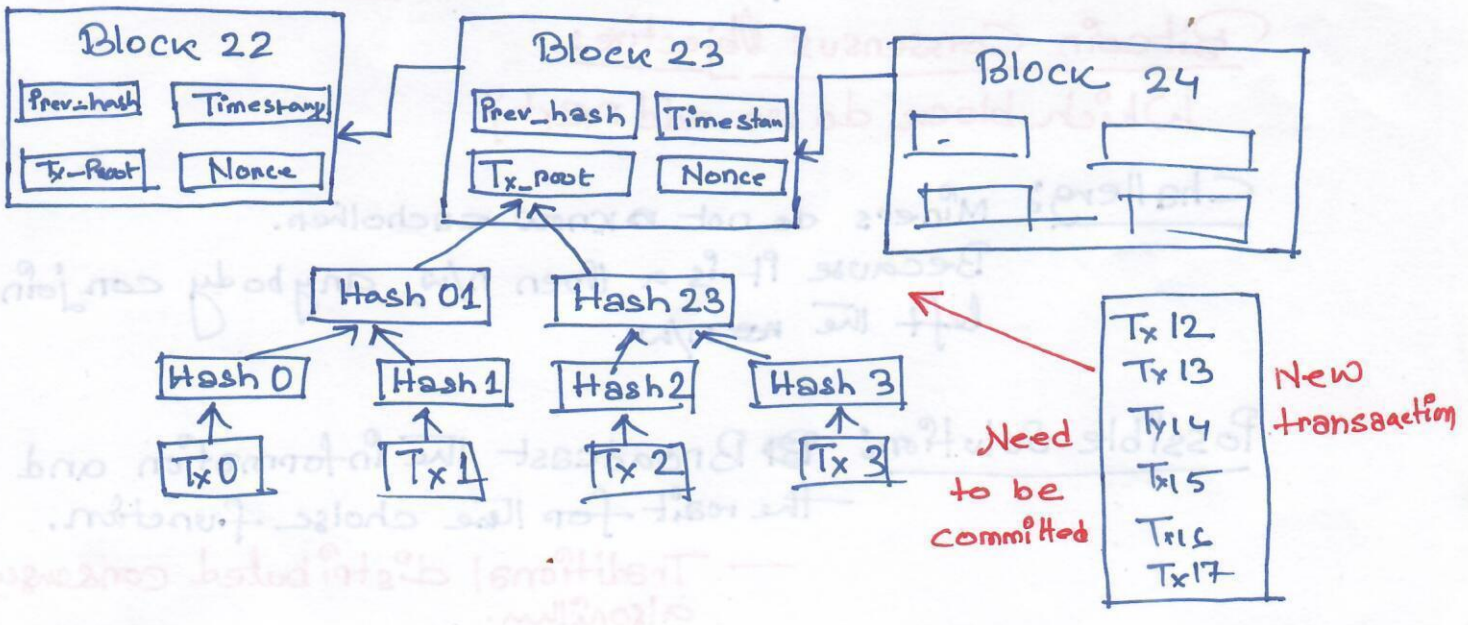
— A node does not known all the peers in the n/w — this is a open n/w

— Some nodes can also initiate malicious trasaction.

Consensus in a Bitcoin or Open n/w

— Every node has block of trasactions that has already reached into the consensus (block of commited #transactions)

— The nodes also has a list of outstanding transactions that needs to be validate against the block of committed transaction.



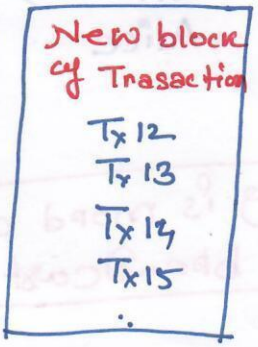
→ So new ~~the~~ Transaction are comming from the authenticated node, n/w need to validate.

Why blockchain requided here?

— Per Transaction consensus

— Inefficient —— complex

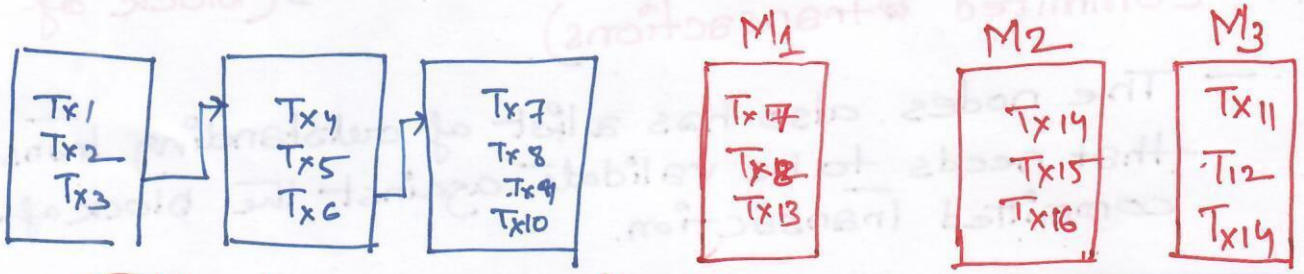
— Time consuming



Apply consensus over the entire block of Transaction.

→ Here Comes Blockchain

Miners: Miners is to build the blockchain of records that forms the bitcoin ledger. ~~The~~ These ledgers are called block, and each block containg all the different transactions that have taken place.



Bitcoin Consensus Objective:

Which block do we add next?

Challeng: Miners do not ~~to~~ know eachother.

Because it is a open n/w, anybody can join or left the ~~n~~ n/w.

Possible Solution: ~~Broadcast~~ Broadcast the information and the wait for the choise function.

— Traditional distributed consenses algorithm.

But this is not possible for Bitcoin or open kind of n/w — as Internet is ~~at~~ asynchronous and may cause arbitrarily delay in response.

↳ As we know the impossible theory of DS. (Ref FLP85)

Observation 1: Any valid block (a block with any valid transactions) can be accepted, even if it is proposed by only one miners.

Observation 2: The protocol can work in rounds.

↳ Broadcast the accepted block to the peers.

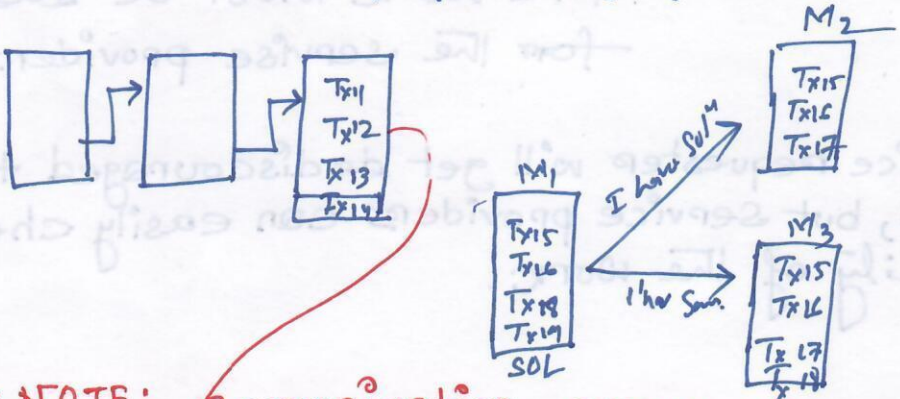
↳ Collect the next set of transactions.

→ Exclude those block which has been comitted already.

Solution: → ~~Be~~ Every miners independently tries to solve a challenge.

→ The block is accepted for the miners who can prove first that the challenge has been sloved.

↓ Proof of work (PoW)



→ NOTE: Communication among Miners can happen asynchorsly.

→ Next all miners can see Tx17 is not updated in most recent block so they can strart constructing new block with Tx17 and other most recent transactions which are present.

What should be the challenge?

↳ To generate the challenge Bitcoin relies on the mechanism called Proof of Work (PoW).

Proof of Work (PoW): A 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 an valid email.

↳ The attacker would be discouraged to send junk emails.

PoW Features:

- Asymmetry:
  - The work must be moderately hard, but feasible for the service requester.
  - The work must be easy to check for the service provider.
- Service requester will get ~~de~~ discouraged to forgot the work, but service providers can easily check the validity of the work.

→ How cryptographic Hash can be the good indicator of PoW.?

→ Given X and Y, find out K such that $Y = Hash(X||K)$

→ if X and Y is known then it is difficult to find out certain K.

→ It is difficult (but not infeasible) to find such K.

→ However once you have a K, you can easily verify the challenge.

→ Used is Hashcash, a proof of work (PoW) that can be added with an email as a good will token.

↳ Adam Back, "Hashcash – A Denial of Service Counter measure", tech report Aug-2002.

→ Valid email generator are encouraged to generate the Hashcash, Otherhand spam email generator are discouraged to generate the Hashcash.

→ Hashcash: 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.

→ The hashcash is included in the email header, looks like this:

X-Hashcash:

→ 1:20:180401:souravkaddya@nitk.edu.in:0000000267674b591257b687:6078

YY mm dd

→ Version : number of zero bit in the hashcode : date : resources : Optional extension : string of random charactecters : Counter.

e.g: first 20 bit of the hash function will be zero

→ Challenge for email sender

by using counter value email sender need to genchate hash value e.g. which have 20 no. of zero

→ only field that sender can change

→ To compute hashvalue use Tradition hash function e.g. 160 bit SHA1 algo.

→ If sender can generate 20 bit zero hash value with 160 bit SHA1 algo, then it is fine else user hash to try with diffinent counter value.

Recipient Check:

— The date should be - within two days.

— Email address

— The random string should not be used repeatedly with in a certain duration (prevent reply)

— Compute the 160 SHA1 hash of the entire received string.

1:20:180401:souravkaddya@nitk.edu.in:0000000267674b591257b687:6078

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

Analysis of Hashcash PoW:

→ On average, the sender will have to try 2^{20} hash values to find a valid header (takes about 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 random selecting a header with 20 zero bits at the prefix is 1 in 2^{20}.
- The recipient requires around 2 microsecond. to validate.

Assigment:

→ Visit hashcash.org

- Download the source try with differents numbers of zero ~~bits~~ bit targets.
- Increase the number of targeted zero bits at the hash prefix, say from 20 to 2020, at a step of 100, and observe the time to compute the hashcash.
- Use sha1sum (in Linux) to comput the sha1 checksum of the obtained obtained hashcash values from the above experiment.

How much time do you require to validate a hashcash?

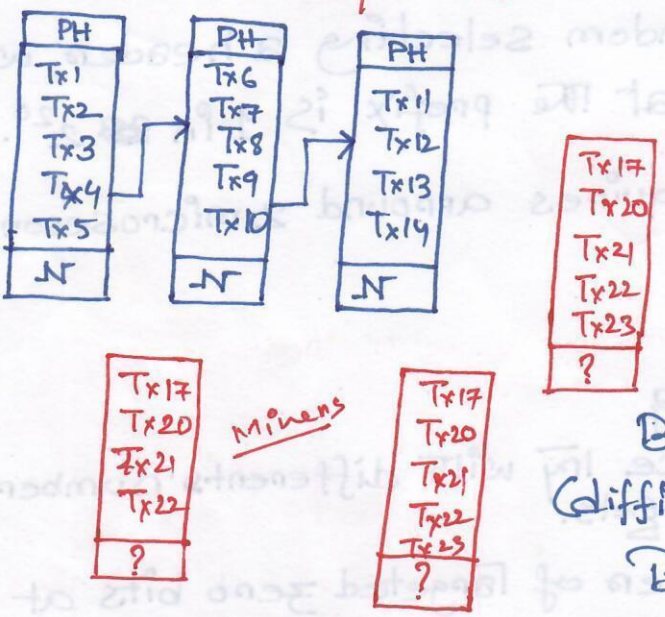
How Hashcash extended in Bitcoin PoW:?

→ It is 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 block.

Bitcoin PoW 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 = ?

→ Miners will try different value of N (nonce) to calculate the Hash value which satisfy the difficulty level.

→ Every Meners will calculate ~~the~~ and who will able to find out ~~first~~ - his block will add into the blockchain.

Information:

→ Most implementation of Bitcoin PoW use SHA-256 hash function.

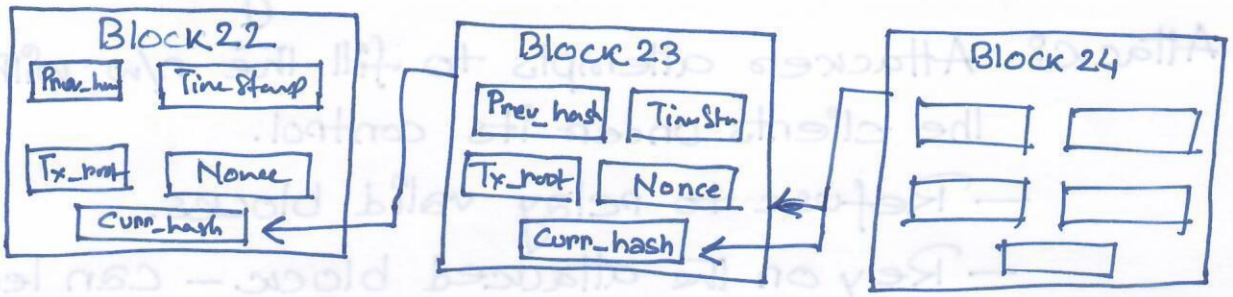
→ The miners collect the transactions for 10min (default setup) and start mining the PoW.

→ The probability of getting PoW is low — it is difficult to say which miners will abel to generate the block.

— No miner will be abel to control bitcoin n/w single handedly.

017
023
028
12 036

→ Why Bitcoin Tamper Proof?



→ The blockchain together contain a large amount of work.

→ The attecker needs to perform more work to tamper the blockchain.

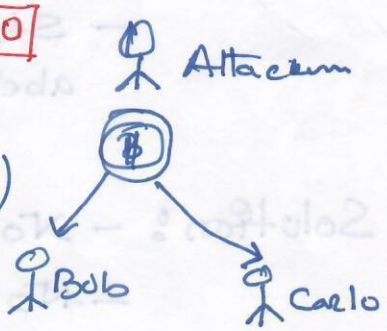
→ This is difficult with current h/w.

Double Spending Problem:

The Attack: Successful use of same fund twice.

→ A transaction is generated with BTC10 to both Bob and Carlo at the same time.

A → B : $20
A → C : $20 → A: $20



Solution:

→ The transactions are irreversible (Computationally impractical to modify)

→ Every transaction can be validated against the existing blockchain.

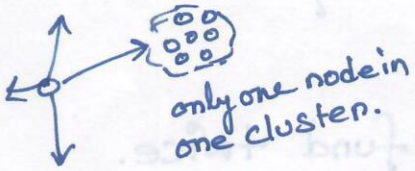
As the commited transaction are assumed to be permanante then new transaction will be validate and conform as per ~~not~~ commited transaction.

→ Double spending problem is not possible in Bitcoin n/w if we do significant PoW.

→ Some more Attacks for PoW based Systems:

Sybil Attack: Attacker attempts to fill the n/w with the clients under its control.
- Refuse to relay valid blocks.
- Rely on the attacked block – can lead to double spending.

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



Ex – if we have 172.16.—.—/16 → Bitcoin allow one one peer from this range.

Denial of Service (DoS) Attack:
- Send lots of data to a node – they will not abel to process the normal bitcoin transaction.

Solution: - No forwarding of orphaned blocks. → not part of main chain
- No forwarding of double spend transaction.
- No forwarding of same block or transaction.
- Disconnect a peer that sends too many messages.
- Restrict the block size to 1 MB.
- Limit the size of each script upto 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 ~~that~~ than the total work of the Blockchain.

→ A known case of double spending

— (NOV 2013) "It was discovered that the Gthash.io mining pool appeared to be engaging in repeated payment fraud against BetCoin Dice, a gambling site (https://en.bilcoin.ing).

Monopoly Problems:

— PoW depends on the computing resources available to the miners.

— Miners having more resources have more probability to complete the work.

— Monopoly can increase over the time (Tragedy of the ~~com~~ commons)

— Miners will get less reward over the time.

— Users will get discouraged to join as the miners.

— Few miners with large computing resources may get control over the n/w.

PoW power Consumption: →

To handling Monpoly and power consumption issue several consensus mechanisim have been proposed

— Proof of Stake (PoS).

↳ Possible proposed in 2011 by a Member in Bitcoin ~~Four~~ Forum.

→ Make a ~~transaction~~ 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 miners.

PoS: Amount of bitcoin that the miners holds — Miners holding 1% of the bitcoin can mine 1% of the PoS blocks.

Proof of Stake (PoS):

- Provides increased protection
  - Executing an attacks is expensive, you need more bitcoins.
  - Reduced incentive for attack — the attacke needs to own a majority of bitcoins — a attacks will have more affect on the attacker.
- Variants of "Stake":
  - Randomization in combination of the stake (used in Next and BlackCoin)
  - Coin-age: Numbe of coins multiplied by the number days of 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. → ~~No~~ less power consumption.
  - Expensive just like PoW, but no external resources are used other than the burned coins.

→ PoW vs. PoB : Real resources vs virtual/digital resource

→ POB : works by burning PoW mined cryptocurrencies.

PoW	PoS	PoB
• Do some work to mine a new block	• Acquire sufficient stake to mine a new block	• Burn some wealth to mine a new block.
• Consume physical resources, like cpu. Power and time.	• Consumes no external resources, but participate in transaction.	• consume virtual or digital resources like coins.
• powe hungry.	• Power efficient	• Power efficient.

Proof of Elapsed Time (PoET) :

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

Basic Idea: :-Each participants in the network waits a random amount of time.

- The first participant to finish becomes the leader of the new block.

PoET over Trusted Environments

How will one verify that the proposer really waited for a random amount of time?

- Utilize special cpu instruction set - Intel Software Guard Extension - a Trusted execution platform.
- The trusted code is private to the rest of the application
- The specialized h/w provides an attestation that the

trusted code has been setup correctly.

Mining Bitcoin:

① · Join the n/w and listen for transactions — Validate the proposed transactions.

② · Listen for new blocks — validate and re-broadcast a new block when it is proposed.

③ · Collect transactions for a predefined time and construct a new block.

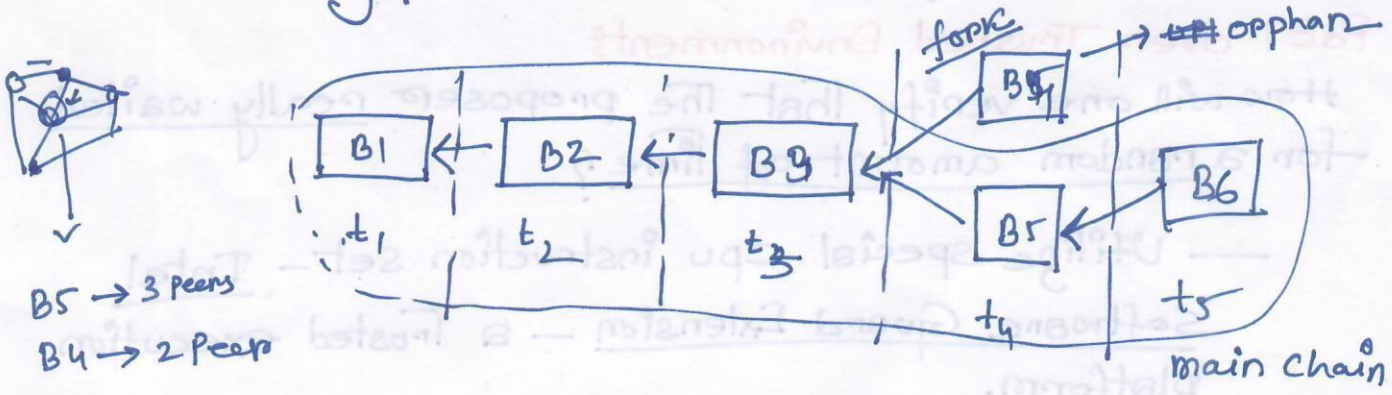
The Life of a Miner:

- Validate transaction and construct a block.
- Use hash power to vote on consensus and commit transactions with a new block.
- Store and broadcast the blockchain to the peers.

④ · Find a nonce to make the ~~new~~ valid block valid.

⑤ · Broadcast the new block — everybody accepts if it is a part of the main chain.

⑥ · Earn the reward for participating in the ~~mining~~ mining porcedure.



Mining difficulty :

- A measure of how difficult it is to find a hash below a given target.
  - bitcoin n/w has a global block difficulty.
  - Mining pools also have pool specific share difficulty.
- The difficulty changes for every 2016 blocks.
  - Desired ~~block~~ rate — one block every 10 min.
  - Two weeks to generate 2016 blocks.
  - The change in difficulty is in poportion to the amount of time over or under two weeks the previous 2016 blocks took to find

(en.bitcoin.it)

Ex : → mining hash of 256 bit
→ of which first 64 bit ~~256~~ is zero.

- compute the following for every two weeks / 2016 blocks generated

current_difficulty = previous_difficulty * (2 weeks in milliseconds) / (milliseconds to mine last 2016 blocks)

→ Based on the current_difficulty the bitcoin n/w dynamically changes the difficulty level.

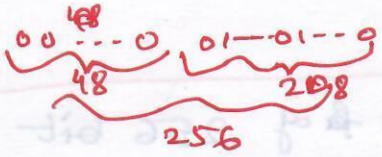
Hashrate Vs. Difficulty

- The hash is a random number between 0 and $2^{256}-1$
  - To find a block, the hash must be less than a given target.

→ We are using double SHA 256.

- The offset for difficulty 1 is $0xffff * 2^{208}$
- The offset for difficulty D is $0xffff * 2^{208}/D$
- The expected number of hashes we need to calculate to find a block with difficulty D is $(D * 2^{256}) / (0ffff * 2^{208})$ → $D * 2^{48}/0ffff$

1 → $0xfff * 2^{208}$ ⟹ initial 48 bit as zero and remaining as 1



source: bitcoin.it

→ if we increase the difficulty level i.e. value of D the we need to generate more no. of hashes to get the result.

Current mining difficulty: → 35110605528899.72 (as an 2nd April 2018)

→ 108105433845147.20 (as on Feb 6, 2025)

→ 125.86 Trillion i.e 125864590119490.00 (as on Feb 19, 2025)

H/W for mining:

① — GPU
— CPU — FPGA

② ASIC → Application specific IC — FPGA boards.

— to generate faster mining

→ Released in 2013

→ fast computation of SHA256

→ Ex → TerraMiner IV

↳ ASIC based bitcoin mining rig

↳ 2 Terahash per second

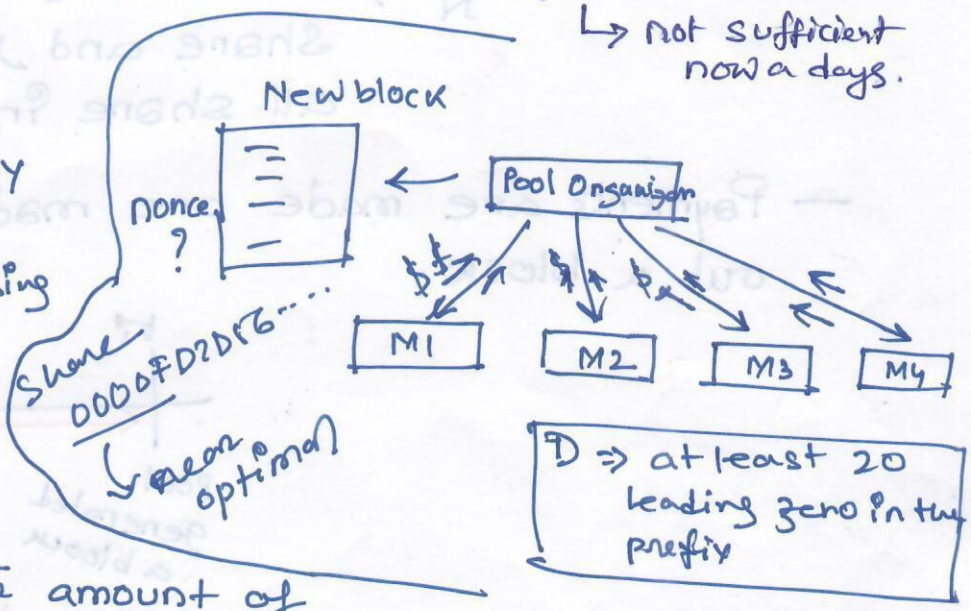
↳ Cost: USD 3500 approx.

↳ not sufficient now a days.

Mining Pool

Pooling of resources by the miners.

- Share ~~a~~ processing power over a n/w to mine a new block



- Split the reward proportionally to the amount of work they contributed.

↳ blockchain.info

Mining Pool methode:

- Contains hundreds ~~or~~ thousands of miners through special protocols.
- B: block ~~rwa~~ reward minus pool fee.
- p: Probability of finding a block in a share attempts ($p = 1/D$), D → block difficulty.

① Pay per share (PPS)

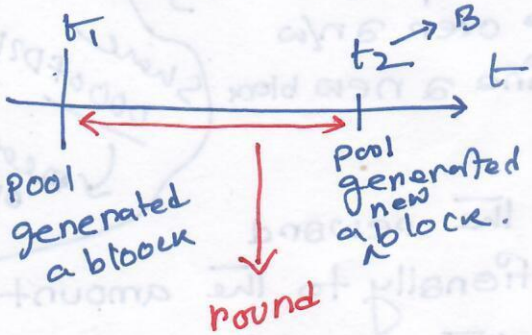
- Instant guaranteed payout to a miner.
- Miners are paid from pool's existing balance, share of a miner is $R = B \times p$.
- Miners get almost equal payment, risk is at the pool operator.

② Proportional:

→ Miners earn share until the pools finds a block (end of a mining round).

— $R = B \times \frac{n}{N}$, where n is the amount of his own share and N is the amount of all share in the round.

— Payments are made are made once a pool finds out a block.



③ Pay per Last N Share (PPLNS)

→ similal to proportional

— ~~Me~~ Miner's reward is calculated ~~o the~~ on the basis of N last shares.

— Miners get more profit for a short round.

Mining Pool — Pors and Cons:

Pros: — Small miners can participate.
— Predictable mining.

Cons: — Leads to centralization
— Discourages miners for running complete mining procedure.