Blockchain
Syllabus

Unit 1 & 2

ek 1: Introduction to Blockchain technology & it's importence.
week 2: Basic Crypto Primitives I - Cryptographic Hash
week 3: Basic Crypto Primitives II - Digital Signature.
week 4: Evolution of Blockchain Technology.
week 5: Elements of a Blockchain.
week 6: Blockchain Consensus I - Permissionless Models.
week 7: " II - Permissioned Models
week 8: Smart contracts Hands on I - Ethereum Smart Contracts
week 9: " II - Hyperledger fabric
week 10: Decentralized Identity Management
week 11: Blockchain Interoperability.
week 12: Blockchain Applications.

Blockchain

The model of Decentralization:

The Blockchain Myths:

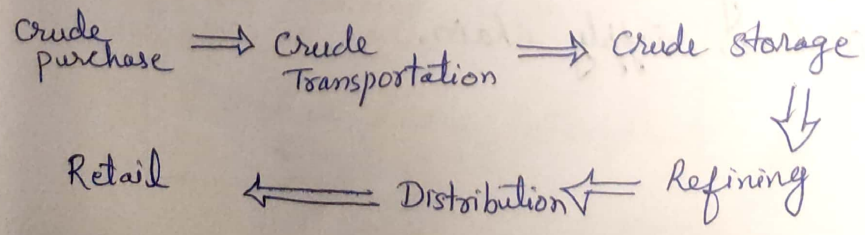
Any kind of cryptocurrencies are equivalent to blockchain.
⇒ Blockchain is not equivalent to bitcoin or any cryptocurrency.
Blockchain is the technology which is used to design the bitcoin or cryptocurrencies.

Anything and everything in the world can not be solved using blockchain.
⇒ Blockchain is good, you can solve many of the interesting problems using blockchain, but it is not like that you can change the society using blockchain.

You can not replace a database with a blockchain.
⇒ Blockchain is not a distributed database.
⇒ Blockchain is not designed to securely store ANY data.
You can not store any data on the blockchain.

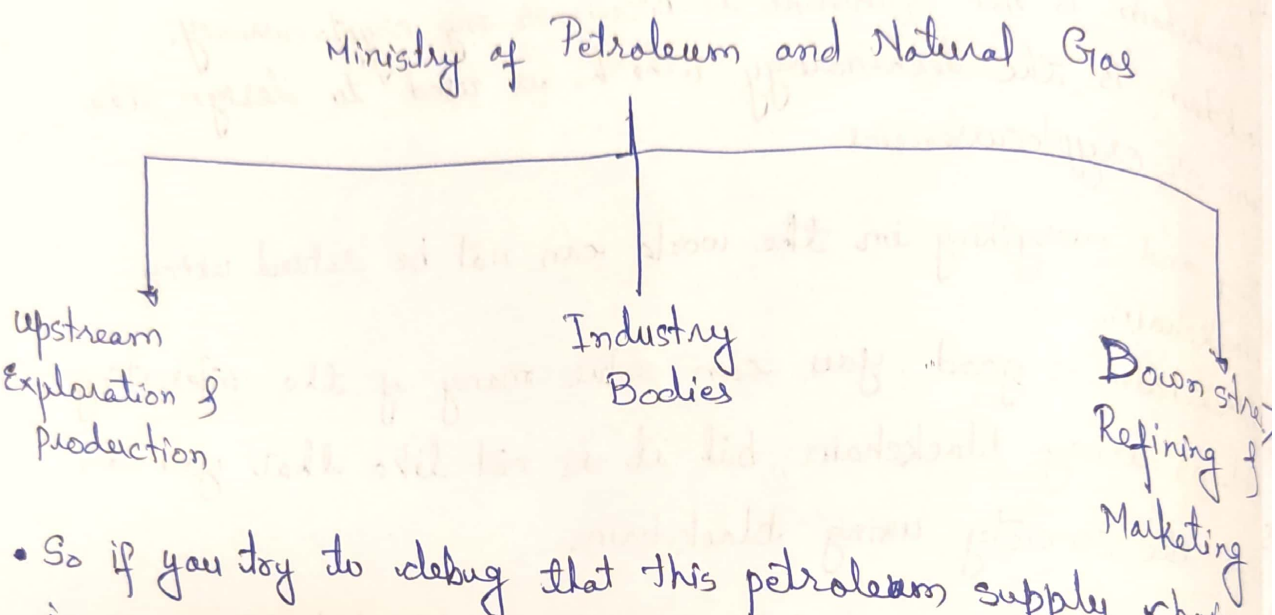
- Objective of the course is
  - To avoid all the hypes and apply blockchain as a solution at the right place where it is really needed.
- What is the place:
  - The terminology that roams around the blockchain is "Decentralization":
  - To understand the term decentralization let us understand the usecase Supply Chain Management.

Example: Supply Chain in Petroleum Industry:



- So that is the kind of a very simplified view of the end to end supply chain management for petroleum industry
- Who are the players in this end to end supply chain management

⇒ Supply chain Petroleum India.



- So if you try to debug that this petroleum supply chain in the context of India, so the entire comes under the Ministry of Petroleum & Natural Gas, & under that ministry there are certain players which are at the upstream directly they are mostly in the exploration and the production of the crude oil side. So they take the crude oil & then they produce or explore the location where you can get the crude oil and then they supply that crude oil to the corresponding storage & they are the downstream operators or the downstream players who basically refines the crude oil & then take care of marketing current products which are basically extracted from the crude ~~oil~~ oil.

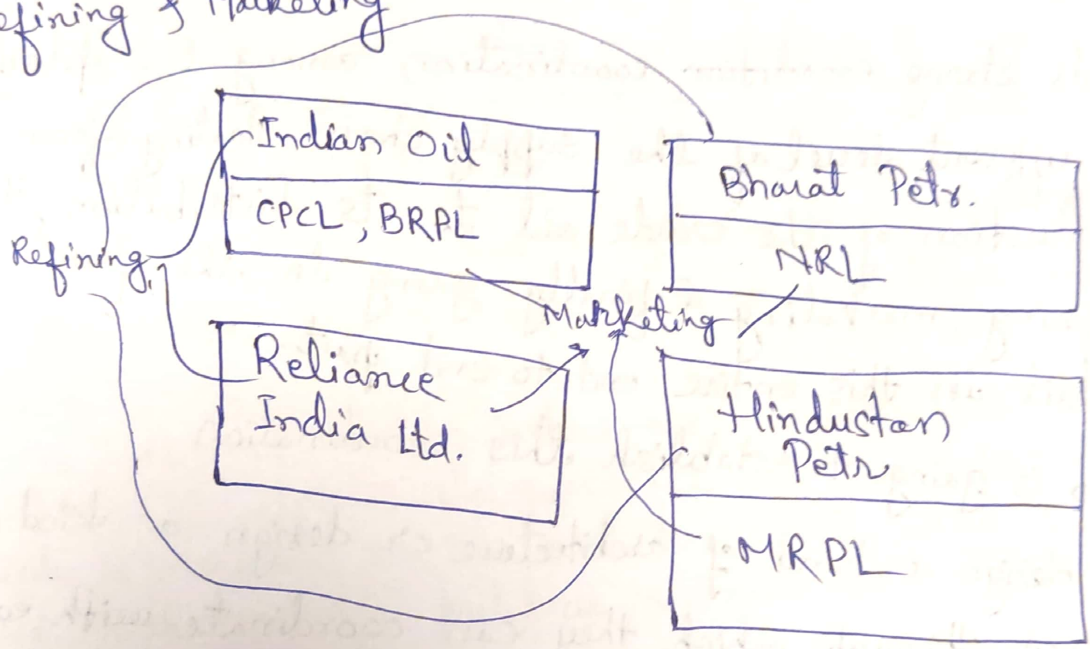
In between the upstream & downstream operators there are different other industry bodies which are associated with this entire petroleum supply chain.

Upstream Exploration & Production

- ONGC
- OVL
- Oil India Ltd
- Pvt E&P Co

Different players / different companies which are participating in upstream supply chain of petroleum Industry.

Downstream Refining & Marketing



Industry Bodies

- Petroleum Planning & Analysis
- Center for High Technology
- PRCA
- PetroFed
- Oil Industry Safety Directorate
- Petroleum India International

Minimum Requirements for successful petroleum supply chain Management:

- Minimization of material procurement.
- Maximization of manufacturing capacity & Sales.
- Meet demand numbers.
- Respond quickly to market opportunity by purchasing the produc shortfall from other players.
- Objective of each production unit would be to maximize the throughput & its margin.
- procurement would purchase the feedstock with not the best yields at lowest cost.

⇒ Needs strong ~~coordition~~ coordination among the players at different level of the supply chain starting from the production of the crude oil to its distribution, then refining, marketing & finally going to the final retail in this entire end to end part.

⇒ Who is going to establish this coordination

* If design a kind of architecture or design a kind of platform through which they can coordinate with each other in a successfull way, i.e they can get the informat about demand, understand particular company / organization has this much of free petrol / diesel that can be possib procure to meet the requirements. So if I can share s kind of information among them, then I can possibly better a kind of management of this, the product in the end to e supply chain.
* So this is basically a kind of decentralization / decentraliz architecture, where you have multiple different

usiness organizations, everyone has their own governing
tructure, everyone has their own policy of doing the
arketing, doing the business, but at the end of the day
ey are working on the common product, working on a
ommon marketplace. And that way if they can collaborate
ith each other or coordinate with each other in some
ay everyone might get benefited out of it.

Here in decentralized architecture, there is no central body or central co-ordinator who is co-ordinating all these different players in supply chain, rather it is like that everyone is taking their own decision in an independent way rather every of the, each of these organizations have their own policy to maximize the profit, doing the business & so on & they are in a common market place.

Now question here is how to get realtime information from the multiple stakeholders, so there are multiple different solutions

One is web based solution where each information is available but what is the guarantee that the information which is available on the portal is correct, who is going to manage the portal as in decentralized architecture there is no centralized authority then who is going to validate that the information is correct or not. And another problem which might arise that what if someone denise that information later on.

So we need a solution i.e decentralized - no one trust each other but they should cooperate with each other.

So to solve this Blockchain is the answer.

Blockchain can specifically solve this particular prolem of decentralization. in this context.

Trustless Decentralization = Blockchain

What is Blockchain:

- Decentralization with a Blockchain
- Fundamental properties of Blockchain
- Formal definition of Blockchain

Decentralization with a Blockchain

- Problem is that I have multiple different players, or multiple different stakeholders & they want to coordinate with each other. The fundamental principle is that I need something where I can submit information, & then I would have a way to verify that information.

⇒ Blackboard example of petroleum supply chain where every stakeholder will write the actions they do & i.e permenent.

- This board has certain properties:

① — The board has infinite space.
- You do not need to erase anything
- Hence keeping all the historical information record & any kind of fraud & corruption can be found

② — Everyone can see all the logs and verify.

③ — Anychange in information is visible to everyone

④ — The board is not erasable, no one can deny later

⇒ How we can utilize this board further

Problem is who is going to maintain this bulletin board

* purely decentralized i.e everyone envolved will maintain individualy & independently.
* everystakeholder will have their own blackboard now
* Everyone holds the exactly same copy of data at the same instance of time.

Now this blackboard is blockchain:

That is blockchain:

- An immutable append-only ever-growing chain of data.
- Data once added can not be deleted or modified later.

– immutable = No one can change

– append only = Not erasable.

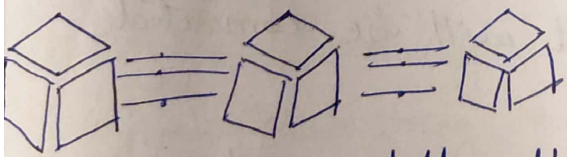
- There is no central database to store the chain – everyone keeps a copy of the chain & process the data locally.
- New information is added to the chain in the form of new blocks. & ensuring same block will be appended in every local chain.

⇒ the entire data structure is basically a chain of blocks hence we call is as blockchain.

⇒ Blockchain ensures that every party has the same view of the blockchain always.

⇒ The information which i have in the blockchain it is transparent to everyone. so everyone can verify & validate.

mal Definition of Blockchain:



decentralized immutable append-only public ledger.

① ② ③ ④

④ everyone is going to maintain their own copy of the blockchain.

Four characteristics of Blockchain

Basic Cryptographic Primitives:

* Primitives means some of the fundamental concepts & technology techniques that are used for implementing Blockchain in real world.
* One of the cryptographic primitives for blockchain is Hash function.
* Basic cryptographic primitives behind blockchain technology
  1. Cryptographically Secure Hash Functions.
  2. Digital Signature

⇒ Hash function: Used to connect the blocks in a "chain" in a tempor proof way.

⇒ Digital Signature: Digitally sign the data so that no one can "deny" about their own activities. Also others can check whether it is authentic.

1. Cryptographic Hash Functions:

- Cryptographic hash functions take any arbitrarily sized string as input i.e

  Input M: The message which represents the message that will be converted into the fixed size output.
- Fixed Size Output (We typically use 256 bits in Blockchain)

  Output H(M): We call this as message digest.
- These functions are efficiently computable.

Cryptographic Hash Functions: Properties:

① Deterministic:
Always yields identical hash values for identical input data.

② Collision-Free:
If two messages are different, then their digests also differ.

③ Hiding:
Hide the original message, remember about avalanche effect i.e if I have message m_1 & apply the cryptographic hash function on m_1 to get its digest $h(m_1)$ & then I have another message m_2 for which I get the digest $m_2, h(m_2)$ then by looking at these digests $h(m_1)$ & $h(m_2)$, I should not be able to figure out that this m what would be the m_2 that would generate that $h(m_2)$ meaning that even if there are small variations in ~~(m_1)~~ m_1 & m_2 the digests $h(m_1)$ & $h(m_2)$ they should not be similar. i.e if m_1 & $h(m_1)$ is known & then we should not predict $h(m_2)$ for m_2.

④ Puzzle Friendly:
Given two string x & y here x is a message & y is output of a hash function. We would like to find out a k such that y is a hash of x concatenated with k

$$y = H(X||k)$$

meaning that given x can we determine a k so that the hash of x concatenated with k generated a y & this y will have certain properties that need to

to be satisfied.
which is used for solving mining puzzle in Bitcoin Proof of work.

① ⇒ Collision Free:

- Hash functions are one way; Given an x, it is easy to find $H(x)$. However; given an $\xi H(x)$, one can not find x.
- It is difficult to find x and y, where $x \neq y$ but $H(x) = H(y)$.
- Note the phrase difficult to find, collision is not impossible.
- try with randomly chosen inputs to find out a collision - but it takes too long.

How do we guarantee:

- It may be relatively easy to find collision for some hash functions.
- Birthday paradox: Find the probability that in a set of n randomly chosen persons, some of them will have the same birthday.
- By Pigeonhole principle, the probability reaches 1, when number of people reaches 366 (not a leap year) or 367 (a leap year).
- 0.999 probability is reached with just ~70 people, & 0.5 probability is reached with only ~23 people.

Hash as Message Digest:

- If we observe $H(x) = H(y)$, it is safe to assume $x = y$.
- We need to remember just the hash value rather than the entire message ~ we call this as the message digest.
- To check if two messages x and y are same i.e whether $x = y$, simply check $H(x) = H(y)$
- This is efficient because the size of the digest is significantly less than the size of original messages.

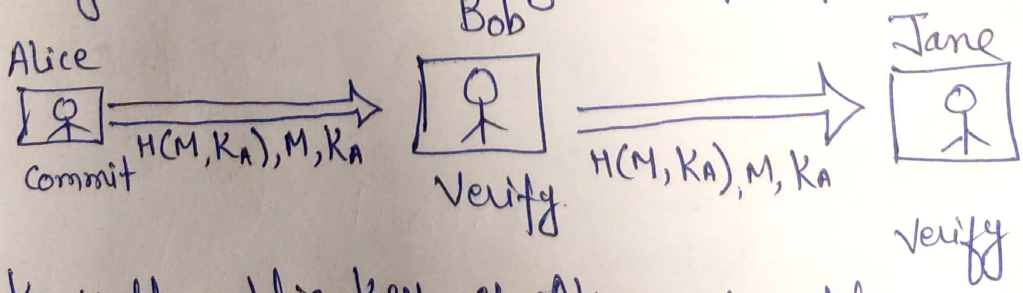
⇒ In SHA 256 message digest is of size 256 i.e 64 hexadecimal digits.

⇒ Hashing illustration through website given.

(2) Information Hiding through Hashing:

- Given an $H(x)$, it is computationally difficult to find x.
- The dificulty depends on the size of the message digests.
- Hiding helps to commit a value then check it later.
- Compute the message digest & store it in a digest store commit.
- To check whether a message has been comitted, match the message digest ~~of the~~ at the digest store.

⇒ Message Commitment through Multiple parties.



K_A is the public key of Alice — A public identity that only Alice can have

③ Puzzle Friendly:

- Say M is chosen from a widely spread distribution, it is computationally difficult to find a K, such that $Z = H(M||K)$ where M & Z are known a priori.
- A search puzzle (Used in Bitcoin Mining)
  M & Z are given, K is the search solution
- Puzzle friendly property implies that random searching is the best strategy to solve the above puzzle.

sic Cryptographic Primitives - II

th Function

ure Hash Functions

sh Function — SHA256

HA256 is used in Bitcoin mining - to construct the
coin blockchain.

cure Hash algorithm (SHA) that generates 256 bit message digest.

part of SHA-2, a set of cryptographic hash functions
igned by ~~US~~ United States National Security Agency (NSA)
ey.

256 Algorithm

rocessing:

d the message such that the message size is multiple
512.

- Suppose that the length of the message M is l; &
  l mod 512 ≠ 0
- Append the bit "1" at the end of the message
- Append k zero bits, whe k is smallest no-negative.
  Solution to the equation l+1+k = 448 mod 512
- Append the 64-bit block which is equal to the
  number l written in binary.
- Total length gets divisible by 512.

Partition the message into N 512-bit blocks $M^{(1)}, M^{(2)} \ldots M^{(N)}$

Every 512 bit block is further divided into 32 bit sub
blocks $M_0^{(i)}, M_1^{(i)} \ldots M_{15}^{(i)}$

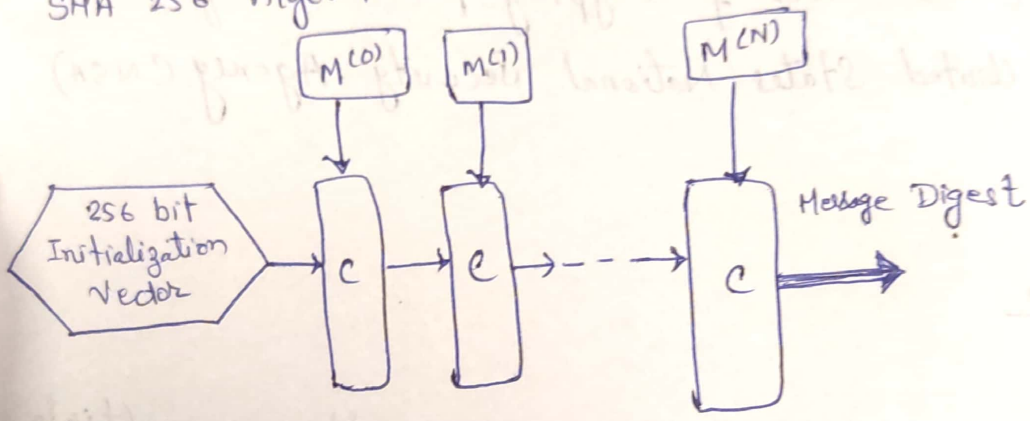
④ The message blocks are processed one at a time

⑤ Start with a fix initial Hash value $H^{(0)}$

⑥ Sequentially compute $H^{(i)} = H^{(i-1)} + C_{M^{(i)}}(H^{(i-1)})$

is the SHA-256 compression function and + means mod 2^{32} addition. $H^{(N)}$ is the hash of M

SHA 256 Algorithm:



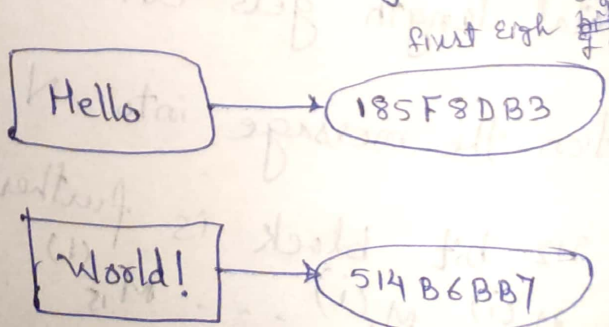
Patterns of Hashing Data:

1. Independent Hashing
2. Repeated Hashing
3. Combined Hashing
4. Sequential Hashing
5. Hierarchical Hashing

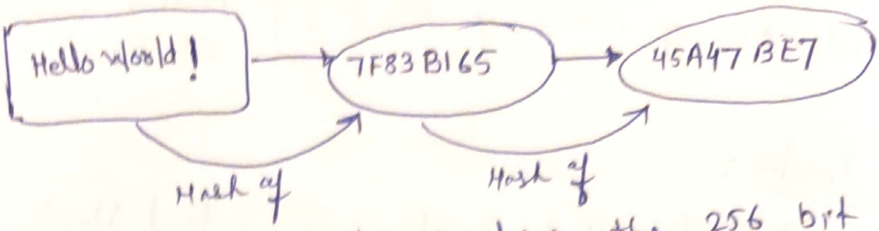
⇒ If there are more than one pieces of data, what are the different ways we could combine their hashes so this is what we call as the patterns of hashing data.

① Independent Hashing:

⇒ for two pieces of message calculate Hashe independently.

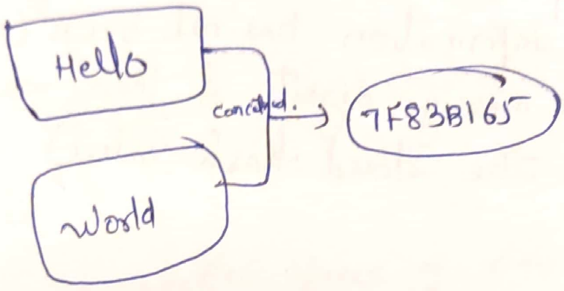


② Repeated Hashing:

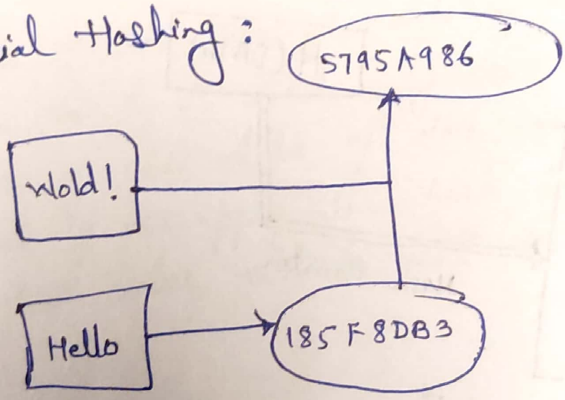


If we repeatedly run this hashing the 256 bit output of the first step & then perform Hashing.

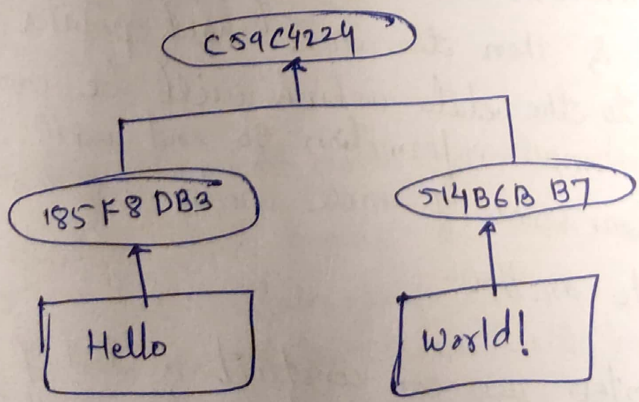
③ Combined Hashing:



④ Sequential Hashing:



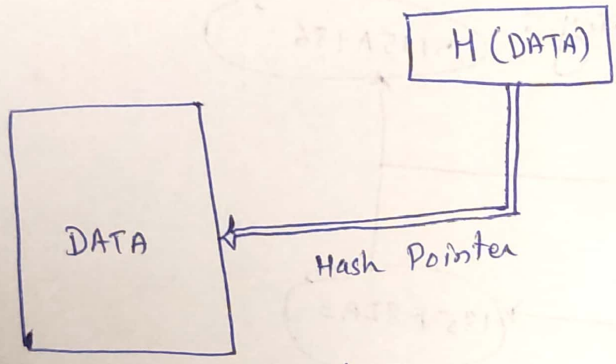
⑤ Hierarchical Hashing:



- Hash Pointers
- Hash chain
- Construction of chain of Blocks:
  - Hash Function
  - Hash Pointer
  - Merkle Tree
  - Blocks

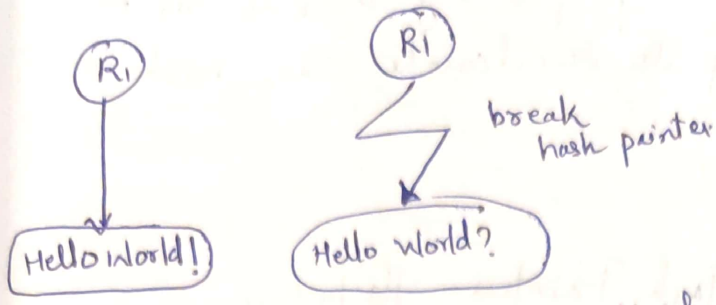
Hash Pointer:

- A cryptographic Hash Pointer (often called Hash Reference) is a pointer to a location where
  - Some information is stored.
  - Hash of the information is stored.
- With the hash pointer, we can
  - Retrieve the information
  - Check that the information has not been (modified (by computing the message digest & then matching the digest with the stored hash value)



- Reminds a linked list.
- Hash pointer kept in another block of data & make it point to this data & then the next hash pointer we can make to point to the data which will be composed of this hash & some more information so and with that we can proceed & have more & more nodes in this linked list from one node to another.
- By using hash pointers we can construct a list of such data connected together but the added advantage is that if there is any modification to the data made then it will get detected when we are going to

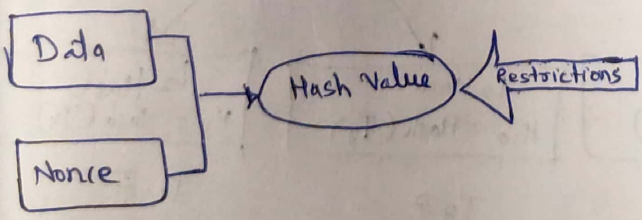
* Temper Detection using Hash Pointer:



Miss match between Hash value of Hello world! & Hello World? as hash pointer keeps track of message Hellow world! & it's hash value.

* Making Tampering a Hash chain Computationally Challenging:

- There will be another piece of data along with the original data of the message which is called nonce i.e number which is used only once so that this concatenation of this data of this data with the nonce & that if we apply the hash function, i will generate a hash value which will have certain restrictions



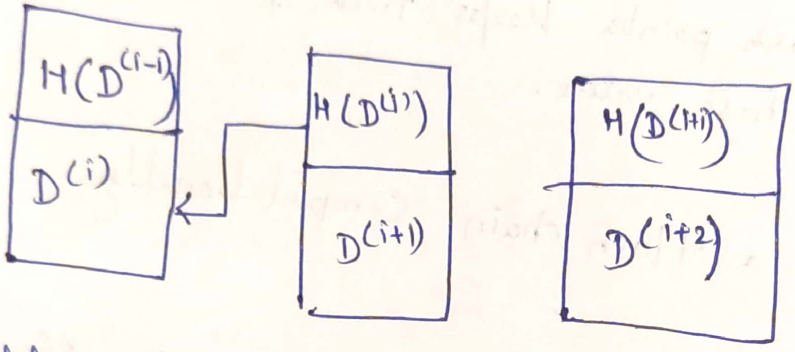
- While generating the hash value of the data, we put certain restrictions that means the hash value should be such that it has certain properties.
- So along with the data there will be another piece of information/data which is called as nonce means number which is used only once, so that this concatenations of this data with the nonse on

& that if we apply hash function, i will generate a hash value which will have certain restrictions

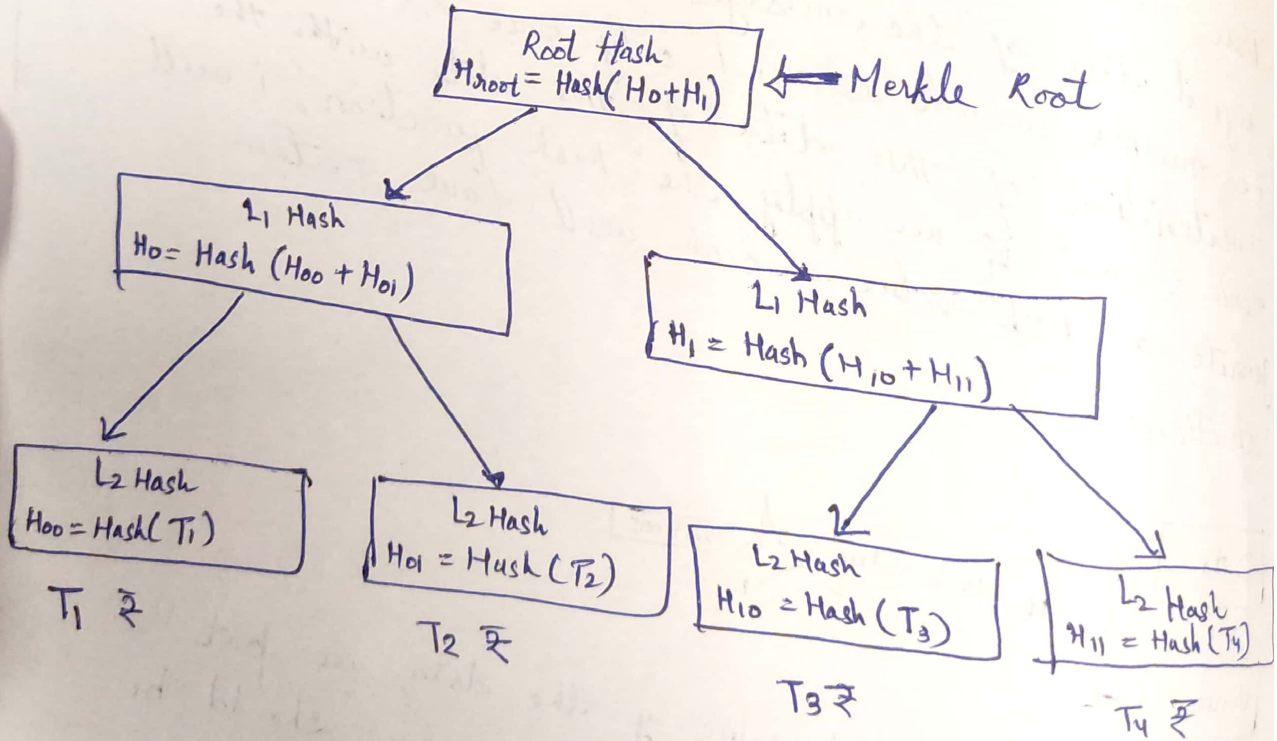
- If we put restrictions in a proper way then it will be computationally challenging to recalculate the hash values by the attacker

Example solving on site.

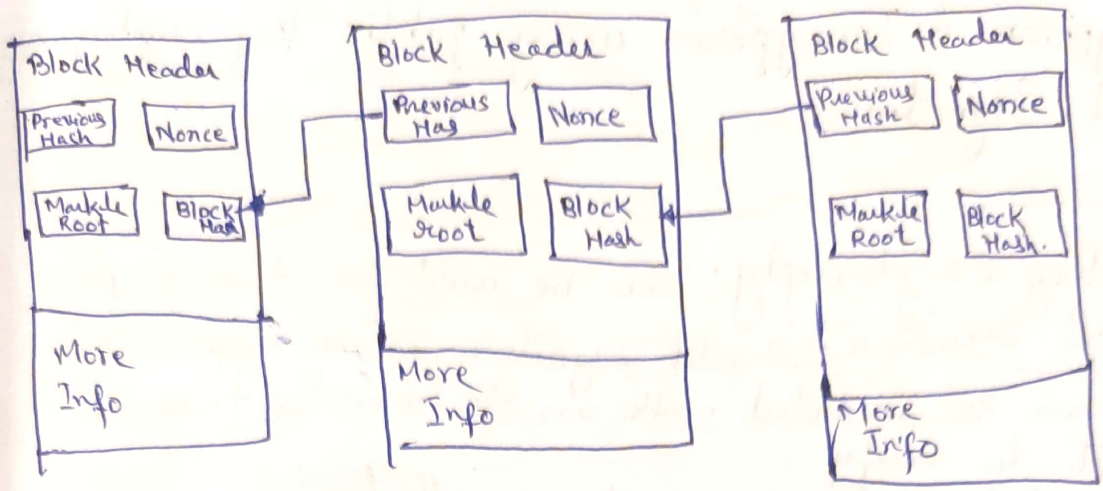
Detect Tempering from Hash Pointers – Hashchain



Merkle Tree – Organization of Hash Pointers in a Tree



Blockchain as a Hashchain



Week-2

① Basic concepts of cryptography
② Public key cryptography
③ Encryption & Decryption using public key cryptography.
④ Digital Signature.

---

* Public key Cryptography
* RSA.

Basic Concepts of Cryptography

→ Symmetric key Cryptography.
→ Same key is used for encryption & decryption.
- How to share the key securely
- Cannot address certain requirements.

→ Public key Cryptography / ASymmetric key cryptography.
- One key for encryption, one for decryption
- Handles several requirements like those in blockchain.

Lecture 6

- Basic concepts of cryptography
- Public Key Cryptography
- Encryption & Decryption using public key cryptography
- Digital Signature.

Public Key Cryptography: can be used for two purposes

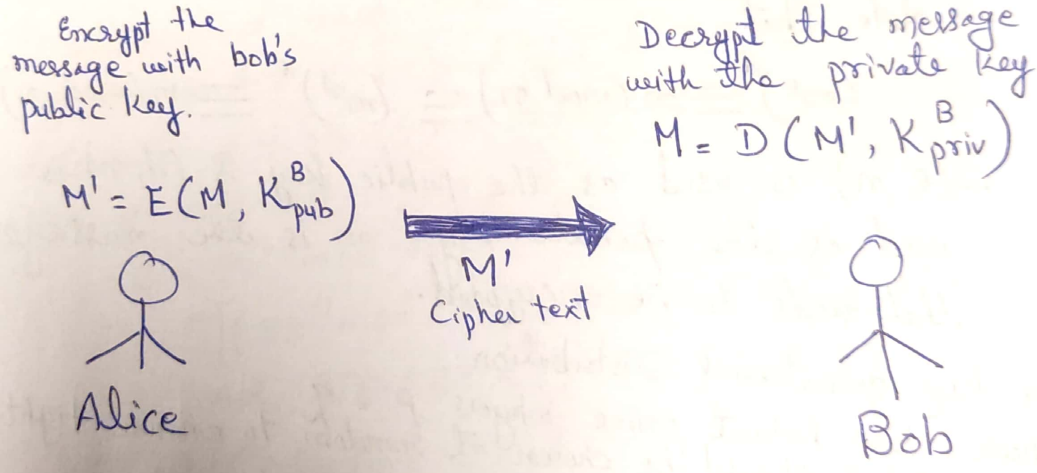
① Digital Signature: Digital signature is a digital code which can be included with an electronically transmitted document to verify
- The content of the document is authenticated
- The identity of the sender
- Prevent non-repudiation - Sender will not be able to deny about the origin of the document.

* Purpose of Digital Signature:
- Only the signing authority can sign a document, but everyone can verify the signature.
- Signature is associated with the particular document
  - Signature of one document can not be transfered to another document.

PKC: . Also known as asymmetrical cryptography or asymmetric key cryptography.
- Key: A parameter that determines the functional output of a cryptography algorithm.
  - Encryption: The key is used to convert a plain text to a cypher-text: $M' = E(M, K_1)$
  - Decryption: The key is used to convert the cypher-text to the original plain text $M = D(M', K_2)$

- Properties of cryptographic key (You need to prevent it from being guessed)
  - → Generate the key truly randomly so that the attacker cannot guess it.
  - → The key should be of sufficient length — increasing the length ~~may~~ makes the key difficult to guess.
  - → The key should contain the sufficient entropy, all the bits in the key should be equally random.

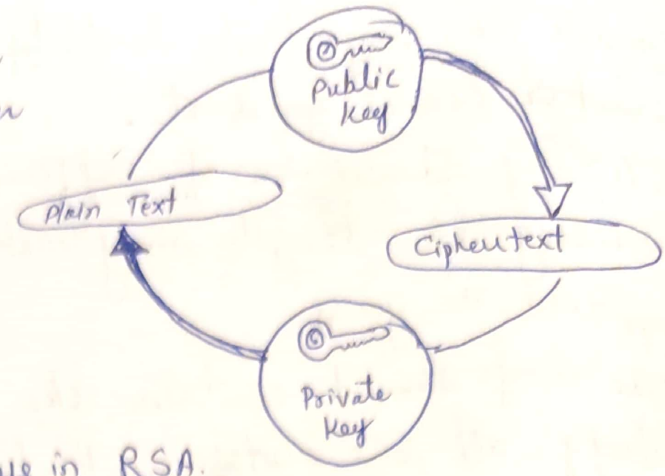
- In public key cryptography
  - ⇒ Two keys are used
    - i) Private key : Only Alice has her private key.
    - ii) Public key : "public" to everyone - everyone knows Alice's public key.



- There are many ways where public key cryptography is used. One of them is RSA
- RSA : (Ron) Rivest — (Adi) Shamir — (Leonard) Adelman. — Inventors of public key crypto systems.
  - → The encryption key is public & decryption key is kept secret. (private key).
    - → Anyone can encrypt the data
    - → Only the intended receiver can decrypt the data.

RSA algorithm

- Four phases
  1) Key generation
  2) Key distribution
  3) Encryption
  4) Decryption



- Public & Private Keys in RSA.
  → It is feasible to find three very large +ve integers e, d and n; such that modular exponentiation for integers $m (0 \leq m < n)$:

$$(m^e)^d \equiv m \pmod{n}$$

  → Even if you know e, n and m; it is extremely difficult to find d.

  → Note that

$$(m^e)^d \equiv m \pmod{n} \equiv (m^d)^e \equiv m \pmod{n}$$

  → (e, n) is used as the public key & (d, n) is used as the private key, m is the message that needs to be encrypted.

RSA Key generation & Distribution:

- Choose two distinct prime integers p & q
  - p & q should be chosen at random to ensure tight security.
- Compute $n = pq$; n is used as the modulus, the length of n is called the key length.
- Compute $\phi(n) = (p-1)(q-1)$ (Euler totient function)
- Choose an integer e such that $1 < e < \phi(n)$ & $\gcd(e, \phi(n)) = 1$; e and $\phi(n)$ are co-prime
- Determine $d \equiv e^{-1} \pmod{\phi(n)}$: d is the modular multiplicative inverse of $e \pmod{\phi(n)}$
  [Note $d \cdot e \equiv 1 \pmod{\phi(n)}$]

Lecture 7:

- RSA encryption & Decryption
- Digital Signature
- Hashing & Digital Signature

- RSA algorithm:

- Let m be the integer representation of a message M.
- Encryption with public key (e, n)

$$c \equiv m^e \pmod{n}$$

- Decryption with private key (d, n)

$$m \equiv c^d \pmod{n} \equiv (m^e)^d \pmod{n}$$

Example:

Key selection

- Select 2 prime numbers $p = 17$, $q = 11$
- Calculate $n = pq = 17 \times 11 = 187$
- Calculate $\phi(n) = (p-1) \times (q-1) = 16 \times 10 = 160$
- Select e such that e is relatively prime to $\phi(n) = 160$ and less than 1. Let $e = 7$
- Determine d such that ~~e is relatively prime to $\phi(n) = 160$~~ $d \cdot e \equiv 1 \mod 160$ & $d < 160$; Can determine $d = 23$ Since $23 \times 7 = 161 = 1 \times 160 + 1$

Encryption of plaintext $M = 88$

- $C = 88^7 \mod 187$

$$= [(88^4 \mod 187) \times (88^2 \mod 187) \times (88^1 \mod 187)] \mod 187 = (88 \times 77 \times 132) \mod 187 = 11$$

Decryption of Ciphertext $C = 11$

- $M = 11^{23} \mod 187$

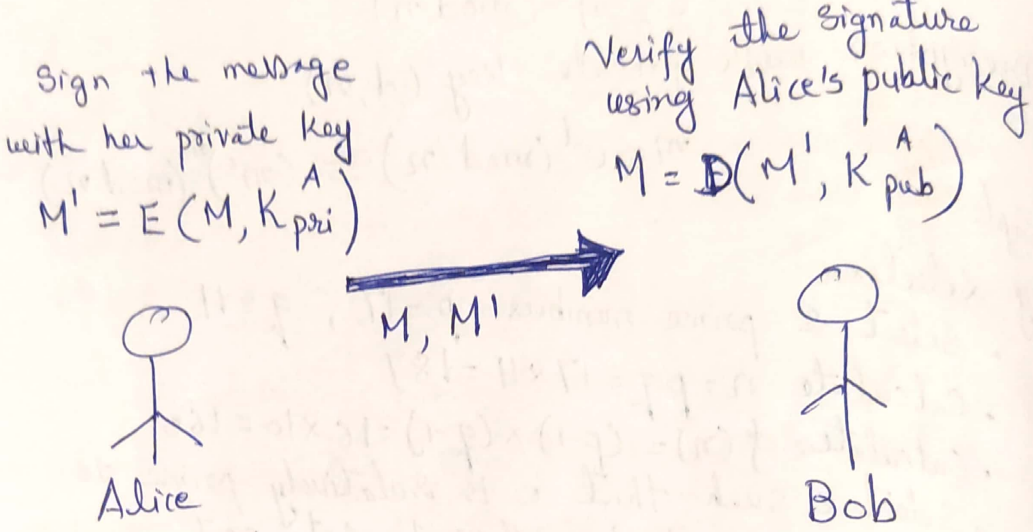
$$= [(11^1 \mod 187) \times (11^2 \mod 187) \times (11^4 \mod 187) \times (11^8 \mod 187) \times (11^8 \mod 187)] \mod 187$$

$$= (11 \times 121 \times 55 \times 33 \times 33) \mod 187 =$$

$$= (79720245) \mod 187 = 88$$

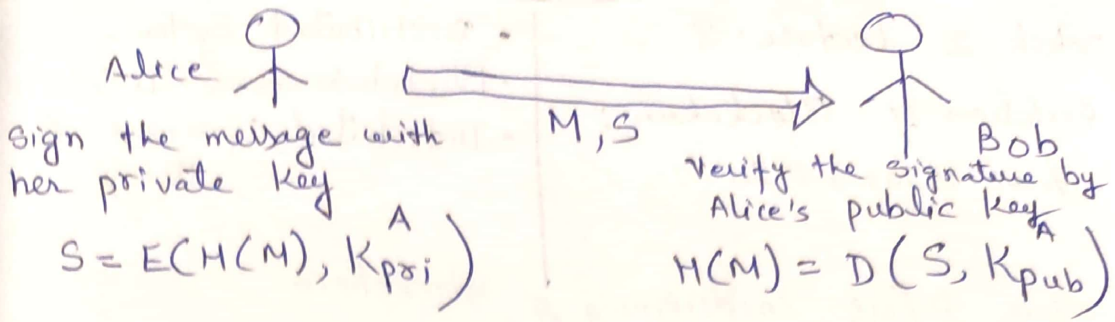
Digital Signature Using Public Key cryptography:

- Sign the message using the private key.
  - only Alice can know her private key.
- Verify the signature using the public key.
  - Everyone has Alice's public key & they can verify the signature.



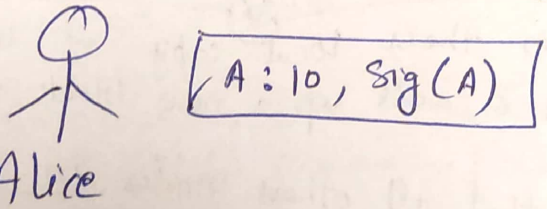
- Here we have to encrypt entire message M by private key of Alice & then we get M' & then it is to be decrypted & all we want to ensure that it must have been sent by Alice i.e the concept of digital signature & what else can be done is that instead of encrypting entire message m using Alice's private key, we generate the message digest for signing instead of the original message i.e we can reduce the signature size. i.e

  ⟹ Use the message digest to sign, instead of the original message.

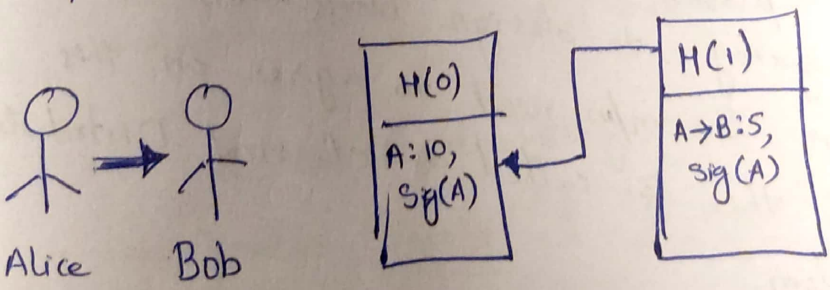


Digital Signature in Blockchain

- Used to validate the origin of a transaction
  - prevent non-repudiation
    - Alice cannot deny her own transactions.
    - No one else can claim Alice's transaction as his/her own transaction.
- Bitcoin uses Elliptic Curve Digital Signature (ECDSA) algorithm.
  - Based on elliptic curve cryptography
  - Supports good randomness in key generations.
- A cryptocurrency using Hashchain & Digital Sign



- Alice generates 10 bitcoins.
- Sign the transaction A : 10 using Alice's private key & put that in blockchain



- Alice transfers 5 coins to Bob
- Sign the transaction A — B:5 using Alice's private key & put that in the blockchain.

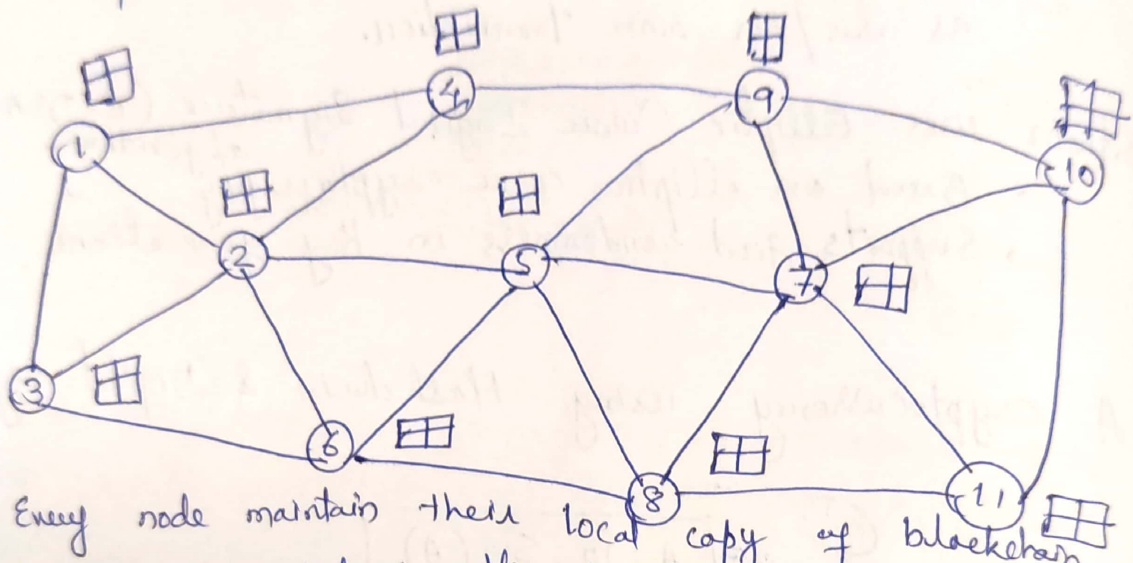
Week-2 Lecture: 8

Evolution of Blockchain:

- From 1970s & 1980s
  - Distributed Systems
  - Blockchain as a DS
  - Distributed Consensus - A History

- Three pillars in designing of Blockchain

- Core problem



- Every node maintain their local copy of blockchain.
- Now suppose Node 6 adds up a new block in the blockchain.
- I need to ensure that all others nodes in the n/w, they need to agree on this new block & to add the block in their local copy of the blockchain.
- This kind of problem that I have to solve while, I am going to design blockchain.
- Other nodes in the n/w need to agree on this new block & this is called as classical Distributed Consenses problem.

Distributed Consensus:

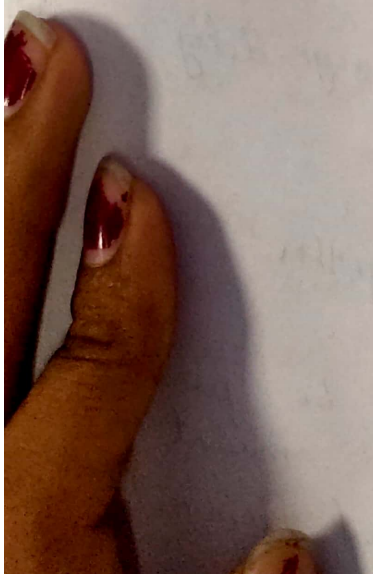



- Four want to go to burger shop or pizza.
- but collective decision.
- You are at home. & you want to decide.
- One will call to everyone.
- Relieng to one person.
- But do not trust anyone.
- How to reach a consensus.

Distributed Consensus – The literature:

- 1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson
  - Consensus is impossible in a fully asynchronous system even with a single crash fault. *
    - → Synchronous means messages are recieved in real time. (Phone call)
    - → Asynchronous means No baund on message delay (SMS)
    - * Node has stoped working.
  - Cannot ensure "safety" & "Liveness" together
    - "safety": correct process will yield the correct o/p
    - "Liveness": The o/p will be produced which a finite amount of time (eventual termination)
  - Safety has more priority than liveness.

- 1989 : Lamport started talking about Paxos.
  - Supports safety but not the liveness.
- 1990's : Everyone were confused about the correctness of Paxos.
- 1998 : Paxos got published in ACM Transactions on Computer Systems.
- 2001 : FLP impossibility paper wins Dijikstra Prize
  - People starts talking about Distributed sys.
- 2009. Zookeeper released.
  - Service for managing ditributed applications.
- 2010's onwards : Different types of consensus algorithms released
  - Multi- Paxos
  - Raft
  - Byzantine Fault Tolerance
  - PBFT



The Evolution of Cryptocurrencies:

- Cryptocurrencies – Requirements
- The evolution of cryptocurrencies
- Design goals for cryptocurrency Development.

Issues with Physical Currencies:

- Difficult to manage.
- Stolen

Cryptocurrency:

- An automated payment system having the properties
  - Inability of the third parties to determine payee, time, or the amount of payments made by individuals.
  - Ability to show the proof of payment
  - Ability to stop the use of payment media reported stolen.

- 1983: eCash by David Chaum
  - Money is stored in the computer – digitally signed by the bank.
  - Use a concept: blind signature, to make the payment anonymous – the contents of a message is "blinded" (disguised) before it is signed.

- 1989: DigiCash Inc. founded by David Chaum
  - ECash could not provide much additional benefit.
  - Not very popular among people – currency management overhead is more than bank notes.
- 1998: The company got bankrupted.

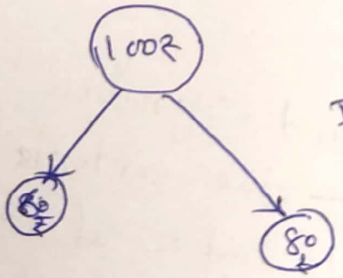
Morphing the Definition:

- An automated payment system having the properties
  - I
- 1998: Wei Dai publishes another anonymous, distributed electronic cash system called b-money.
- Nick Szabo describes "bit gold"
  - Participants solve a cryptographic puzzle that depends on the previous puzzle.
  - Some central control still needs to verify that the puzzle has been solved correctly.

- Can we verify the proof of puzzle solving in a distributed way?

→ Distributed Consensus

Majority agrees that the puzzle has been solved correctly.



. Consensus Over an Open Network
. Bitcoin – Open Blockchain Network
. The Success of Bitcoin as a cryptocurrency

→ The network is open
→ Needs the identity of others.
→ Works within a closed system.

Bitcoin Proof of Work: An open Consensus
2008: A whitepaper got floated on the internet by Satoshi Nakamoto.

- Need a good puzzle
- Solving the puzzle is complex but verification easy.

whenever someone is solving the puzzle during that time it is difficult to ~~go~~ solve.

- Best puzzle is given by hash function by including Nonce i.e

$$y = H(x \| N) \text{ i.e } (y, x) \rightarrow \text{Given find } N.$$

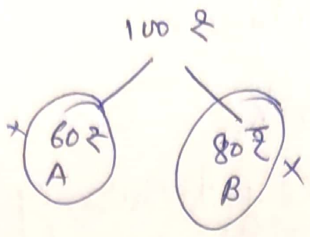
⇒ Bitcoin Proof of Work

- 2008: A whitepaper got floated on the internet.
  - Hash chain + puzzle solving as a proof (from Bitgold)+ Coin Mining in an Open P2P setup.
  - Proof of work (PoW) – Nakamoto Consensus.

⇓

⇒ Give more emphasis on "Liveness" rather than "Safety".

ie Participants may agree on a transaction that is the final one in chain.



One of two

What if two persons solve the puzzle simultaneously.

- Have not coined the term "Blockchain" in the paper.

2011 : Litecoin got introduced.

2015 : Ethereum network went live.

Sometime around 2016 : Term "Blockchain" got popular.

⇒ Classical distributed consensus can't be applied on the blockchain for cryptocurrencies.

- Open network can't support message passing.

⇒ Use puzzle solving to reach open consensus - used on Bitcoin.

⇒ But why should someone solve the puzzle?

- The puzzle is hard to solve, need computing power.

Week-3

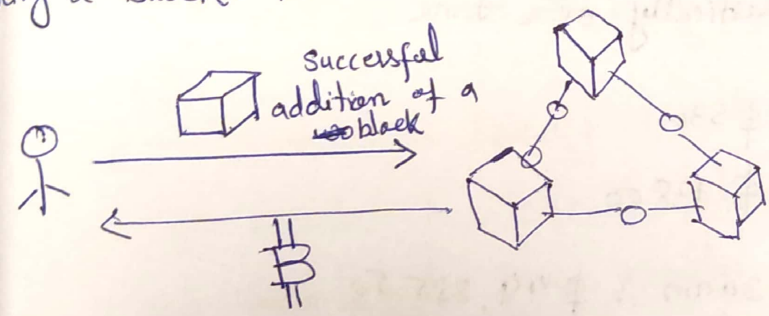
→ Bitcoin Mining.
→ The economic model of Bitcoin
→ Popularity of Cryptocurrency.

Bitcoin Mining: The Key to consensus.

. There are special nodes, called the Miners.
. Miners propose new blocks – solve the puzzle (find the nonce corresponding to a target block hash) & add the solution as a proof of solving the challenge to be the leader.
  . Solving the challenge needs some work to be done – proof of work (PoW)

Why someone would want to be the leader/Miner?
⇒ Earn Money (bitcoin)

Mining a Block: The reward



The economics behind Reward.

Encourage the community to participate in the mining through incentivization

Produces new Bitcoins in the system (Similar to a Mining new coins.)

The Bitcoin network works like a Reserve Bank to regulate the flow of Money (Bitcoin) in the market, but without explicit governance

Journey

Blockchain 1.0: Distributed Ledger.

Ledger: ⇒ where you can log the financial transaction

- Use of the Distributed ledger Technology (DLT) to design the "Money of the Internet" = Bitcoin & other cryptocurrencies.

- 3rd January 2009: Nakamoto mined the first block of Bitcoin n/w (called the genesis block)
  - 2013: Coinbase reported selling US$1 Million worth of Bitcoin.
- Many other cryptocurrencies have been evolved after that
  - Ethereum
  - Litecoin
  - ...

Prices of Bitcoin:

- Bitcoin value increased drastically over time
  - May 2010: < $0.01
  - April 2014: $340 - $530
  - December 2017: ~$13800
  - 24th Sept, 2021 10:30am: $44,285.50

Popularity of Cryptocurrencies:

Ethereum in 2018: Growing interest in developing decentralized Applications (Dapps)

What are Dapps:

- DLTs can contain information beyond financial transactions.
  - What about submitting an executable code as a transaction?
- Transactions gets executed == your code is getting executed.
  - And you have consensus

Smart contracts:

→ Smart contracts & automated code execution:
→ Permissioned Blockchain.

Smart contracts: Automatically Execute code over a Decentralized platform.

⇒ Automated execution of the code in transaction is called Smart contact.

⇒ Submit the anonymized (through public key ecryption) contract to a blockchain network.

⇒ Everyone in the n/w can see & validate the execution steps.

⇒ Cryptokitties → A popular Game on Ethereum.

Permissioned Model of Blockchain

- PoW (Nakamoto Consensus) works good in an open n/w.
  - But, transaction Latency is very high & throughput is low.
  - ~10 minutes in Bitcoin block commitment.
  - Few seconds to few minutes for Ethereum (depending on the cost that you pay)
- Can we think of any other BCA beyond cryptocurrency?
  - The high latency makes them unsuitable for most of the real time application.
- Many decentralized applications do not demand an open environment.
  - The food supply chain
  - Know Your Customer
  - Trade financing
  - ....

Blockchain 3.0

- Trustless Decentralization" over a closed n/w
  - Automatically transact assets among multiple organizations who do not trust each other.
  - Run smart contracts within a consortium of various organizations - the individual organizations know each other but do not trust each other.

Advantages:

- Go back to the classical distributed consensus protocols - low latency for commitment & high transaction throughput
- Use "Witness Cosigning" instead of "Proof Mining" for new block generation.
  - Classical distributed Consensus + Digital Signature

Permissioned Blockchain:

- The participants are pre-authenticated & pre-authorized
  - But they can still behave maliciously.
- Run Blockchain (& smart contracts) on top of this closed network
  - Ensure trusted computing among the participants.

Blockchain Elements 1:

What is Blockchain

Blocks in a Blockchain

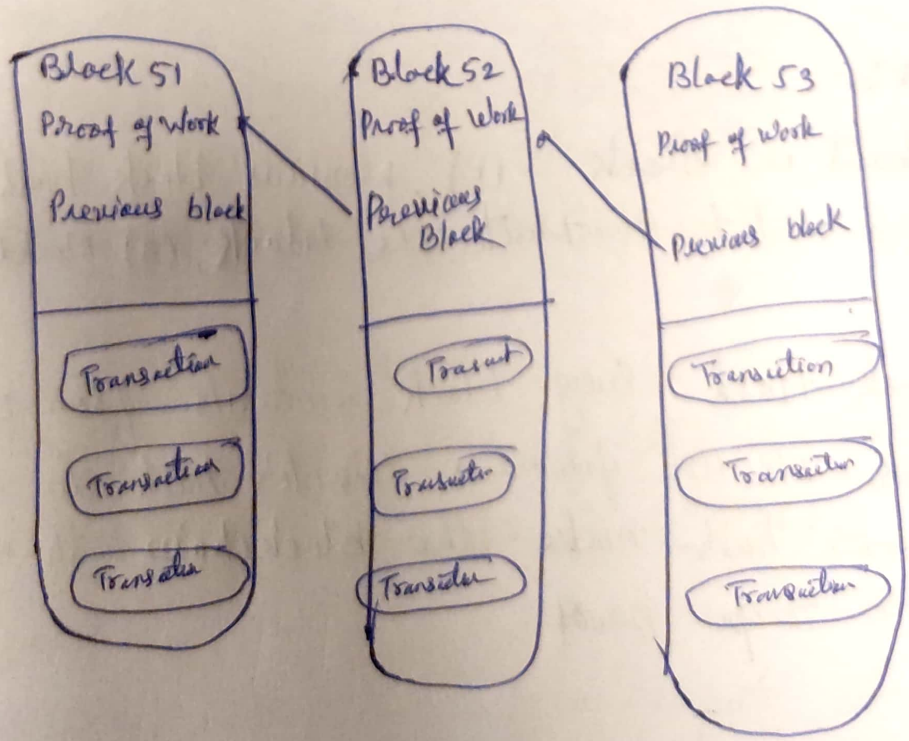
- Block Headers

What is Blockchain?

- A platform for executing transactional services.
- Spanned over multiple organizations or individuals who may not (need not) trust each other.
- An appened-only shared ledger of digitally signed & encrypted transaction replicated across a n/w of peer nodes.

The Block in Blockchain – Securing data cryptographically.

- Digitally signed & encrypted transactions "verified" by peers.
- Cryptographic Security - Ensures that participants can only view information on the ledger that they are authorized to see



Structure of Block:

- A block is a container data structure that contains a series of transactions.
- In Bitcoin: A block may contain more than 500 transactions on average, the average size of a block is around 1 MB (an upper bound proposed by Santoshi Nakamoto in 2010)
  - May grow upto 8 MB or sometime Higher (several conflicting views on this)
  - Larger blocks can help in processing large number of transactions in one go.
  - But longer time for verification & propagation

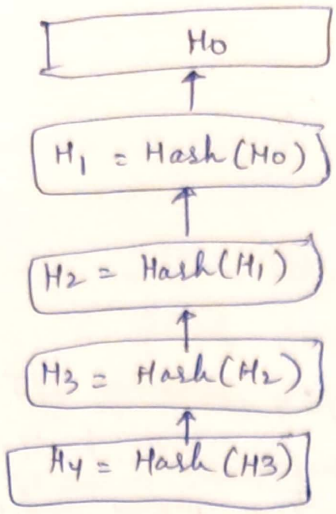
Structure of a Block

- Two components:
  - Block Header
  - List of Transactions

BTC.Com

Block Header:

- Metadata about a block – (1) previous block hash, (2) Mining statistics used to construct the block, (3) Merkle tree root.
- Previous block Hash: Every block inherits from the previous block - we use previous block's hash to create the new block's hash - make the block ~~chain~~ hash - make the blockchain temper proof.



Block Header:

Mining — the mechanism to generate the hash.

- The mechanism needs to be complicated enough, to make the blockchain tamper proof.
- Bitcoin Mining: $H_K = Hash(H_{K-1} \| T \| Nonce \| \text{something more})$ (Merkle tree)
- Find the nonce such that H_K has certain predefined complexity (Number of zeros at the prefix)

- The header contains mining statistics — timestamp, nonce & difficulty.

- Understanding Difficulty & Bits
- "Bits" written in Hex eg 0x17(index)0e2632 (coefficient)
  - First byte is index & next three bytes from coefficient.
  - Target = Coefficient * 2^(8*(index-3))

Hashes in a Block Header ~~— the~~

- Block identifier — the hash of the current block header (Hash algorithm: Double SHA256)
- Merkle Root
- Previous block hash is used to compute the current block hash.
- Timestamp, Previous hash, Markle root, Difficulty Bits, Nonce & version used to compute current hash.

Blockchain Elements - II

→ Block Generation cost
→ Transactions in a Block
→ Bitcoin Scripts.

Block Generation Cost

- Energy efficiency ~0.098 J/GH = ~100 J/TH
- ASIC H/w for bitcoin can perform about 750 TH/s
- Hash rate approx. 120M TH/s! Many actually go waste
- N/w consumes about 80 TW-hours of electricity annually. Figures vary between sources & are some form of estimates.
- Average household in Germany of four people consumes approx. 4,000 kW-hours of electricity per year.
- Can power about 20,000 households.
- Concept of pooling is used.
- What ensures tampeproof operation in terms of honest nodes?

Blockchain Replicas

- Every peer in a Blockchain n/w maintains a local copy of the Blockchain.
- Size is just about 351 GB.
- As a new user joins the network, she can get the whole copy.

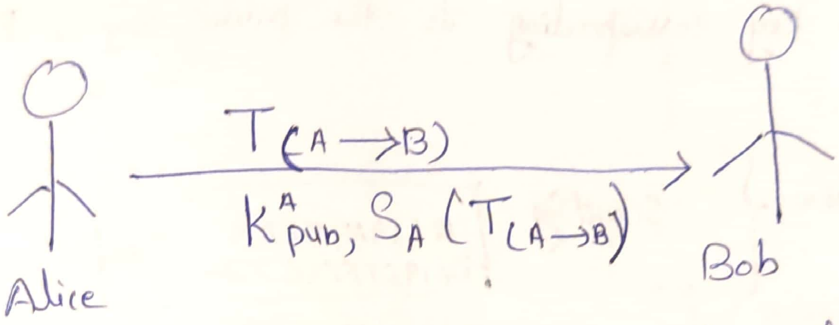
Requirements:

- All the replicas need to be updated with the last mined blocks.
- All the replicas need to be consistent - the copies of the blockchain at different peers need to be exactly similar.

Transactions in a Block

. Transactions are organized as a Merkle Tree. The Merkle Root is used to construct the block hash.
. If you change a transaction, you need to change all the subsequent block hashes.
. The difficulty of the mining algorithm determines the toughness of tampering with a block in blockchain.

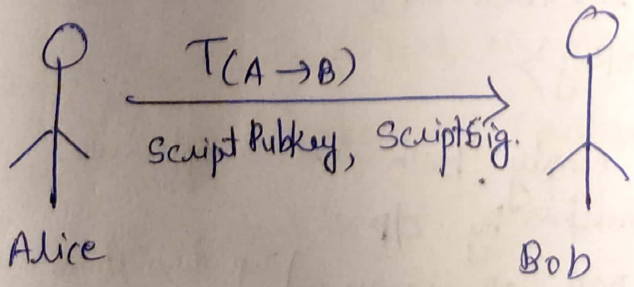
Bitcoin Scripts



How Bob will verify that the transaction is actually originated from Alice.

1) Send the public key of Alice along with the signature. → Bob verify this.

2) Bitcoin indeed transfers scripts instead of the signature & the public key.

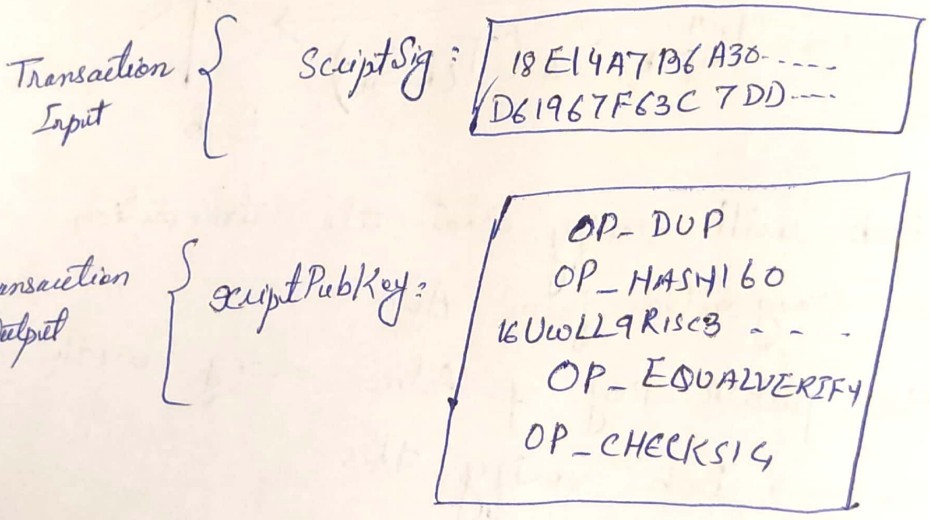


3) Bob can spend the bitcoins only if both the scripts return True after execution.

Bitcoin Scripts

- Simple, compact, stack based & processed from left to right
- FORTH like language
- Not ~~Turing~~ Turing complete (no loops)
- Halting problem is not there.

- With every transaction Bob must provide
  - A public key that, when hashed, yields the address of Bob embedded in the script.
  - A signature to provide ownership of the private key corresponding to the public key of Bob.



Script PubKey: OP_DUP
OP_HASH160 <pubkeyHash>
OP_EQUALVERIFY OP_CHECKSIG.

scriptSig: <sig> <PubKey>

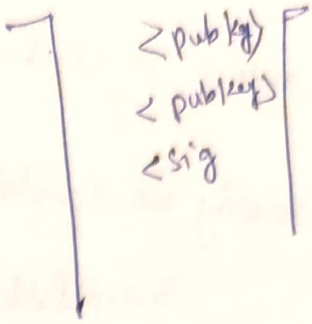
- The stack is initally empty. Both the scripts are combined - input followed by o/p.

<sig> <PubKey> OP_DUP OP_HASH160 <PubkeyHash> OP_EQUALVERIFY OP_CHCKSIG.

Top two items are pushed to stack one after another.

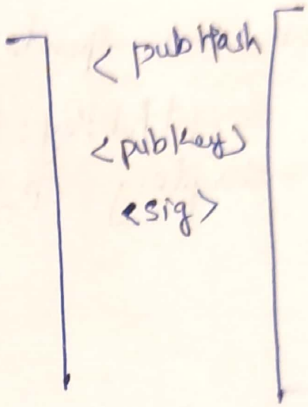
OP_DOP → Duplicating the top of the stack.

<pubkey>
OP_HASH160 <pubkey Hash>
OP_EQUALVERIFY OP_CHECKSIG.

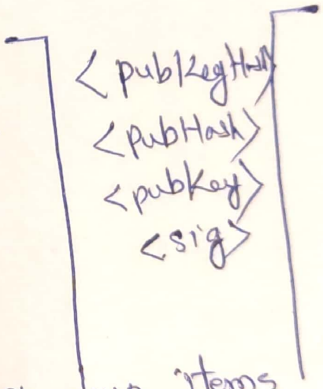


OP_HASH160 <pubkeyHash>
OP_EQUALVERIFY OP_CHECKSIG
Top stack item is hashed.

<pubkeyHash> OP_EQUALVERIFY
OP_CHECHSIG.



<pubkeyHash>
OP_EQUALVERIFY OP_CHECKSIG.
The constant is pushed in the stack.



P_EQUALVERIFY OP_CHECHSIG
Equality is checked between the top two items in the stack.

OP_CHECKSIG.

1. In script Sig Bob has sent his public key to Alice.
2. Pub key is duplicated (pubkey)
3. then hash of pub key is generated <pubHash)
4. Then Alice generate hash of pubkey that she recieved <PubkeyHash>

OP_CHECKSIG

signature is checked based on the top two stack items.

- Provably un-spendable or prunable o/Ps

  ScriptPubKey: OP_RETURN
  (Zero or more ops)

- Anyone can spends o/Ps

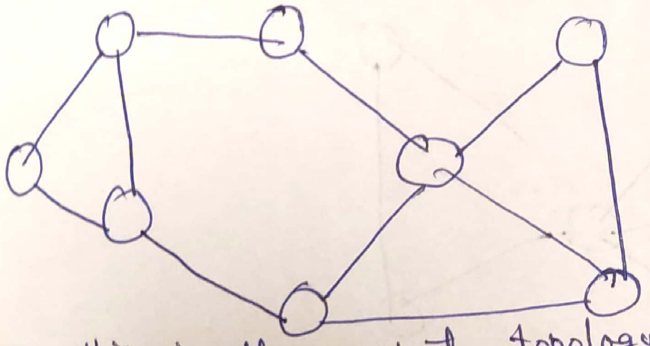
  ScriptPubKey: {empty}
  ScriptSig: OP_TRUE.

- Joining a Bitcoin N/w
- Transaction flooding
- Block Mining
- Block Propagation
- Forking & propagation of Longest chain

Bitcoin P2P Network

- An ad-hoc network with random topology, Bitcoin protocol runs over TCP
- All nodes (users) in the bitcoin n/w are treated equally
- New nodes can join anytime, non-responding nodes are removed after 3 Hours.

Joining in a Bitcoin P2P network:



Suppose this is the current topology of the bitcoin network. There are 8 nodes. Now assume it is your nodes which is simply a m/c with internet & has executed & installed scripts on it. Now you want to be the part of the existing P2P network.

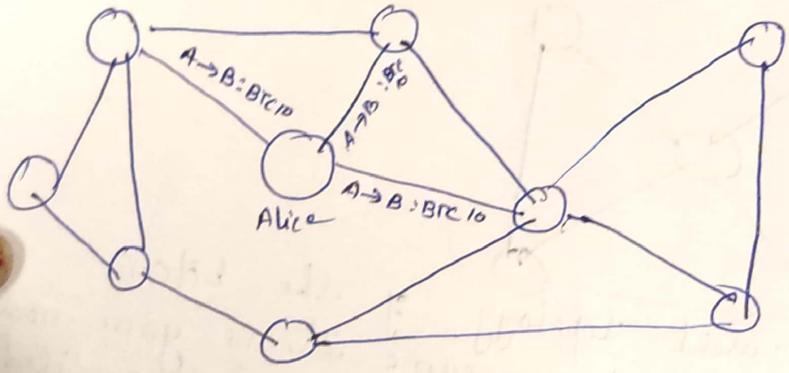
- First node will ask the seed node that give me an address for joining this particular n/w & each's seed node gets this request, it will send the address list.
  <address list>
- Every information is not sent to everyone. There are certain list of peers or neighbours to whome node can communicate.

- Now this new node have the information of these connected nodes.
- New node will ask to its neighbours to get most recent blockchain.
- Now the node is connected.
- Once this node is in the n/w it can carry on trans.

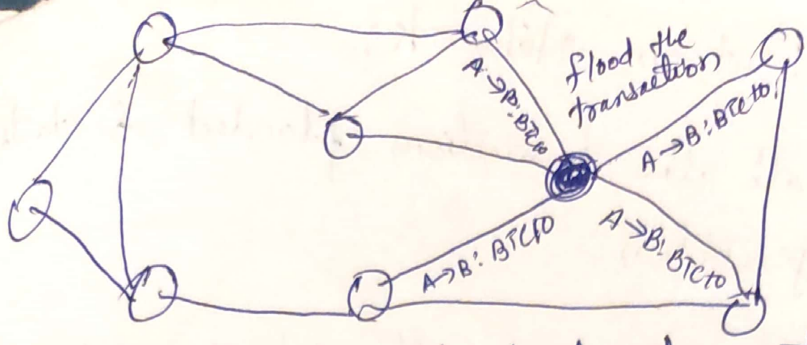
Example:

- Alice joins the Bitcoin network by opening her applet.
- Alice makes a transaction to Bob: A → B: BTC
- Alice includes the scripts with the transactions
- Alice broadcasts this transaction in the Bitcoin n/

⇒ Transaction flooding in a Blockchain/Bitcoin n/w.



- first Alice will flood the transaction to her neighbor nodes & then neighbour node will flood to it's neighbours & so on.
- So every such node who gets Alice's trasaction will validate the transaction so that they will check based on the script sig that alice had provided whether indeed it is a valid that everything is in place, Alice has signed it & it has been proved that indeed Alice had got those bitcoins which she is now spending.
- Now after validation, neighbouring node will now flood the transaction to its neighbours.



- There will be no duplicate transaction.

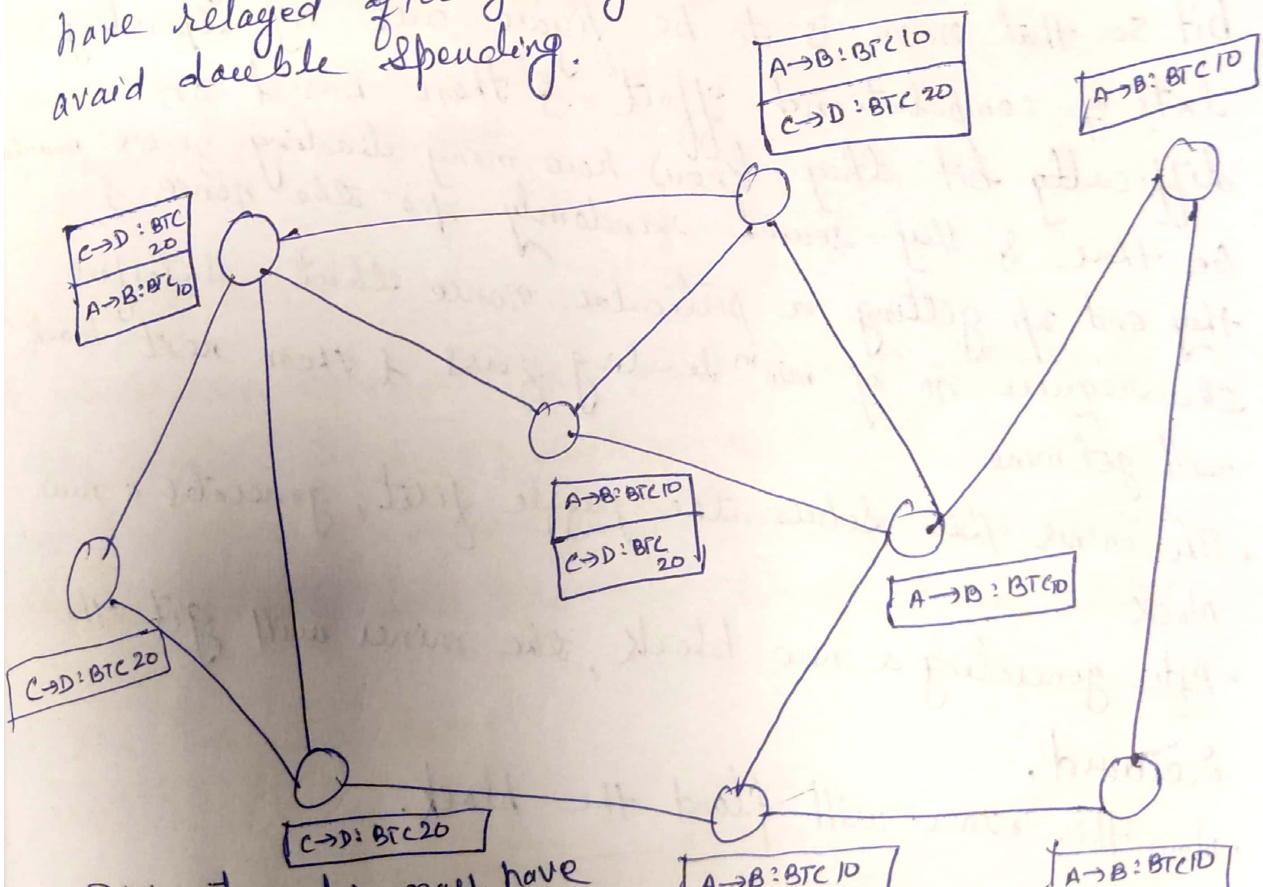
Which Transaction is valid with current Blockchain.

To the the transaction is valid with current blockchain.

- No conflict
- No double spending

- The Scripts matches with a pre-given set of whitelist scripts.
- Avaid unusual scripts, avaid infinite loops.

- Does not conflict with other transactions that I have relayed after getting the blockchain updated— avaid double spending.



- Different nodes may have different transaction pools.
- Accept the first transactions that you have heard.

Mining in a Bitcoin Network:

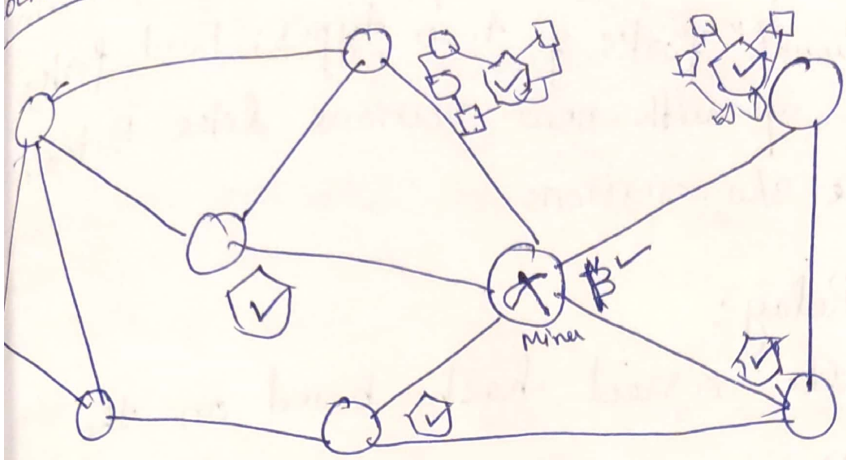
- Miners collects all the transactions flooded & starts mining i.e new block generation
- Not all the nodes are doing the job of mining.
- The nodes who have more computational power, they will do the job of mining.
- Mining means including a set of transactions and then from there including/creating markle tree out of those transactions & then once those are there you have to get the root hash of the markle tree & ~~if you~~ five other components including the nonce, the timestamp, the version, the previous has & the difficulty bit so that nonce is to be figure out by spending lots of computational effort & then based on difficulty bit they know how many leading zero's would be there & they search randomly for the nonce & they end up getting a particular nonce that satisfies the require no. of min^m leading zeros & then next block will get mine

The miner that solves the puzzle first, generates a new block.

After generating a new block, the miner will get the reward.

Now this miner will flood the block.

ock Flooding :

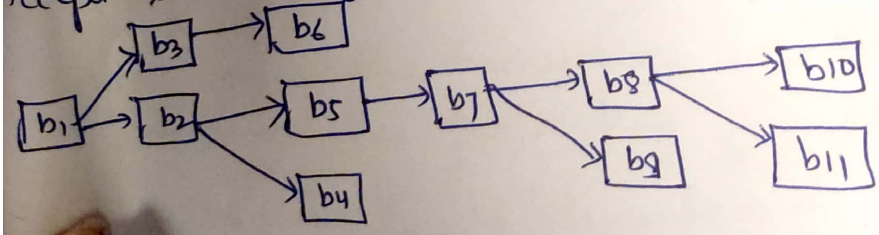


ed the blockchain with the new block included, added to
miner will flood these blocks which has been
blockchain & now it sends it to its peers & ~~they~~ then
y will be now adding it to their peers & so on, but
ore they add they will ensure that they check whether
block has been generated correctly the hash of this new
ck is matching with the required number of leading
os & that is the ~~step that~~.

lock Propagation :

ltiple miners can mine a new block simultaneously or
a near identical time i.e generation of a fork.
re are two pieces of reward that a miner node gets
en they successfully mine the new block to be added
the n/w

Accept the longest chain:



- "Accidental" forks occur rarely. Even if they occur eventually only one becomes part of the longest chain.
- There are "intentional" forks of two types: hard forks, soft forks to come up with new versions like Bitcoin Cash, etc to upgrade s/w versions.

Which Block to Relay:

- Block contains the correct hash based on the existing blockchain
- All the transactions inside the block are valid.
  - check the scripts
  - Validate with the existing blockchain.

The block is included in the current longest chain.

- Do not relay the forks.

Basics – Creation of Coins

Bitcoin
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 adjusted every 2016 blocks to aim for a constant two week adjustment period.
The last bitcoin will be mined in 2140 (estimated & unless changed)
Number of bitcoins generated per block is set to decrease geometrically, with a 50% reduction for every 210,000 blocks, or approximately 4 years.
This reduces with time the amount of bitcoins generated per block

- Theoretical limit for total bitcoins: Slightly less than 21 million
- Miners will get less reward as time progresses.
- How to pay the mining fee – increase the transaction fee.

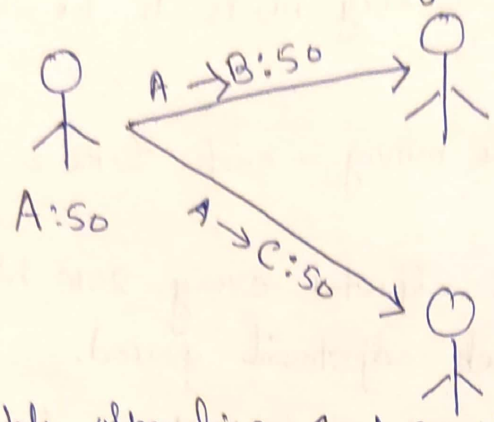
Sending Payments

Alice wants to send bitcoin to Bob

- Bob sends his address to Alice
- Alice adds Bob's address & amount of bitcoins to transfer in a "transaction" message
- Alice signs the transaction with her private key, & announce her public key for signature verification.
- Alice broadcasts the transaction on the Bitcoin network for all to see.

Double Spending:

- Same bitcoin is used for more than one transaction



- Double spending Cash ?? Not possible same note can't be
- In a centralized system for digital currency, the bank prevents double spending.
- How can we prevent double spending in a decentralized network?

Handle Double Spending Using Blockchain:

- When multiple valid continuation to this chain appear, only the longest such branch is accepted & it is then extended further (longest chain)
- Once a transaction is committed in the blockchain, everyone in the n/w can validate all the transactions by using Alice's public address.
- The validation prevents double spending in bitcoin.

Bitcoin Anonymity

- Bitcoin is permission-less, you do not need to set up any "account", or required any e-mail address, user name or password to login to the wallet.
- The public & 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.

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

Sample bitcoin address

1HYzmdJ22MKbJevpb3MBNpVcKjZHt89hz

person can have many such addresses, each with its balance

difficult to know which person owns what amount.

oins do not really "exist" as any tangible or sonic object.

re is no bit "coin" as you see in its logo.

ing a bitcoin simply means you have access to a air that includes

- A public key to which somebody else had sent some bitcoin.
- A matching private key that gives you the authority to send the previously received bitcoin to another address.

you lose your private key, you lose the corresponding oins).

ical Payment using Bitcoin:

that is needed is a (set of) private key(s) - public key n be generated from the private key.

fely store the private key - in your desktop, on the web, obile phone, special h/w attachment, printed on a iece of paper as QR.

or online payment, you can use the wallet & an ppropriate mode of applying the private key.

- for offline payments like in store payments or paying your friend, you can use your mob-phone to present the private key or use the hard copy !! as simple as using PayTm, Google Pay & so on.

Bitcoin Exchanges

- Trading bitcoin as commodity
- Centralized exchanges - (In India: WazirX, CoinDCX, Zebpay, CoinSwitch Kuber, etc).
  - Identity verification using KYC documents
  - Maintain your balance in Bitcoin & another currency like INR.
  - You set the buying & selling prices & quantities
  - If necessary, you can take the money out in a refered currency.
  - Some exchanges provide the payout option in anonymous prepaid cards.

There can also be decentralized exchanges with appropriate procedures for handling similar requirements.

Permissionless Model
Consensus Requirements for open n/ws
FLP Impossibility & Open consensus

The Permissionless Model: Basically works upon

- Open Network
  - Anyone can join in the n/w & initiate transactions
  - Participants are free to leave the n/w, & can join later again.

- Assumption: More than 50% of the paticipants are honest.
  - A society can not run if majority of its paticipants are dishonest!!

Many Permissionless Models are available.
One of the most prominant n/w is bitcoin n/w i.e cryptocurrency n/w.

- Ethereum: Ethereum supports decentralized application design. like gaming applications where large no. of paticipants can join

Consensus Challenges:

- Participants do not know each others.
  - Cannot use message passing!
- Anyone can propose a new block
  - Who is going to add the next blocks in the bc?
- The network is asynchronous.
  - We do not have any global clock
  - A node may see the blocks in different orders.

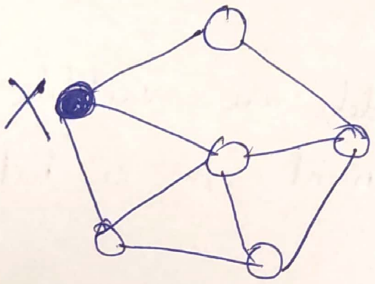
- Any types of monopoly needs to be prevented
  - A single user or a group of users should not gain the control – we don't trust anyone.

Synchronous Vs Asynchronous N/ws:

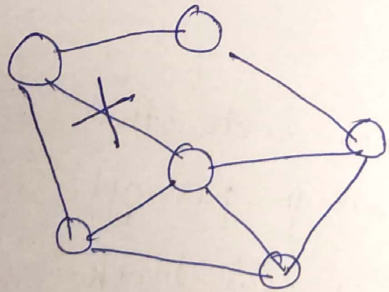
- Synchronous : I am sure I'll get the message in real time (theoretically no delay or minimum delay)
- Asynchronous : I am not sure whether & when the message will arive.

Failures in Network:

- Crash Fault: A node stops responding

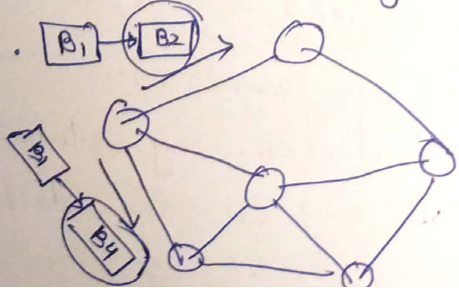


- Link fault (or Network Fault): A link fails to deliver the message



- Byzantine Fault: A node starts behaving maliciously

↳ Attack. B1 → B2



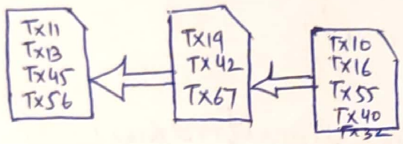
Remember FLP Impossibility?

- The impossibility theorem: Consensus is not possible in a perfect asynchronous n/w even with a single crash failure.
- Can not ensure safety & liveness simultaneously.

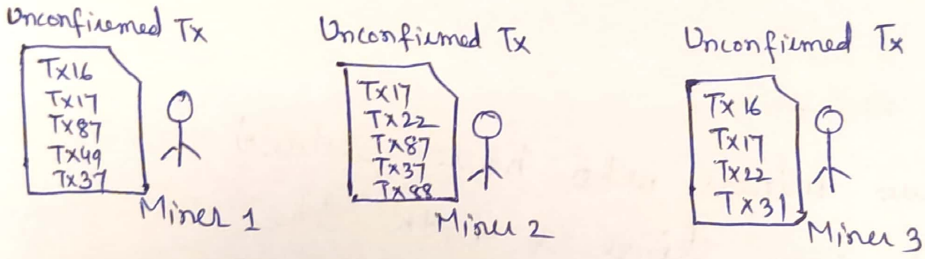
Nakamoto Consensus (PoW) Proof of Work

Liveness is more important than safety.

The Consensus Problem



Which one would be the next block?



Safety Vs liveness

Safety 1: The next block should be "correct" in practice

- Transactions are verified, block contains correct Hash & Nonce.
- The block mined by a miner is verified by all — that should be assured.

Safety 2: All the miners should agree on a single block.

- The new block of blockchain should be selected unanimously.

(Miners don't know each other) that's why PoW compromises here.

Liveness: Add a block as long as it is correct (contains valid transactions from the unconfirmed tx list) & move further.
(Two (or more) different miners may add two (or more) different blocks.

Consensus finality: Safety 1

Generate the Proof (Nonce)
- Generation: Complex
- Verification: Easy.

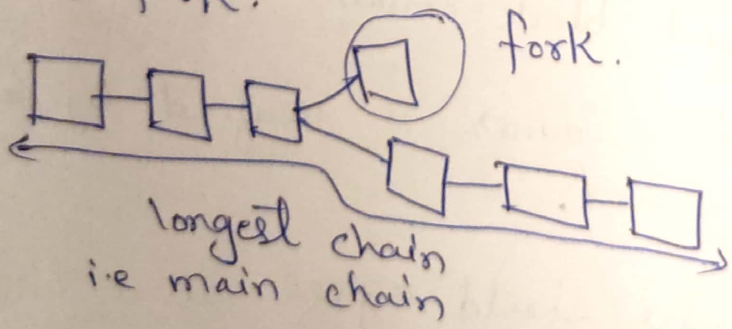
Limitations of PoW: Forking & Security

(What if two miners solve the puzzle simultaneously?)

Main chain

Fork.

If there are two miners, who have solved the puzzle at the same time & both the blocks are correct, initially add ~~two~~ both the blocks in blockchain i.e one will be treated as fork.



Momentary Decision: Miners remove the TXs corresponding to both the blocks, from their unconfirmed Tx list.

Forks are resolved eventually

- For the next block creation, a miner accepts the previous block that it hears from the majority of the neighbor. (Majority rule)
- For a forked block, if the transactions are not yet committed, include them in the unconfirmed Tx list.

Week-5

Open consensus beyon PoW

- Proof of Stake (PoS)
- Proof of Burn (PoB)
- Proof of Elapsed Time (PoET)

The Limit of PoW

- The Good : A fully decentralized consensus for permissionless models.
  - Works good for cryptocurrencies – serves its purposes.
- The Bad : Do not trust the individuals, but trust the society as a whole
  - You need a real large n/w to prevent the 51% attack – not at all suitable for enterprise applications.
- The Ugly : Low transaction throughput, Overuse of computing power!!
  - (Bitcoin) 3.3 to 7 transactions per second, (Ethereum) ~15 transactions per second.
  - Millions of miners – Thousand tries, but only one gets the success.

Bitcoin Energy Consumption:

Carbon Footprint
825.47 kg/Tx
Equivalent
to 137,578 hours of
watching Youtube

Electrical Energy
1737.82 kWh/Tx
Equivalent to power
consumption of an
average U.S household
over 59.56 days

Proof of Stake (PoS):

- Possibly proposed in 2011 by a Member in Bitcoin.

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.

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 effect on the attacker.
- Variants of "Stake"
  - Randomization in combination of the stake (Used in Nxt & BCoin)
  - 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 resources Vs Virtual/digital resources.

- PoB works by burning PoW mined cryptocurrencies.

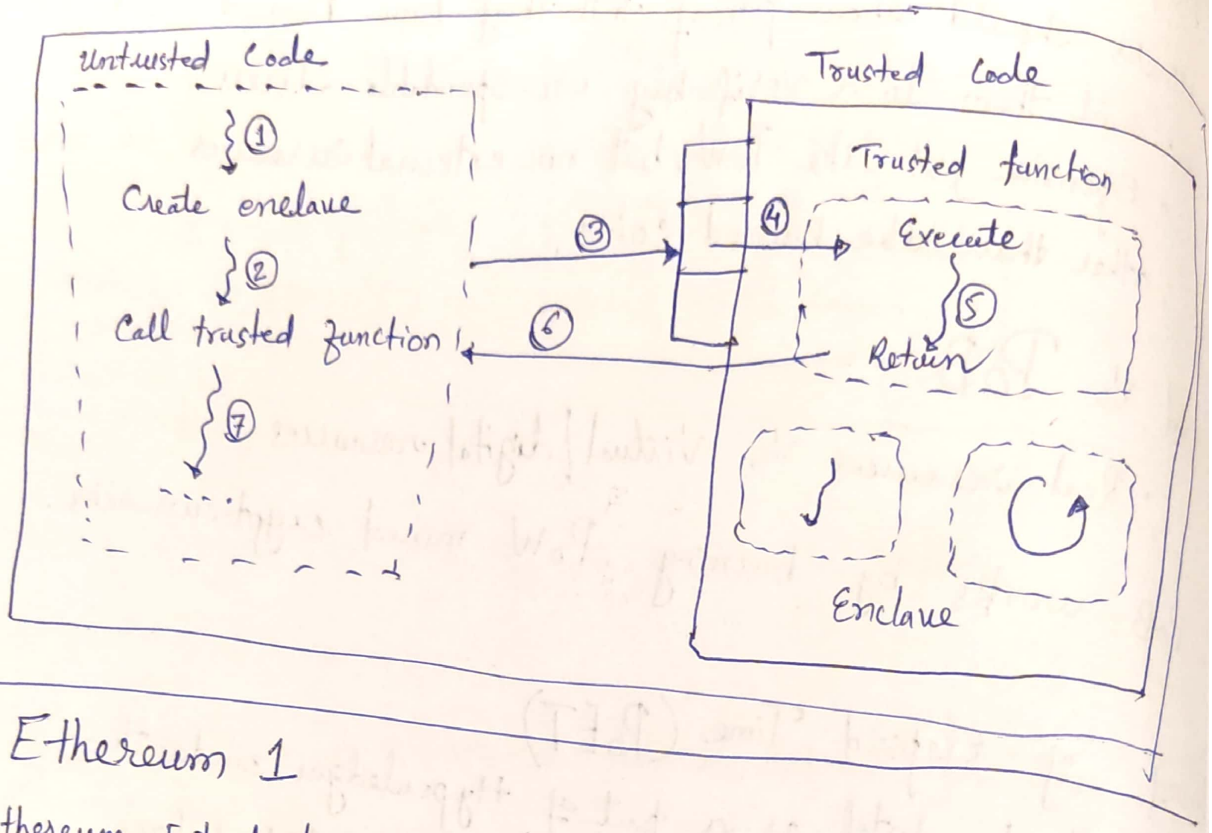
Proof of Elapsed Time (PoET)

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

- Basic Idea:
  - Each participant in the BC n/w waits a random amt of time.
  - The first participant to finish becomes the leader for the new block.

- How will one verify that the proposer has really waited?
  - 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 h/w provides an attestation that the trusted code has been set up correctly.

Intel SGX



Ethereum 1

- Ethereum Introduction
- Ethereum Network
- Go Ethereum
- Query Using RPC over HTTP

Ethereum:

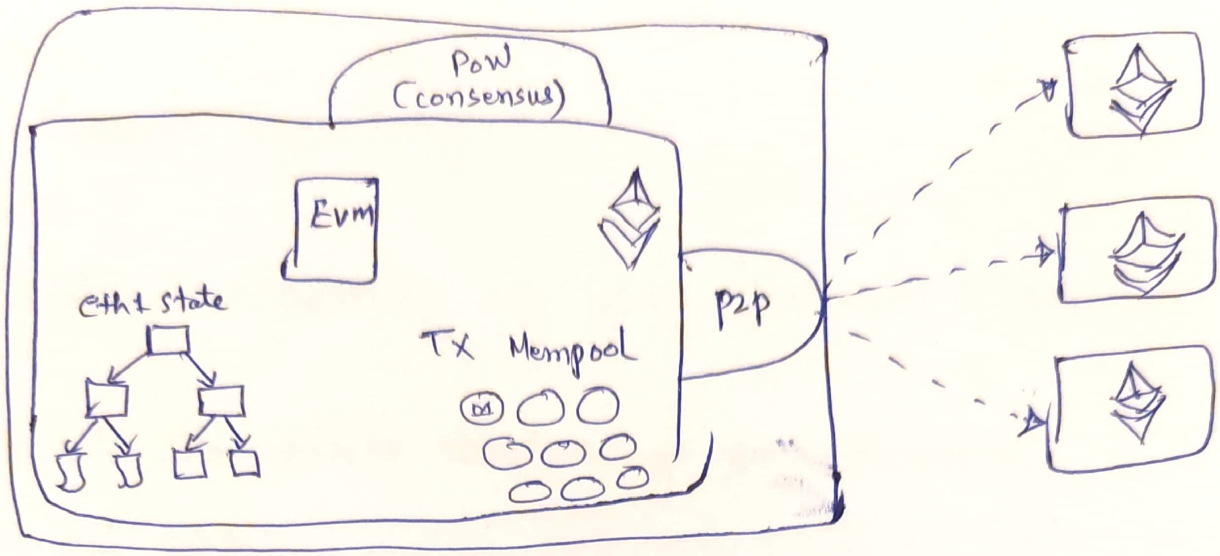
- Ethereum is a technology that lets you send cryptocurrencies to anyone for a small fee. It also powers the applications that everyone can use & no one can take down.

permissionless blockchain capable of executing smart contracts.

Ethereum Network:

- Distributed network of computers, known as nodes that can verify blocks and transaction data.
- An application, known as a client, running on your computer is a node.

Ethereum Network



Installing Geth

for Ubuntu

1. Add ethereum repository:

~~sudo apt~~

Sudo add-apt-repository -y ppa:ethereum/ethereum

. Then install the stable version of go-ethereum:

sudo apt-gel update

udo apt-gel install ethereum.

eck whether Ethereum is properly installed or not.
ive command
th version

lanaging Ethereum Accounts:

LSAGE:

geth account command [command options] [arguments....]

MMANDS:

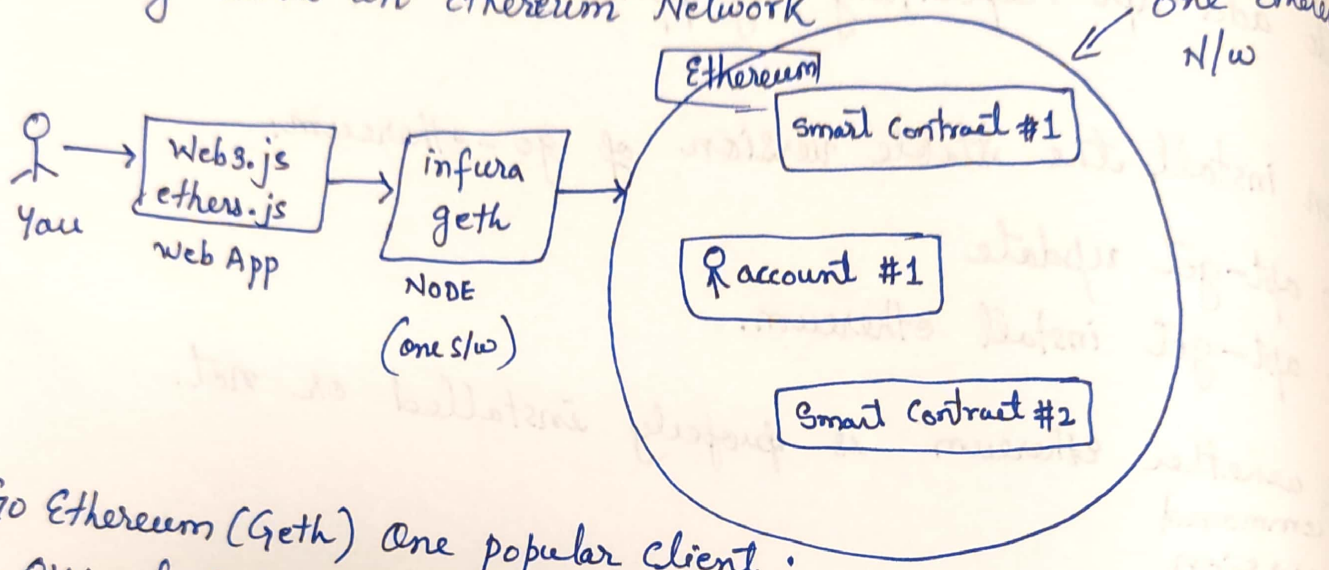
list	Print summary of existing accounts.
new	create a new account
update	update an existing account
import	Import a private key into a new account.

you have any account which exist previously, you can
port it into new account by import command.
ch account in Go Ethereum is password protected
nce you provide your password that account will be
ed into the Ethereum wallet.

Ethereum Mainnet

- No central server
- Independent nodes connected in a P2P n/w.

Interacting with an Ethereum Network



Go Ethereum (Geth) One popular client :

- Official Go implementation of the Ethereum protocol } Entry P Ethereu
- Main Ethereum CLI client
- Entry point into the Ethereum network
  - main, test, or private ~~test~~net
- Capable of running as :
  - full node (default)
  - Archive node (retaining all historical state)
  - Light node (retrieving data live)
- Provides JSON RPC endpoints exposed on top of HTTP, WebSock &/or IPC transports.

eth account list
no account presently.

eth account new // new account.
sword:
d password:
will get public key of the account that will be
d while transaction as well you will get the path of
r Ethereum account.

eth account list.
ount #0 // you will get & keystostore path
u seeing give command.
t - path (keystore path) wallet file.
// you will get public key & "crypto"

Connect to a network
Starting geth without any flag connects to the Ethereum
mainnet.
In addition to the mainnet, geth recognizes a few
estnets which you can connect to via the respective
gs:
- ropsten, Ropsten proof-of-work test network
https://ropsten.etherscan.io/
- rinkeby, Rinkeby proof-of-authority test n/w
https://rinkeby.etherscan.io/
- goerli, Goerli proof-of-authority test n/w.
https://goerli.etherscan.io/.

Test n/ws

Sync Modes:

⇒ You can start Geth in one of three different sync modes using the "--syncmode "<mode>" argum that determines what sort of node it is in the n/w:

- full: Downloads all blocks (including headers, transactions & reciepts) & generates the state of th BC incrementally by executing every block.
- fast: Downloads all blocks (including headers, Txns & reciepts) vaifies all headers & downloads th state & vaifies it against the headers.
- Snap (Default): Same functionality as fast, but with a faster algorithm.
- Light: Downloads all block headers, block d & vaifies some randomly.

Connecting to Goerli testnet:

geth --goerli --syncmode "light"
(two hyphens)

Interacting with Geth

- You can interact with Geth in two ways:
  - Direct with the node using the JavaScript conso over IPC.
    or
  - connecting to the node remotely over HTTP using

- IPC allows you to do more, especially when it comes creating & interacting with accounts, but you need direc access to the node.

c allows remote applications to access your node but
s limitations & security considerations.

ng RPC over HTTP

h --goerli --syncmode "light" --http

ery Balance

erying the balance of an account:
T request to the HTTP endpoint (default: 127.0.0.1:8545)

ON Payload:

"jsonrpc": "2.0",
"method": "eth_getBalance",
"params": [
"address", "latest" ← Time
],
"id": 1

rying using curl

T request using curl:

rl -X POST http://localhost:8485 \-H "Content-Type
lication/json" \--data '{"jsonrpc":"2.0","method":"eth_getBal
"params":[" ","latest"],"id":

o convert hexadecimal balance to decimal

echo $((16#hex no))

Ethereum Units:

* In Ethereum, the balance is represented in ethereum units called as wei

1 ether = 10^{18} wei

Unit	wei value	wei
wei	1 wei	1
Kwei (babage)	1e3 wei	1,000
Mwei (lovelace)	1e6 wei	1,000,000
Gwei (shannon)	1e9 wei	1,000,000,000
microether (Szabo)	1e12 wei	1,000,000,000,000
milliether (finney)	1e15 wei	1,000,000,000,000,000
ether	1e18 wei 10^{18} wei	1,000,000,000,000,000,000

Ethereum - II

Obtaining Ethereum for testnets
- Unlocking account
- Geth transactions using RPC using HTTP

---

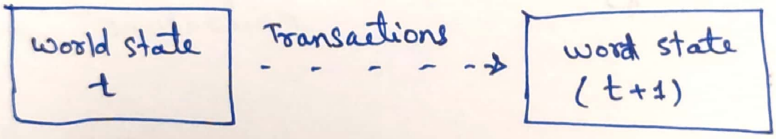
write something on Ethereum ledger

thereum Transactions:

Transactions are cryptographically signed instructions from accounts.

An account will initiate a transaction to update the state of the Ethereum n/w.

The simplest transactions is transfering ETH from one account to another.



To execute any transaction you need two things:
- Access to account private key (sign the transaction)
- Gas (fees)

Gas:

Each Ethereum transaction requires computational resources to execute.

Each transaction requires a fee

Gas refers to the fee required to conduct a transaction on Ethereum successfully.

- Gas is paid as amounts of Ethers.

Getting ether in testnet.

- for executing transactions in a testnet, we can obt ethers from faucet.
- Goerli faucet:
  - Social faucet: https://faucet.goerli.mudit.blog/
  - Simple faucet: https://goerli-faucet.slock.it/
- Rinkeby faucet: https://faucet.rinkeby.io/

*Get Ether in Goerli testnet:

- post a tweet containg ethereum account address.
- post tweet link in faucet.

  6 ethers / 1 day
  15 ethers / 3 days
  37.5 Ethers / 9 days

* Unlock Account:

- Create new account for goerli testnet, suppose password is set to "123"
- Write account password in a text file, here we name it pwd.txt.

```
geth --goerli account new
echo "123" > pwd.txt.
```

- Unlock account while starting geth

```
geth --goerli --syncmode "light" --http --allow-insecure-u
--unlock ____key____ --password pwd.txt
```

geth account list

geth--goerli account list.

geth --goerli account new.

ssword.

epeat password.

geth --goerli account list.

geth account list

copy keystore path.

th_sendTransaction:

OST SendTransaction

o send ether from account A to B.

t Prepare Transaction.

```
jsonrpc": "2.0",
method": "eth_sendTransaction",
params": [
{
 from": " Hex address ",
 to": " Hex address ",
 value": " 0x2386F26FC10000 "
                          ↖ 0.1 eth
}
]
 id":0
}
```

Submit Transaction

curl -X POST http://localhost:8545 \

View Transaction

Get Transaction by Hash

{ by : "eth_getTransactionByHash"

Query Block

: "eth_getBlockByNumber"

thereum - III

thereum application – DApps

sing web3.js to programmatically access Ethereum.

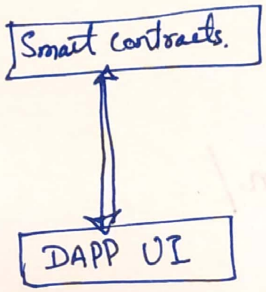
DAPPS in Ethereum

DAPPS - decentralized Application

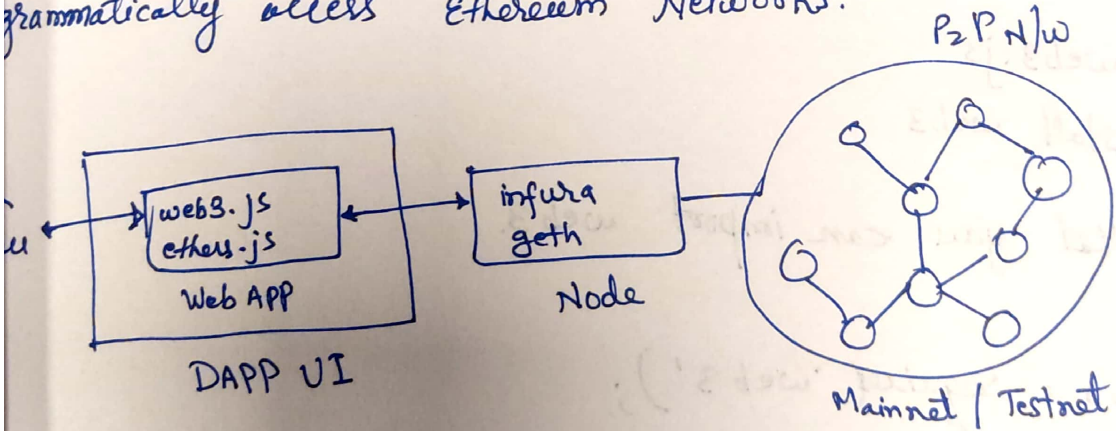
Application that is built for decentralized n/w like Ethereum.

Combines two components:

1. Smart Contracts
2. User Interface for executing transactions & contracts.



rammatically access Ethereum Networks.

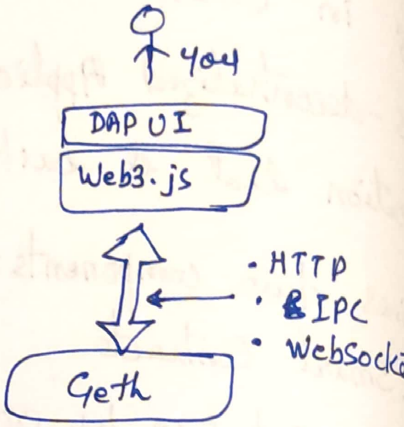


Web3.js

- Ethereum JavaScript API

  https://web3js.readthedocs.io/

- Collection of libraries that allow you to interact with a or remote ethereum nodes.



- It can connect using:
  - HTTP
  - IPC
  - WebSocket

Install Web3.js

- Install Node.js

  https://nodejs.org/en/

- Check node & npm

- Install web3.js

  npm install web3

- Verify that you can import web3.

  node

  ```
  var web3 = require('web3');
  console.log(web3.version)
  ```

- Goto Node.js site, Right click & copy the link
- goto terminal &

  wget paste address.

- copy name of compressed file.

  ls