Unit 2 Notes

Cryptography Blockchain

CRYPT- means 'hidden/vault'
GRAPHY- stands for writing

'***Cryptography is the study and practice of keeping secret information away from adversaries.***'
Includes:

1. hashing
2. public and private key pairs
3. digital signatures

- Bitcoin was initially proposed as a cryptography-based currency
- The idea of transferring value through a chain of digital signatures, which are like handwritten signatures

cryptography : method to encrypt your message
cryptonalysis : analysis on encrypted message

History

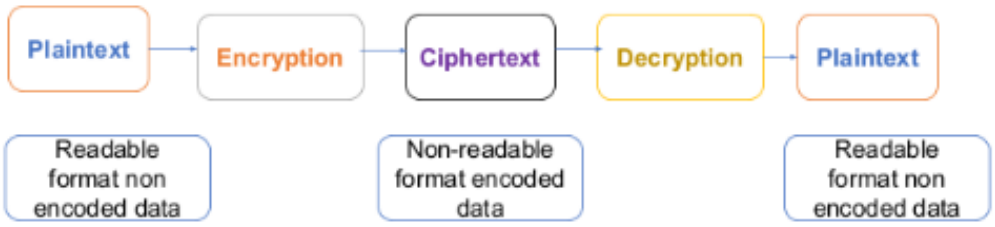
- The origin of cryptography- dated 2000 B.C with the Egyptian practice of hieroglyphics.
- The first known use of a modern cipher was by Julius Caesar
- A character would be shifted three positions ahead of it - **Caesar Cipher**

Features of Cryptography

1. **Confidentiality**
   - only target will receive information, secure info, cannot be understood by anyone for whom it was unintended
2. **Integrity**
   - data can't be modified, no data tampering without the alteration being detected
3. **Non repudiation**
   - Non-repudiation is the assurance that someone cannot deny the validity of something.
   - Sender of the information cannot deny at a later stage their intentions in the creation or transmission of the information

- **Authentication**
  - identity of sender/receiver verified before transactions

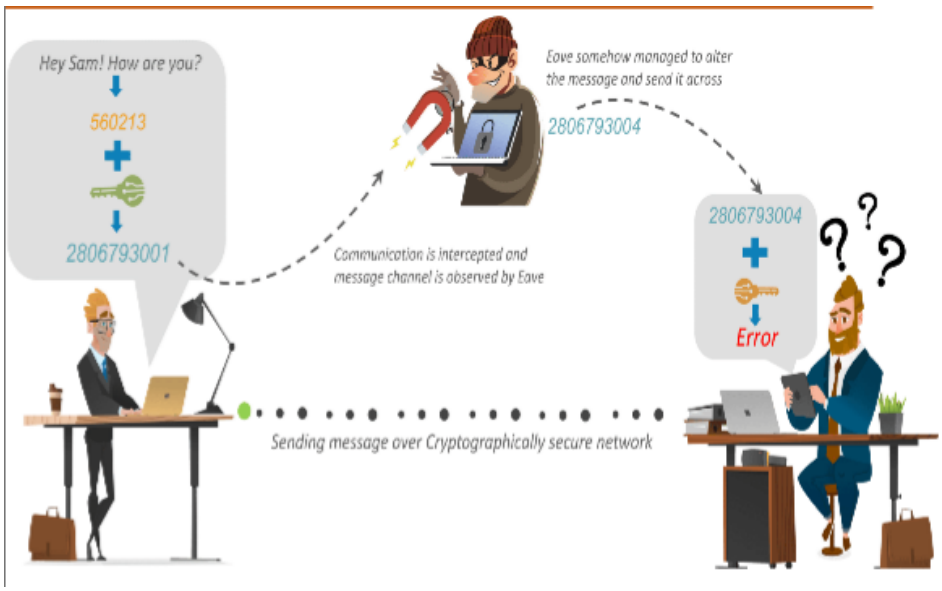
Cryptographic key components



- plain texts(original message)
- encryption(over original message)
- ciphertext(changed/hidden message)
- decryption(cipher to original)
- cipher(to encrypt and decrypt messages)

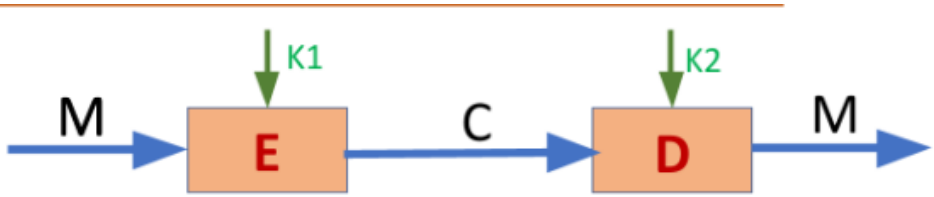
Vulnerabilities over a network

- A and B don't notice thief in middle



- To protect a message, a key is key to encrypt his message. Which convert his readable message (plain text) to unreadable form using a cipher
- In case of tampering of the message over a cryptographically secure network, one would require a key to decrypt the message. On decryption, it would produce an error revealing that the message was tampered with.

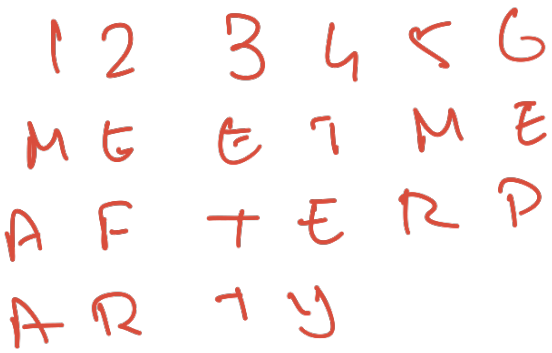
Cryptographic Ciphers



- **Single-key or Symmetric-key cryptography**
  - k1=k2=k
  - one key for encryption and decryption
  - ex AES, DES
  - AES is the successor to DES and DES3. It uses longer key lengths (128-bit, 192-bit, 256-bit) to prevent brute force and other attacks.
- **Public-key or Assymmetic-key cryptographic**
  - k1!=k2
  - different keys for encryption and decryption
  - RSA , Diffie-Hellman, ECDSA, DSA

Symmetric Key Ciphers

- Transposition Cipher
- Substitution Cipher
- Stream Cipher
- Block Cipher



1. TRANSPOSITION CIPHER

- Plain Text: MEET ME AFTER PARTY
- Key: 421635

1	2	3	4	5	6
M	E	E	T	M	E
A	F	T	E	R	P
A	R	T	Y		

4	2	1	6	3	5
T	E	M	E	E	M
E	F	A	P	T	R
Y	R	A		T	

- Cipher Text= TEMEEMEFAPTRYRAT

1234

TEY EFR MAA EP
ETT MR

Doesn't take care of spaces- How to decrypt this?

- Any spare spaces are filled with nulls or left blank or placed by a character

if read col wise
write row wise
if read row wise
write col wise

2. SUBSTITUTION CIPHER

PES UNIVERSITY

- Plain Text : Data Encryption
- Key word : 4

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

"D" will be replaced with the next 4th alphabet "H" in cipher text

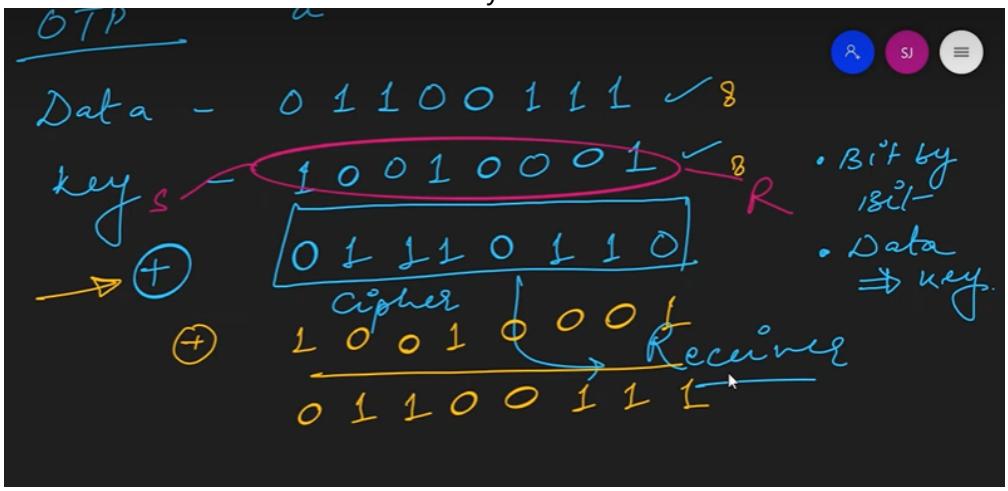
Decryption- Go backwards and select

- Cipher Text : Hexe Irgvctxmsr

- Shift by 'k' no of characters:
- To decrypt-go backwards
- Receiver should know the key shift

3. STREAM CIPHER

- Bit by bit conversion
- key should be the same size of the data(8 bits)
- performs a xor operation(same - 0, different - 1)
- 1 byte of data at a time example OTP/RC4
- The receiver and sender should have the same key

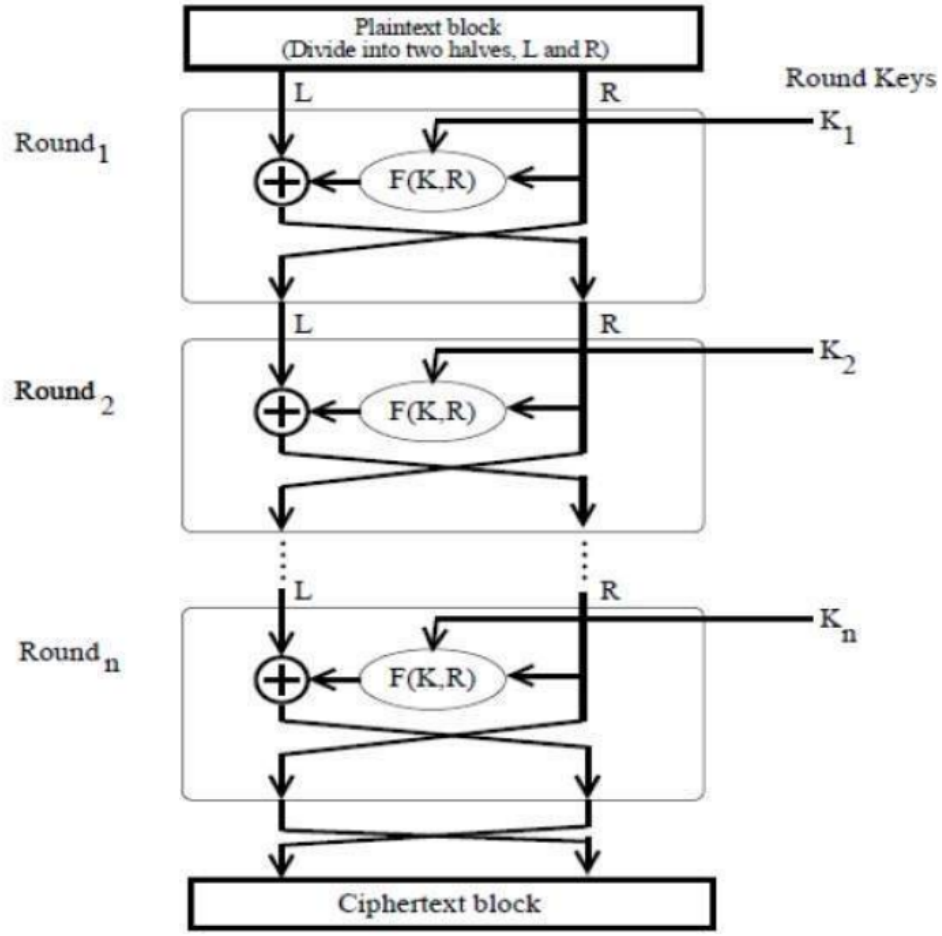


4.BLOCK CIPHER:

- In stream cipher, one byte(8 bits) is encrypted at a time while in block cipher ~64 bits or more are encrypted at a time.
- stream cipher is too small(a lot of overhead)
- *Feistel Cipher* model is a structure or a design used to develop many *block ciphers* such as DES.
- DES - feistal structure(16 rounds) block size 64- Proposed by IBM.
  - 56-bit key
  - initial 64-bit key is transformed into a 56-bit key by discarding every 8th bit of the initial key.
  - 8 bits acts as check bits only

- AES- encrypts data in block size 128bits each
  - The number of rounds depends on the key length as follows :
    - 128 bit key – 10 rounds
    - 192 bit key – 12 rounds
    - 256 bit key – 14 rounds

FIESTEL CIPHER



- Apply an encrypting function 'f' that takes two input – the key K and R. The function produces the output f(R,K). Then, we XOR the output of the mathematical function with L.
- Swap L and R for next round of permutation
- Key of each round can be different

Disadvantage of Symmetric key cryptography

(no of parties)C2

- 2 parties, then you have 1 key - 2C2
- 3 parties then you need 3 keys -3C2
- 4 parties then you need 6 keys - 4C2
- 5 parties then you need (4+3+2+1)= 10 keys - 5C2

- no of keys between parties will also increase quadractially very quickly

Asymmetric key cryptography

public and private encryption-decryption

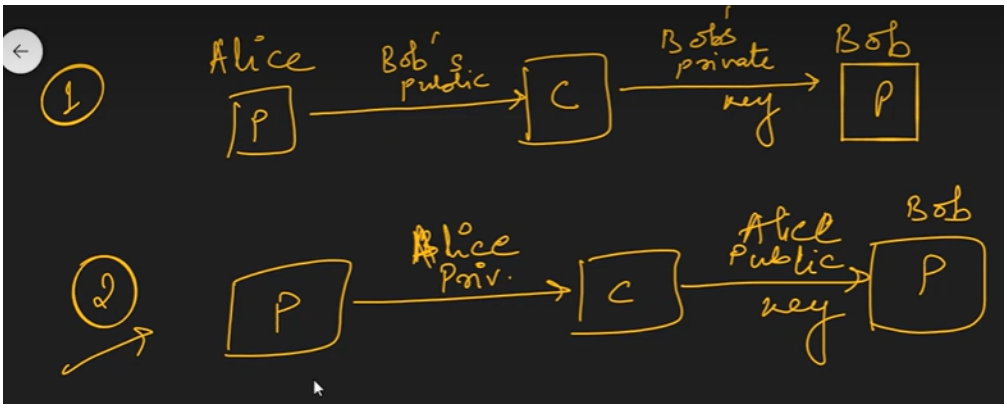
Public key cryptography

Approach 1:

- Alice encrypts with Bobs public key
- Bob decrypts with Bobs private key

Approach 2:

- Alice encrypts with Alice's private key
- Bob decrypts with Alice's public key



Problem with the first approach?

- Non-repudiation- how will bob know if this is a real message sent by Alice
- cause the senders key is not involved
- anyone can use bobs public key to encrypt - does not identify the sender of message

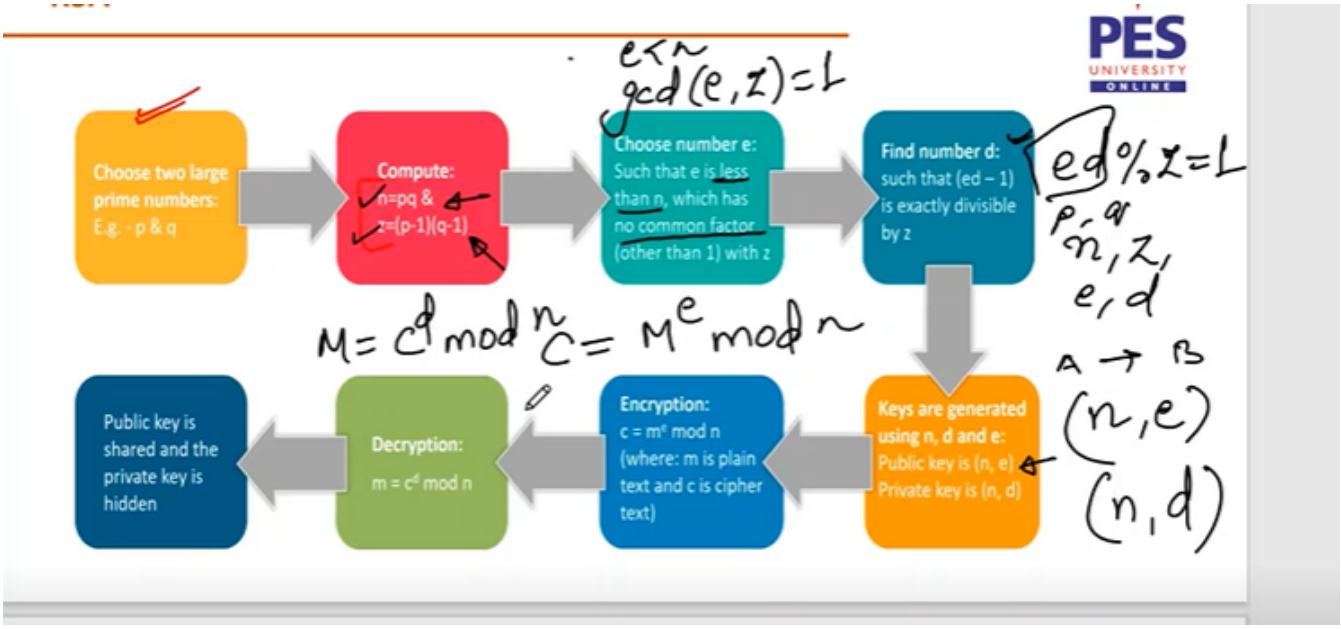
What is the problem with the second approach?

- Alice's public key is with everyone, so if anyone gets the cryptographic text, then anyone can decrypt

Hence what is the solution?

- Digital signature
- the sender has to sign(so that they can't deny that they didn't send it)

PES UNIVERSITY



Question:

Suppose Alice wants her friends to encrypt email messages before sending them to her. Computers represent text as long numbers (01 for "A", 02 for "B" and so on), so an email message is just a very big number. The RSA Encryption Scheme is often used to encrypt and then decrypt electronic communications.

Alice (n,d) Bob (n,e)

PUBLIC KEY= (n,e)
PRIVATE KEY =(n,d)

Alice's Setup:
Choose two prime number p and q
Let's take p=11 and q=3

- n = pq = 11 × 3 = 33
- z = (p − 1)(q − 1) = 10 × 2 = 20.

If e = 3 and d = 7, then ed = 21 has a remainder of 1 when divided by m = 20.

- Publish (n, e) = (33, 3).

Bob's Setup:
Bob encrypts message M = 14 :
• (n, e) = (33, 3).
When 14^3 = 2744 is divided by 33, the remainder is C = 5.
• Sends ciphertext C = 5 to Alice.

Alice's Setup:
Alice decrypts ciphertext C = 5:
• (n, d) = (33, 7).
When 5^7 = 78125 is divided by 33, the remainder is M = 14.
• M = 14 , the original message from Bob!

Question:

The Diffie Hellman algorithm: Alice and Bob have chosen prime value q = 17 and primitive root = 5. If Alice's secret key is 4 and Bob's secret key is 6, what is the secret key they exchanged? Explain.

n = 17
a = 5

Private key of Alice = 4
Private key of Bob = 6

Both Alice and Bob calculate the value of their public key and exchange with each other.

Public key of Alice
= 5^(private key of alice) mod 17
= 5^4 mod 17
= 13

Public key of Bob
= 5^(private key of Bob) mod 17
= 5^6 mod 17
= 2

Secret key obtained by Alice
= 2^(private key of Alice) mod 7
= 2^4 mod 17
= 16

Secret key obtained by Bob
= 13 ^(private key of Bob) mod 7
= 13^6 mod 17
= 16

both Alice and bob get the same value of secret key.
Therefore **common secret key = 16.**

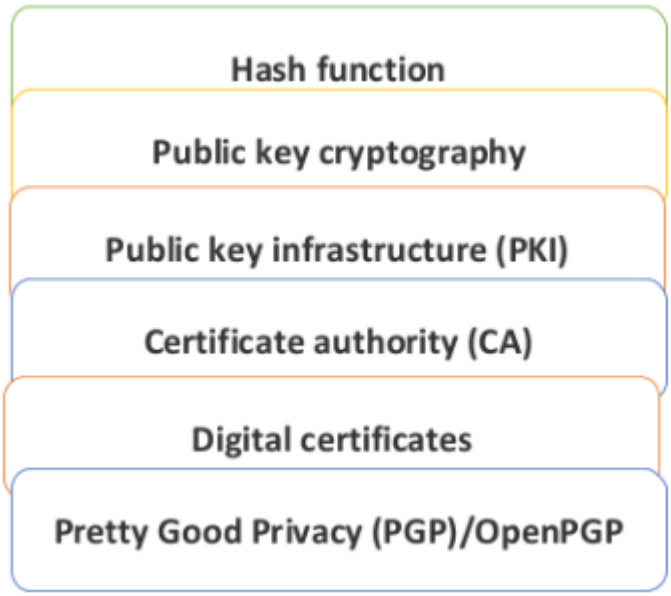
Digital Signature

- Electronic fingerprints
- Associates a signer with a document in a recorded transaction
- Digital signature uses a standard, accepted format, called Public Key infrastructure(PKI)
- History: signature recorded in Talmud(Roman usage of signatures)

Why Digital Signature

- To provide Authenticity , Integrity and non repudiation to electronic documents
- use internet as the safe and secure medium for Banjing, e-commerce and e-governance with security of servers

Key components of Digital Signature

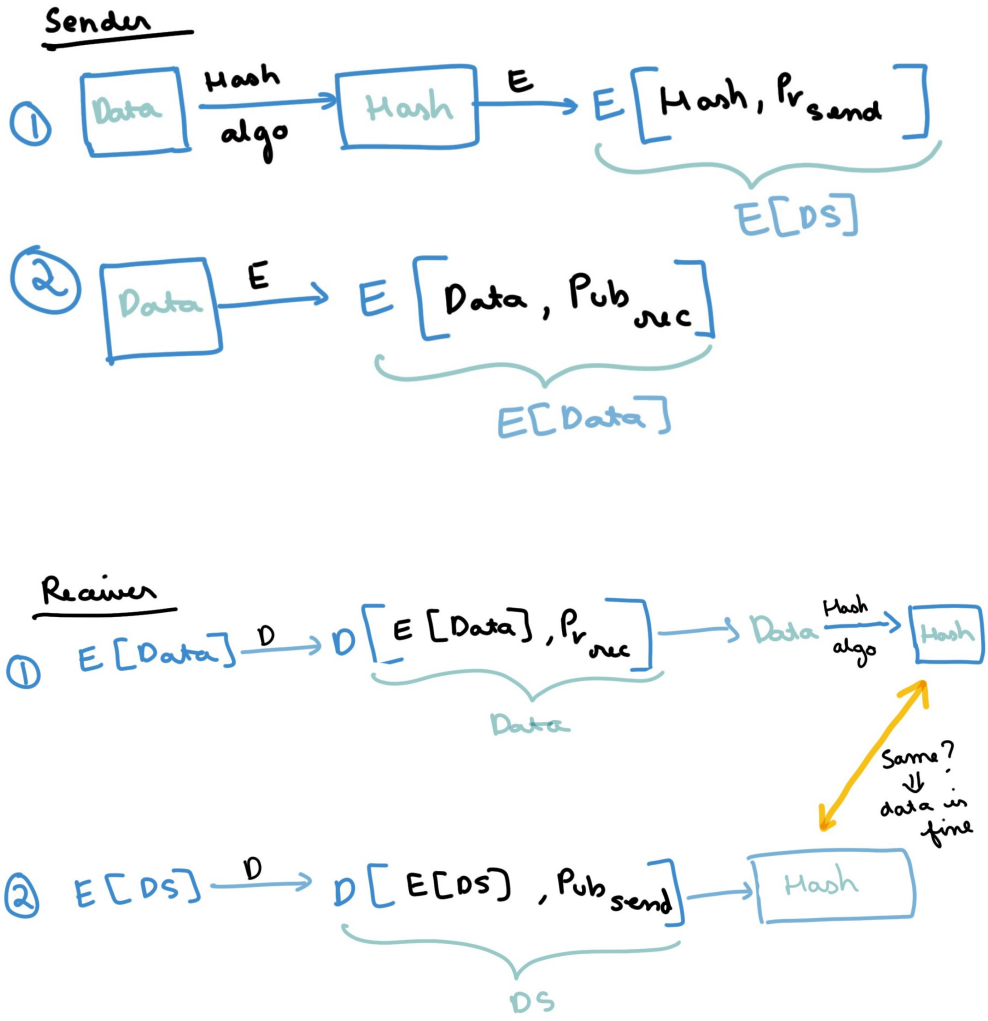


1. **Hash function**
2. **Public key cryptography** (2 keys involved)
3. **Public key infrastructure(PKI)** (responsible for making standards /policies to be followed during creation of the keys and key distribution across network- whatever needed to make it work in public key cryptography).
    - *A public key infrastructure (PKI) is a set of roles, policies, hardware, software and procedures needed to create, manage, distribute, use, store and revoke digital certificates and manage public-key encryption.**
4. **Certificate authority(CA)** (trusted party, validation of identity of users, generates public/private key pair, on validation of identity sends certificate

PES UNIVERSITY

5. **Digital Certificates** (CA issues these Digital Certificates) - identify the holder of a certificate. Digital certificates contain the public key and other information about individual
6. **Pretty Good Privacy(PGP)/Open GP** (have larger network , it is a protocol)
    - alternate to PKI
    - users “trust” other users by signing certificates of people with verifiable identities

Digital Signature for Blockchain



Digital signature in public key crypto

- Bitcoin uses **Elliptic Curve Digital Signature Algorithm (ECDSA)** Based on elliptic curve cryptography Supports good randomness in key generation

Application

- Electronic Mail
- Data storage
- Electronic funds transfer
- Software Distribution

- eGovernance Applications

Digital Signature Ensures

- Confidentiality
- Authentication
- Integrity
- Non-repudiation

Hash Functions

- Provide data security in a blockchain network
- A cryptographic hash function takes data and essentially translates it into a string of letters and numbers
- Bitcoin hashes to 256 or 64 characters -> 64* 4(hexadecimal)=256 bits
- Input can be extremely short or infinitely long, and you will get a unique output string of uniform length

Features of Hash sha-256bit 64 characters

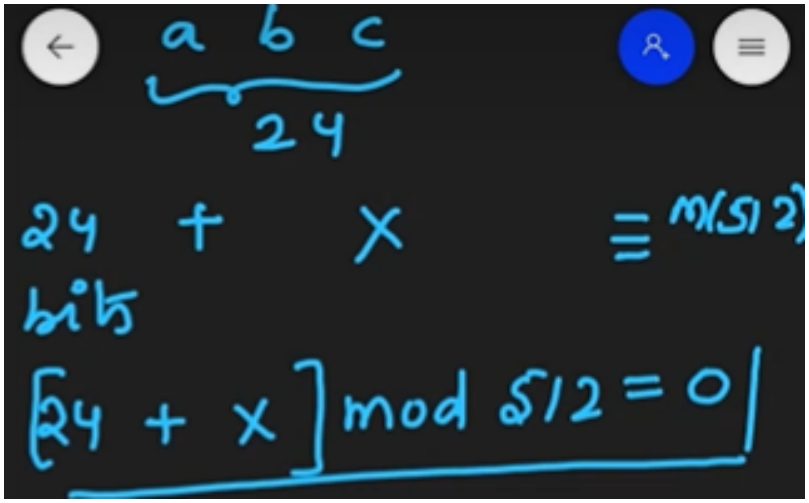
- Takes any input length string
- Produces fixed size output
- Easy to compute
- Almost impossible to reverse
- Collision resistant
- Hides original string
- Almost impossible to reverse pass
- collision resistant(no two string will have the same value of hash)
- Hides the original string
- Almost impossible to get original string from hash value
- Puzzle friendly

Popular Hash Function

- MD (Message Digest)
  - md5-128 bit hash. Taken out because collision resistance was broken after ~2^21 hashes
- SHA(Secure Hash Function)
  - 0(160bitsfunction)
  - 1(Secure sockets) Collision resistance broke after ~2^61.
  - 2(224,256,384,512(bitcoin))
  - 3(etheruem uses) depends on number of bits
- RIPEMD (Race integrity Primitives Evaluation Message Digest)

- Whirlpool 512 bits
- SHA 256: currently being used by bitcoin.
- Keccak-256: currently used by ethereum.

Hash function sha256



- SHA256 is used in Bitcoin mining – to construct the Bitcoin blockchain.
- Secure Hash Algorithm (SHA) that generates 256 bit message digest.

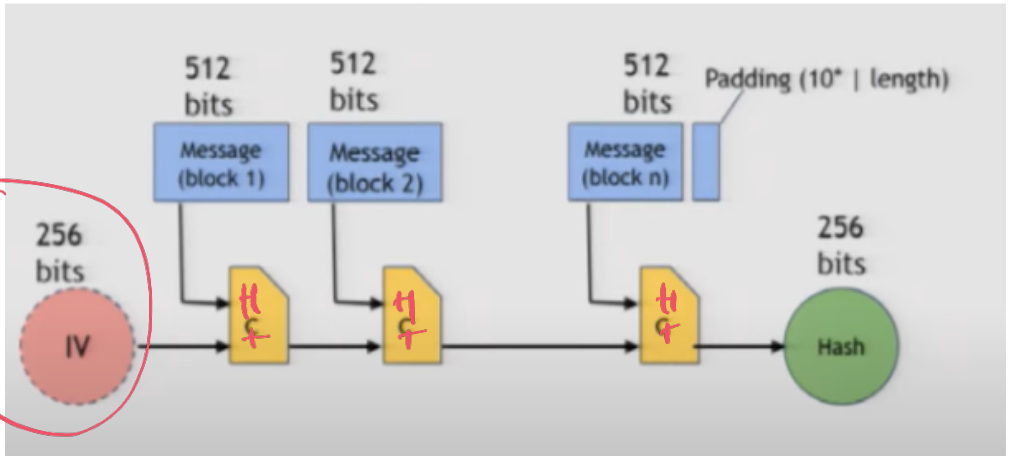
STEPS

$l+1+k+64 \equiv 512 \bmod 512$

- SHA256 Algorithm - Preprocessing
- Pad the message such that the message size is a multiple of 512
- Suppose that the length of the message M is l; and $l \ mod$ 512≠0
- Append the bit "1" at the end of the message
- Append k zero bits, where k is the smallest non-negative solution to the
- equation l+1+k≡448 mod 512 $512 - 448 = 64$
- Append the 64-bit block which is equal to the number l written in binary
- The total length gets divisible by 512 64 bits for len of msg.
- Partition the message into $\boldsymbol{N}$ 512-bit blocks $\boldsymbol{M(1)}$, $\boldsymbol{M(2)}$,..., $\boldsymbol{M(N)}$
- Every 512 bit block is further divided into 32 bit sub-blocks $\boldsymbol{M0}$ $(\boldsymbol{i})$, $\boldsymbol{M1}$ $(\boldsymbol{i})$,..., $\boldsymbol{M15}$ $(\boldsymbol{i})$ $\frac{512}{32} = 16 (0 \ldots 15)$
- The message blocks are processed one at a time
- Start with a fix initial hash value H (0)
- Sequentially compute H (i) = H $(i$-1) + Cm(i) (H $(i-1)$);
  - C is the SHA-256 compression function and + means mod 2^32 addition.
  - $H(N)$ is the hash of M.

$H(i) = H(i-1) + Cm(i)(H(i-1))$;

PES UNIVERSITY



Hash Function Properties

- Collision Free(n no of inputs, they don't have the same hash)
- Deterministic
- Pre-Image Resistance
- Small Changes in the input changes the hash
- Puzzle Friendly
- Quick Computation
- Information Hiding
- Message Digest

Collision Resistant Nature of Hash Function

- Hash functions are one-way; Given an x, it is easy to find H(x). However, given an H(x), **one cannot find x**
- It is difficult to fid x and y, where x !=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
- It maybe relatively easy to find collision for some hash functions

> If sender S hashes message and sends H(M) to R. If an attacker A can generate a Message M' which will generate the same hash H(M') = H(M). Then there is an collision

- **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, and 0.5 probability is reached with only ~23 people
- If a hash function produces N bits of output, an attacker needs to compute only 2^(N/2) hash operations on a random input to find two matching outputs with probability > 0.98
- For a 256 bit haash function the attacker needs to compute 2^128 hash operations- time consuming

Hash as a 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 the message digest
- size of digest is significantly less than the size of the original messages

Information Hiding

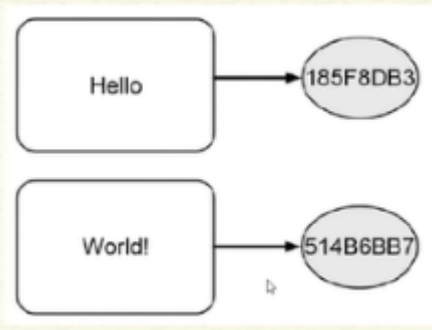
- Given H(x), it is "computationally difficult" to find x
- The difficulty depends on the size of the message digests
- Hiding helps to commit a value and then check it later
- Compute the message digest and store it in a digest store - commit
- To check whether a message has been committed, match the message digest at the digest store
- It is computationally expensive as multiple rounds of the hash is performed

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)
- A search puzzle (used in bitcoin mining)
  M and Z are given, k is the search solution
  Note: It might be exactly a particular value Z, but some properties that Z satisfies, i.e. Z could be a set of possible values
- Puzzle friendly property implies that random searching is the best strategy to solve the above puzzle

Patterns of hashing

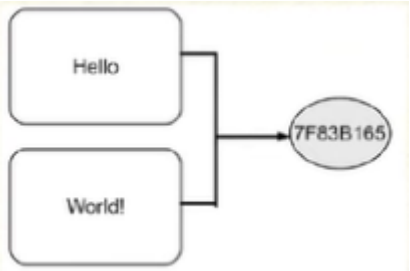
1. **Independent Hashing**- combinational hash of all hashes
   - String hello world, generate final [any one of the new hashes will be selected as the final hash]



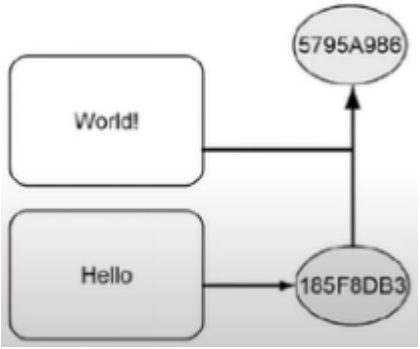
2. **Repeated Hashing**
   - H[word] -> H[H[word]]

PES UNIVERSITY

3. **Combined Hashing**
    - remove spaces, and hash entire string

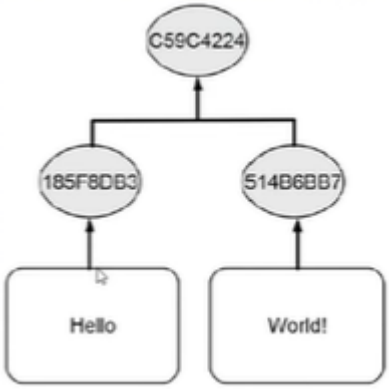


4. **Sequential hashing**
    - find hash of the word, then append to string, and then find hash of the entire thing



5. **Hierarchical hashing**

- find hash of all the words, append to each other, then find hash of new string



Hash Pointer

Hash Pointer

- A cryptographic Hash pointer (hash reference) is a pointer to a location where some info is stored

- Information is stored
- Hash of the information is stored

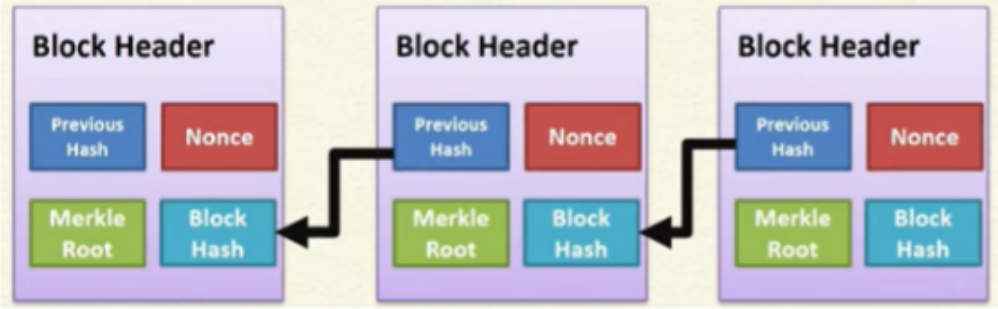
With the hash pointer, we can:

- Retrieve the information
- Check That the information has not been modified(by computing the message digest and then matching the digest with the stored hash value)

Block header

- Is used to identify a particular block on an entire blockchain
- A block header contains
  - Version
  - Time (added to blockchain)
  - Difficulty (0*100) 100 zeroes example
  - Prev (hash pointer to the previous block)
  - Nonce
  - Merkle root
  - block identifier, hash of current block header

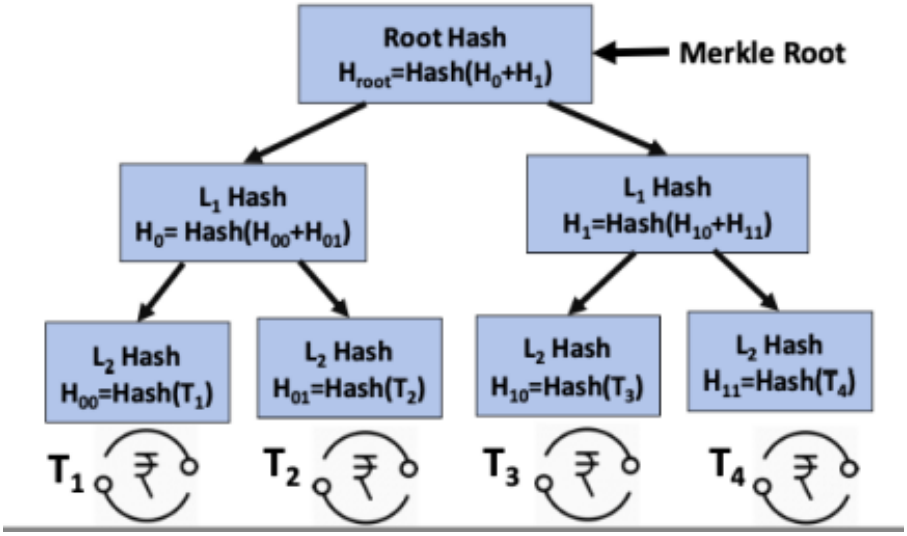
Gas Limit- the maximum amount of gas that you're willing to pay to run a transaction.
Gas Price - the amount paid per unit of gas as a fee to the miner.

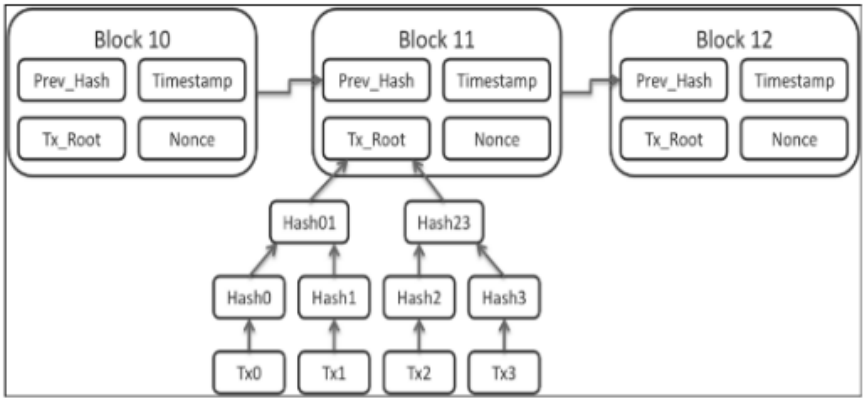


Merkle Tree – Organization of Hash Pointers in a Tree

- A Merkle tree, each non-leaf node is the hash of the values of their child nodes. Leaf Node: The leaf nodes are the nodes in the lowest tier of the tree. The leaf nodes will be L1, L2, L3 and L4.

- The nodes labeled “Hash 0-0” and “Hash 0-1” are the child nodes of the node labeled “Hash 0”.





Transaction and Trade

- A transaction is when two parties exchange something.

Trade

- Blockchain in financial trading means **transparent pricing**, **new alternative markets**, **faster payment processing** and **immutable transaction record keeping.**
- Blockchain's ledger technology is enabling people to trade for **lower costs** and at **faster speeds**
- From a security perspective, blockchain allows simple, secured share trade-related data between different financial institutions.

Public Witness

- A public witness is a person that is attesting to a fact or event.
- Their testimony allows others to believe that something took place.

A public witness serves two main functions

1. It spreads knowledge to more than one individual and allows history to become persistent.

PES UNIVERSITY

2. Allows the individual to make a choice about the information they have been given.

Blockchain technology is an extension of the public witness concept in that it spreads knowledge, encourages persistence of information, and allows each individual node to make a choice with the information that they are given.

Three pillars of blockchain

1. Distributed Systems
2. Crypto
3. Economic Models

DISTRIBUTED CONSENSUS

Classical distributed consensus problem:

- Getting all the nodes to agree to addition of the new block

Centralized Approach

- Have to trust a 3rd party to make the decisions
- There can be a bias during decisions

Decentralized Approach

- Expensive operation to send data across all data
- Can't trust the data sent to every node, the different data can be sent to multiple nodes. This is a malicious behaviour. **Byzantine fault**

Distributed Consensus

- A procedure to reach in a common agreement in a distributed for decentralized multi agent platform
- communicate with **message passing** system

Why Consensus?

- **Reliability** and **fault tolerance** in a distributed system
  - Ensure correct operations in the presence of faulty individuals

Examples

- Commit a transaction in a database
- State machine replication
- Clock synchronization

No failure case

- If no failure it is easy and trivial to reach to a consensus
  - Broadcast the message
  - Apply a choice function, majority voting

  Choice function is usually decided while developing the blockchain network

Types of Faults

1. Crash fault

- A node suddenly crashes or becomes unavailable in the middle of a communication

2. Network or Partitioned fault

- A network fault occurs and the network gets partitioned

3. Byzantine fault

- A node starts behaving maliciously

Easy to solve the first two faults, it is difficult to solve the third fault
Byzantine fault is an important problem to solve in distributed systems

Distributed Consensus Properties

1. Termination

- Every correct individual decides some value at the end of the consensus protocol

2. Validity

- If all the individuals proposes the same value, then all correct individuals decide on that value
- cannot change later

3. Integrity

- Every correct individual decides at most one value, and the decided value must be proposed by some individuals

4. Agreement

- Every correct individual must agree on the same value

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

Consensus is impossible in a fully asynchronous system even with a single crash fault

- Cannot ensure "Safety" and "Liveness" together

Safety

- correct individuals must not agree on an incorrect value
- correct process will yield correct output
- Nothing bad happened

Liveliness (or liveness)

- Every correct value must be accepted
- The output will be produced within a finite amount of time (eventual termination)
- Something good eventually happens

HISTORY

1. 1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson
    - Consensus is impossible in a fully asynchronous system even with a single crash fault
    - Cannot ensure "Safety" and "Liveness" together
2. 1989: Lamport started talking about "Paxos"
    - Supports safety but not the liveness
3. 1990's: Everyone were confused about the correctness of Paxos
4. 1998: Paxos got published in ACM Transactions on Computer Systems
5. 2001: FLP Impossibility paper wins Dijkstra Prize
    - People starts talking about Distributed Systems
6. 2009: Zookeeper released
    - Service for managing distributed applications

Why can't we have Classic Distributed Consensus (CDC) approach in BC[distributed consensus limitation]

- have to send a lot of messages
- No requirement to send all the transactions
- anonymous entities
- open network, we don't know the layout of the network

Consensus in an open network: Puzzle Solving

- majority agrees that the puzzle has been solved correctly
- A leader is elected, and the decisions by the leader have to be accepted
- A network gives a puzzle and everyone tries to solve it
- One who gives the solution becomes the leader
- Whatever the leader says, everyone agrees to that

How to check if the solution solved by leader is the correct one?

- The puzzle is really difficult to solve
- The puzzle should be easy to verify

The puzzle

$H(M||k) = H1$
Miner has to find `k`

NAKAMOTO WHITE PAPER(2008)

- Give more emphasis on "Liveness" rather than "Safety"
- Participants may agree on a transaction that is not the final one in the chain
- Proof of Work (PoW) -- Nakamoto Consensus
- Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup
- Have not coined the term "Blockchain" in the paper !!

DATES

2011: Litecoin got introduced
2015: Ethereum network went live
2016: Term "Blockchain" got popular

Why do we require consensus in bitcoin network?

- A node does not know all the peers in an open network
- Some nodes can also initiate malicious transactions

Consensus in a Bitcoin Network

- Every node has block of transactions that has already reached into the consensus **(block of committed transactions).**
- The nodes also has a list of outstanding transactions that need to be validated against the block of committed transactions.

Bitcoin Consensus Objective:

Which block do we add next?

Challenge:

The miners do not know each other

Possible Solution:

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

May not be Feasible:

You do not have a global clock! How much time will you wait to hear the transactions. Remember the impossibility result

Observation - 1:
• Any valid block (a block with all valid transactions) can be accepted, even if it is proposed by only one miner

Observation - 2:
• The protocol can work in rounds
• Broadcast the accepted block to the peers
• Collect the next set of transactions

Solution:

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

Note: This communication can work asynchronously

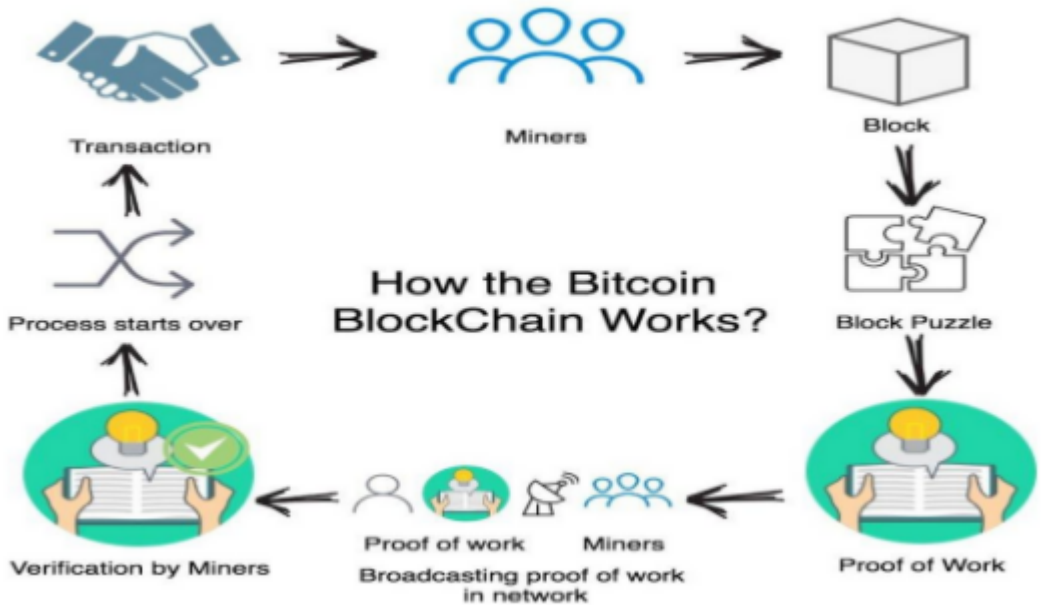
Every node has a block of transactions that has already reached into the consensus(block of committed transactions)

- The nodes also has a list of outstanding transactions
- A miner might not take all the transactions
  - Miner 1: T15,T16,T18,T19
  - Miner 2: T15,T16,T17
  - Miner 3: T15,T16,T17,T18
- Transactions in a block can be different
- If a transaction is not in the block which has been added in the latest blockchain, the transaction will be added in the next block
- When it's taken the next time from the pool, validation happens before adding

Bitcoin Blockchain Network

- Decentralized digital currency enables instant payments to anyone, anywhere in the world
- no central authority-peer to peer
- Two broad operations:
  - **Transaction Management**: transfer of bitcoins from one user to another
  - **Money Issuance**: regulate the monetary base
- A node can be a user operating the system

- A user uses a wallet to use the system
- A wallet has to generate public and private key
- in a bitcoin network public address is hashed and is used for transfers



Miners

There are special nodes, called the Miners
• Miners propose new blocks –

- solve the puzzle (find the nonce corresponding to a target block hash),
- and 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)

Creation of Coins

- **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 difficulty is changed every **two weeks**
- The **number of bitcoins generated** per block is set to decrease geometrically, with a 50% reduction for every 210,000, or approximately **4 years**
- This reduces, with time, the amount of bitcoin generated per block
  - Theoretical limit is 21 million
  - Miners will get less reward as time progresses
  - How to pay the mining fee – **increase the transaction fee**

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)

Example:

- Alice wish to transfer some bitcoin to Bob.
  - Alice can **sign a transaction** with her private key
  - Anyone can **validate the transaction** with Alice's public key

How can we prevent double spending in a decentralized network?

Handle Double Spending using Blockchain

- Details about the **transaction are sent and forwarded** to all or as many other computers as possible.
- Use Blockchain – a **constantly growing chain of blocks** that contain a record of all transactions
- The blockchain is maintained by **all peers have a copy of the blockchain**
- To be accepted in the chain, transaction blocks must be valid and **must include 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.**
- When multiple valid continuation to this chain appear, only **the longest branch is accepted** and extended further
- Once a transaction is committed in the blockchain, everyone in the network 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 setup any "account"
- The public and the 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 user name 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: 1PHYrmdJ22MKbJevpb3MBNpVckjZHt89hz
- Each person can have many such addresses, each with its own balance- Difficult to know which person owns what amount

Bitcoin P2P Network

- An ad-hoc network with random topology, Bitcoin protocol runs on **TCP port 8333**.
- All nodes (users) in the bitcoin network are **treated equally**.
- New nodes can join any time, non-responding nodes are removed after 3 **hours**.

Joining in a Bitcoin P2P Network

1. Create a wallet
2. The joining node asks, a seed node to send the address list of neighbouring nodes
3. Then it connects with the neighbouring nodes from the list, and asks to get most recent blockchain
4. Then it can start transactions and being part of the blockchain
5. Alice broadcast all the transaction to neighbors, with the scripts
6. The neighboring nodes validate the transactions
7. They then send to the other neighbours
8. If already seen the transaction doesn't add it to the list of transactions
9. Different nodes may have different transactions pools at any point of time
10. Each node accepts the first transactions that the have heard
11. Miner collects all the transactions flooded and starts mining(creating merkle tree hash)
12. Miner who solves the puzzle first generates a new block and gets reward
13. Flood the blockchain with the new block included
14. Multiple miners can mine a new block simultaneously or in a near identical time

What are these Dapps?

- DLTs can contain information beyond financial transactions
- What about submitting an executable code as a transaction?
- Transaction gets executed == Your code is getting executed
- And you have consensus on the code execution!

Smart Contract

A smart contract is a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of code.

- Nick Szabo, an American computer scientist who invented a virtual currency called **"Bit Gold" in 1998**, defined smart contracts as computerized transaction protocols that execute terms of a contract.
- Smart contracts render transactions **traceable, transparent, and irreversible.**

Smart Contract Gives you

- Autonomy
- Trust
- Backup

- Safety
- Speed
- Savings
- Accuracy

Smart Contract Anatomy

1. Agreement Identification
-Identifying cooperative opportunities for multiple parties
-Potential agreements on transfer of rights and asset swaps
2. Condition Setting
- Event based conditional triggers like natural disaster
- Temporal conditions triggers like anniversary and death
3. Business Logic Coding
- Fully automated coding logic which triggers where certain logic conditions are met
4. Digital Signature
- Provision of secure authentication and message verification between parties related to a smart contract
5. Process Execution
- Once the consensus about authentication and verification reached, a smart contract gets executed and its outcomes are memorialized for compliance and audit
6. Updating Network
- After a smart contract is executed , every distributed ledger nodes gets updated with the same state so the new updates can only be appended

Smart Contract Components

- **Smart contract code:** For instance, Ethereum Solidity code (that is stored, verified and executed on a blockchain).
- **Smart legal contracts:** These are written as a specification for using smart contract code as a complement/substitute for legal contracts executed in traditional usage.

Smart Contract Example

A solidity smart contract includes the following:

- **Data** - This maintains the current state of the contract.
- **Function** - This applies the logic for transitioning the state of the contract.

Pragma Directive

- The 'pragma' keyword is used to enable some compiler features or checks.
- Pragma solidity >=0.4.0 <=0.6.0

Solidity

- It is an object-oriented high-level programming language used to implement smart contracts.
- Solidity helps create contracts for crowdfunding, blind auctions, voting, and multi-signature wallets.
- Solidity is influenced by languages such as Python, JavaScript, and C++.

Two types of variables one needs to be familiar
with in smart contracts.

- **Value Type :** These variables are passed by value.
- **Reference Type:** These variables are of complex types and are passed by reference.

Contract Declaration

- This is declared through the keyword 'contract.'
- This will declare an empty contract identified by the name of "PurchaseOrder."

This is depicted as follows:

- **contract PurchaseOrder{**
  ...
  }

Adding Data Variables to the Smart Contract

- **INT** -refers to an integer.
- **U** -means unsigned, which means that this type can represent only positive integers and not both positive and negative integers.
- **25** -This refers to the 256 bits in size.
- The minimum value of uint 256 can be assigned to is **0**.
- The maximum value that can be assigned for uint 256 is **2^256-1**.

Defining Constructor

- A constructor is one that is **called at the time of deployment** of a contract.
- The constructor uses some values to initialize the contract.
- It is also possible to create a **parameterized constructor** which can be created by passing a variable and initializing the function using the passed in value.
- The access modifier **"public"** associated with the constructor is the key point to note here.
- The keyword "public" denotes that anyone can access the function. Hence, this is **not a restricted function.**

Adding Functions

- Functions refer to controlled capabilities that can be added to a program.
- Any function will always be preceded by the **keyword function**.

A function declaration looks like this:

- **function <function name> <access modified> <state mutator> <return value>**

Get Functions

- The most common requirement is to read the stored value.

Setter Functions

- It is necessary to read the data, but it is also necessary to have the capability to write/update data.
- state-mutator- not needed as the state is updated by the functions.

Remix

- Remix is an online playground that is used for Ethereum smart contracts.
- It is completely based on a browser.
- It provides an integrated development environment where you can write your smart contracts.
- Remix provides an online solidity compiler capability.
- Remix IDE helps compile a smart contract seamlessly using
- a specific compiler version.
- It helps in the quick testing of a smart contract.