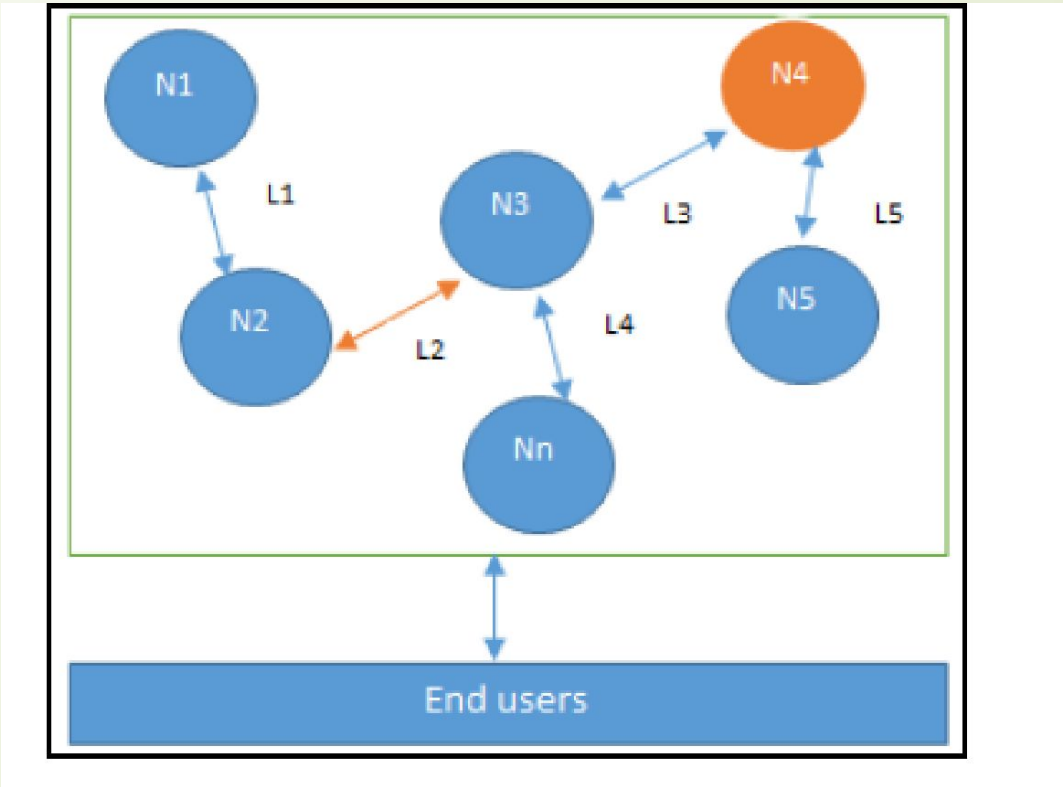
Blockchain Essentials & DApps

Distributed systems

- Distributed systems are a computing paradigm whereby **two or more nodes work with each other in a coordinated fashion** in order to achieve a common outcome
- DS modeled in such a way that end users see it as a **single logical platform**.
- Node can be defined as an **individual player in a distributed system** and have their own memory and processor.
- All nodes are capable of **sending and receiving messages** to and from each other.
- Nodes can be honest, faulty, or malicious

- A node that can exhibit **arbitrary behavior** is also known as a **Byzantine node**.
- This arbitrary behavior can **be intentionally malicious**, which is **detrimental** to the operation of the network.
- Generally, any **unexpected behavior** of a node on the network can be categorized as **Byzantine**.



Challenge in distributed system Design

- **Coordination** between nodes
- **Fault tolerance**.
  - Even if some of the nodes become faulty or network links break, **the distributed system should tolerate**
  - D S should **continue to work** flawlessly in order to achieve the desired result.
  - Several algorithms and mechanisms has been proposed to overcome these issues.
- Distributed systems are so challenging to design that a theorem known as the **CAP theorem** has been proved and states that a **distributed system cannot have all much desired properties simultaneously.**

CAP theorem (Brewer's theorem)

- Introduced originally by *Eric Brewer* in 1998;
- In 2002 it was proved as a theorem by *Seth Gilbert* and *Nancy Lynch*.
- The theorem states that any distributed system cannot have Consistency, Availability, and Partition tolerance simultaneously
  - Somehow blockchain manages to achieve all these properties

CAP theorem

- **Consistency** is a property that ensures that all nodes in a distributed system have a single latest copy of data
- **Availability** means that the system is up, accessible for use, and is accepting incoming requests and responding with data without any failures as and when required
- **Partition tolerance** ensures that if a group of nodes fails the distributed system still continues to operate correctly

- In order to achieve fault tolerance, replication is used.
- Consistency is achieved using consensus algorithms to ensure that all nodes have the same copy of data. This is also called **state machine replication.**
- Blockchain is basically a method to achieve **state machine replication.**

Types of fault

- Faulty node has simply **crashed**
- Faulty node can exhibit **malicious or inconsistent behavior arbitrarily**.
  - Difficult to deal with since it can cause confusion due to misleading information.

Byzantine Generals problem

- Generals = Computer Components
- The abstract problem…
  - Each division of Byzantine army is directed by its own general.
  - There are n Generals, some of which are **traitors**.
  - All armies are camped outside enemy castle, observing enemy.
  - Communicate with each other by **messengers.**
  - Requirements:
    - G1: All ***loyal generals*** decide upon the same plan of action
    - G2: A **small number of traitors** cannot cause the loyal generals to adopt a bad plan
  - Note: We **do not** have to identify the traitors.

Motivation

- Coping with **failures** in computer systems
- Failed component sends **conflicting information** to different parts of system.
- **Agreement** in the presence of faults.
- P2P Networks?
  - Good nodes have to "**agree to do the same thing**".
  - Faulty nodes **generate corrupted and misleading messages**.
  - **Non-malicious:** Software bugs, hardware failures, power failures
  - **Malicious reasons**: Machine compromised.

Byzantine Generals problem

- In September 1962, *Paul Baran* introduced **the idea of cryptographic signatures** with his paper *On distributed communications networks*.
  - This is the paper where the concept of **decentralized networks** was also introduced for the very first time.
- Then in 1982 a thought experiment was proposed by *Lamport et al.* whereby a group of army generals who are leading different parts of the Byzantine army are planning to attack or retreat from a city.
- The only way of communication between them is a messenger and they need to agree to attack at the same time in order to win.
- The issue is that one or more generals can be traitors and can communicate a misleading message.

- Need for a **viable mechanism** that allows agreement between generals even in the presence of treacherous generals so that the attack can still take place at the same time.
- In distributed systems, generals can be considered as **nodes**, traitors can be considered **Byzantine (malicious) nodes**, and the messenger can be thought of as a **channel** of communication between the generals.
- In 1999,*Castro* and *Liskov* presented the **Practical Byzantine Fault Tolerance** (**PBFT**) algorithm.
- In2009, the first practical implementation was made with the invention of bitcoin where the **Proof of Work** (**PoW**) algorithm was developed as a mechanism to achieve consensus.

Consensus

- Consensus is a process of agreement between distrusting nodes on a final state of data.
- It is easy to reach an agreement between two nodes
- But when multiple nodes are participating in a distributed system and they need to agree on a single value – Difficult Task
- Distributed consensus: Achieving consensus between multiple nodes

Consensus mechanisms

- A consensus mechanism is a set of steps that are taken by all, or most, nodes in order to agree on a proposed state or value.
- Consensus mechanisms have recently come into the **limelight and gained much popularity** with the advent of **bitcoin and blockchain**.

Requirements which must be met in order to provide the desired results in a consensus mechanism.

- **Agreement**: All honest nodes decide on the **same value**.
- **Termination**: All honest nodes terminate execution of the consensus process and eventually **reach a decision**.
- **Validity**: The value agreed upon by all **honest nodes** must be the same as the **initial value** proposed by at least one honest node.
- **Fault tolerant**: The consensus algorithm should be able to run in the presence of faulty or malicious nodes **(Byzantine nodes).**
- **Integrity**: The **nodes make decisions only once** in a single consensus cycle. No node makes the decision more than once.

Types of consensus mechanism

- **Byzantine fault tolerance-based**:
  - This method relies on a simple scheme of nodes that are **publishing signed messages**.
  - When a **certain number of messages are received**, then an agreement is reached.
- **Leader-based consensus mechanisms**:
  - This type of mechanism requires nodes to compete for the ***leader****-election lottery*
  - The **node that wins it proposes a final value**

Practical implementations

- **Paxos**, the most famous protocol introduced by **Leslie Lamport** in 1989.
- In Paxos nodes are assigned various roles such as
  - Proposer
  - Acceptor
  - Learner.
- Nodes or processes are named replicas and consensus is achieved in the presence of faulty nodes by agreement among a majority of nodes.

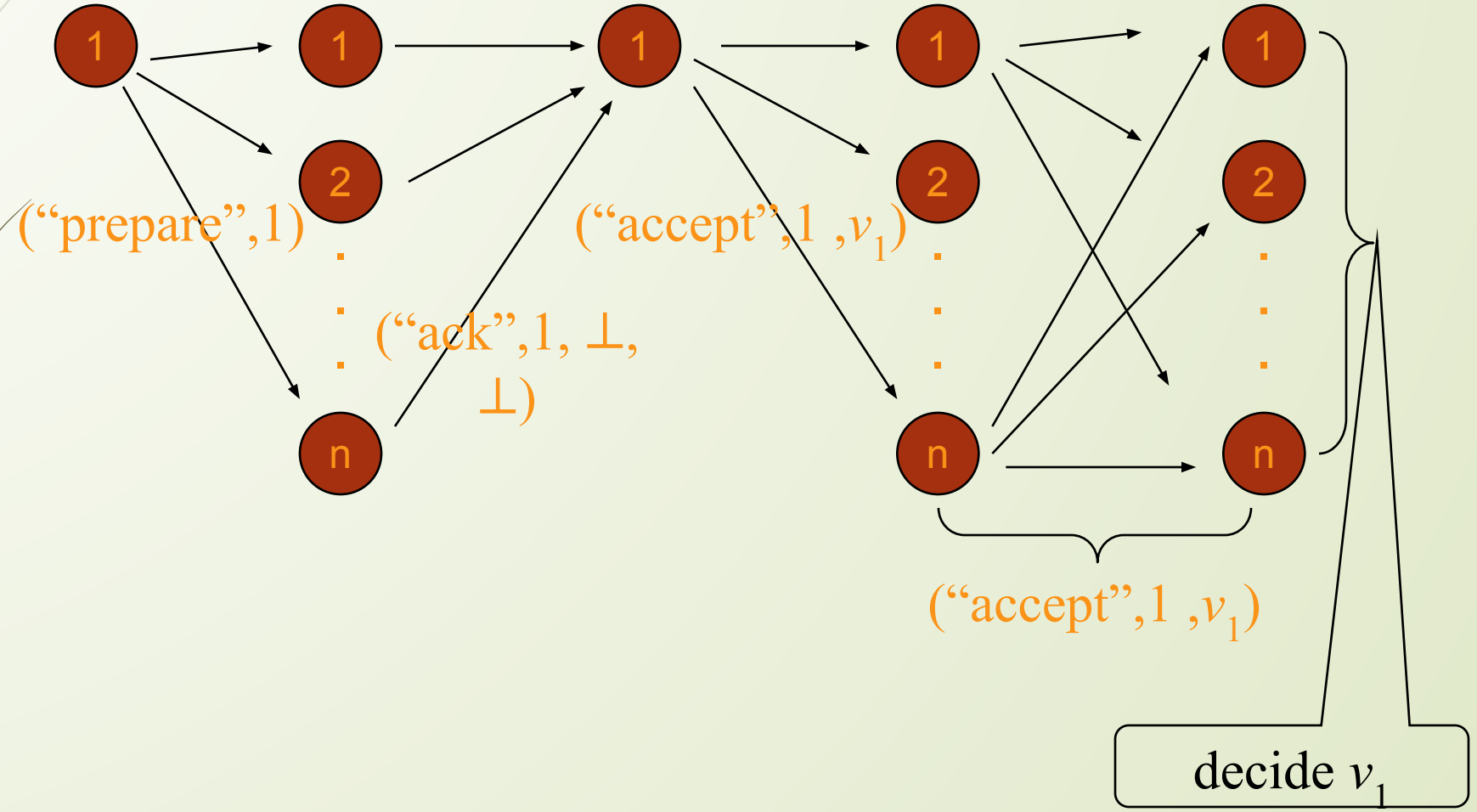
Paxos Terms

- ***Proposer***
  - Suggests values for consideration by Acceptors.
  - Advocates for a client.
- ***Acceptor***
  - Considers the values proposed by proposers.
  - Renders an accept/reject decision.
- ***Learner***
  - Learns the chosen value.
- In practice, each node will usually play all three roles.

- ***Proposal***
  - An alternative proposed by a proposer.
  - Consists of a unique ***number*** and a proposed ***value.***

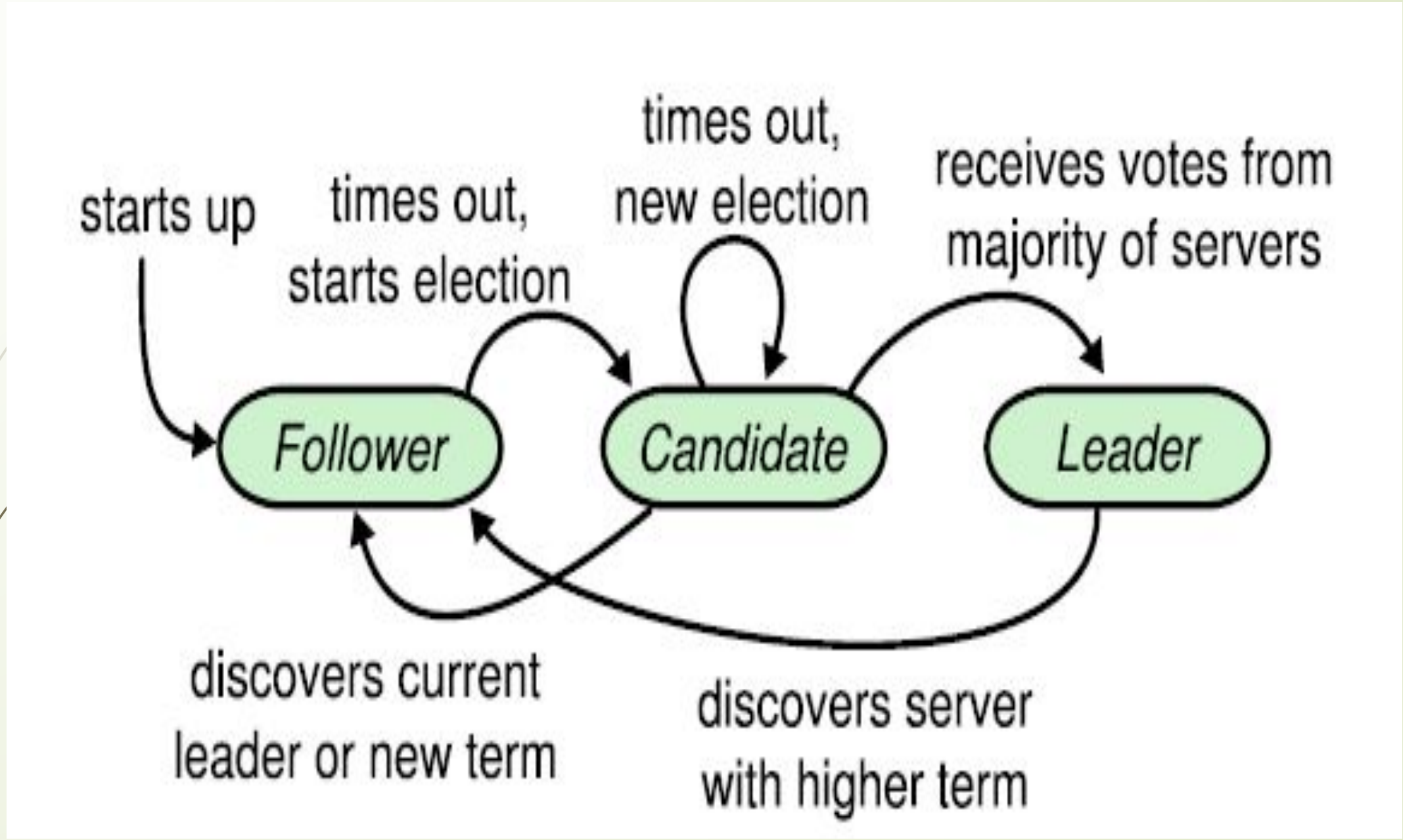
    ```
    ( 42, B )
    ```

- We say a value is ***chosen*** when consensus is reached on that value.


1
1
1
1
1
2
2
2
n
n
n
("prepare",1)
("ack",1, ⊥, ⊥)
("accept",1 ,v_1)
("accept",1 ,v_1)
decide v_1

Practical implementations

- Another alternative to Paxos is **RAFT**, which works by assigning any of three states, that is,
  - Follower
  - Candidate
  - Leader.
- A Leader is elected after a candidate node receives enough votes
- All changes now have to go through the Leader, who commits the proposed changes once replication on the majority of follower nodes is completed.


starts up
times out,
starts election
times out,
new election
receives votes from
majority of servers
Follower
Candidate
Leader
discovers current
leader or new term
discovers server
with higher term

The history of blockchain

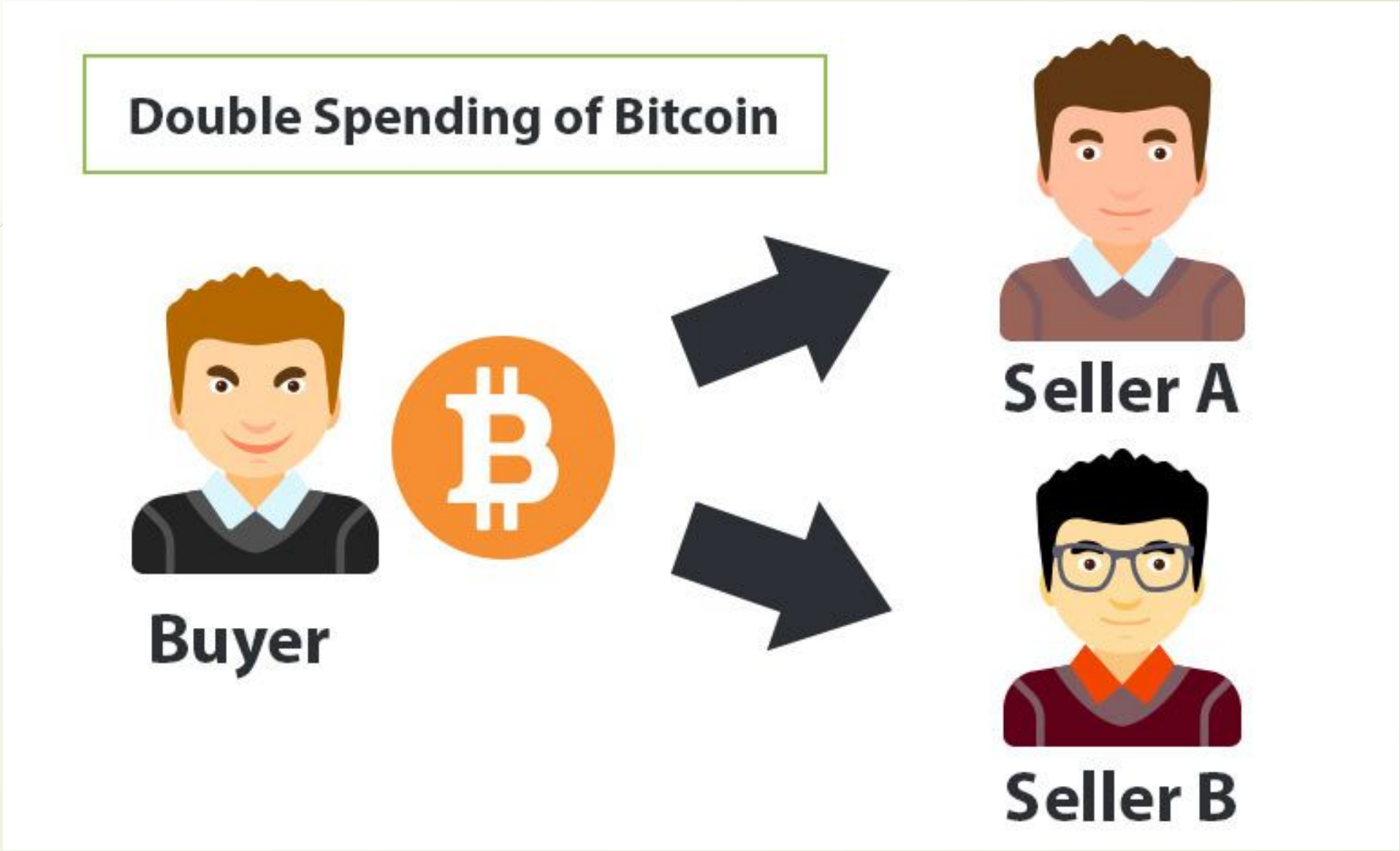
- Blockchain was introduced with the invention of bitcoin in 2008 and then with its practical implementation in 2009.
- The concept of electronic cash or digital currency is not new.
- Since the 1980s, e-cash protocols have existed that are based on a model proposed by David Chaum.

Electronic cash

- The idea of electronic cash is also essential to appreciate the first and astonishingly successful application of blockchain:
  - the bitcoin, or broadly cryptocurrencies.
- Theoretical concepts in distributed systems such as consensus algorithms provided the basis of the practical implementation of Proof of Work algorithms in bitcoin;
- Ideas from different electronic cash schemes also paved the way for the invention of cryptocurrencies, specifically bitcoin

The concept of electronic cash

- Fundamental issues that need to be addressed in e-cash systems are
  - Accountability
  - Anonymity.
- David Chaum addressed both of these issues in his seminal paper in 1984 by introducing two cryptographic operations,
  - Blind signatures
  - Secret sharing.
- Blind signatures allow signing a document without actually seeing it
- Secret sharing is a concept that allows the detection of using the same e-cash token twice (**double spending**).
- Chaum, Fiat, and Naor (CFN) Protocol : e-cash schemes that introduced anonymity and double spending detection.


Double Spending of Bitcoin
Buyer
Seller A
Seller B

Brand's e-cash

- **Brand's e-cash** is another system that improved on CFN, made it more efficient, and introduced the concept of security reduction to prove statements about the e-cash scheme.
  - Security reduction is a technique used in cryptography to prove that a certain algorithm is secure by using another problem as a comparison.
  - cryptographic security algorithm is as hard to break as some other hard problem; thus by comparison it can be deduced that the cryptographic security algorithm is secure too.

hashcash

- A different but relevant concept called **hashcash** was introduced by *Adam Back* in 1997 as a PoW system to control e-mail spam.
- if legitimate users want to send e-mails then they are required to **compute a hash as a proof that they have spent a reasonable amount of computing resources** before sending the e-mail.
- **Generating hashcash is a compute intensive process** but does not inhibit a legitimate user from sending the e-mail
  - because the usual number
  - e-mails required to be sent by a legitimate user is presumably quite low.
- If a spammer wants to send e-mails, usually thousands in number, then it becomes infeasible to compute hashcash for all e-mails, thus making the spamming effort expensive;

- Hashcash is popularized by its use in the **bitcoin mining** process.
- This idea of using **computational puzzles or pricing functions** to prevent e-mail spam was introduced originally in 1992 by *Cynthia Dwork* and *Moni Naor*.
- Pricing function was the name given to the **hard functions** that are required to be computed before access to a resource can be granted.
- *Adam Back* invented hashcash independently in 1997, which introduced the usage of computing hash functions as **PoW.**

b-money

- In 1998 b-money was introduced by Wei Dai and proposed the idea of **creating money** via **solving computational puzzles** such as hashcash.
- It's based on a peer-to-peer network where each node maintains its own list of transactions.

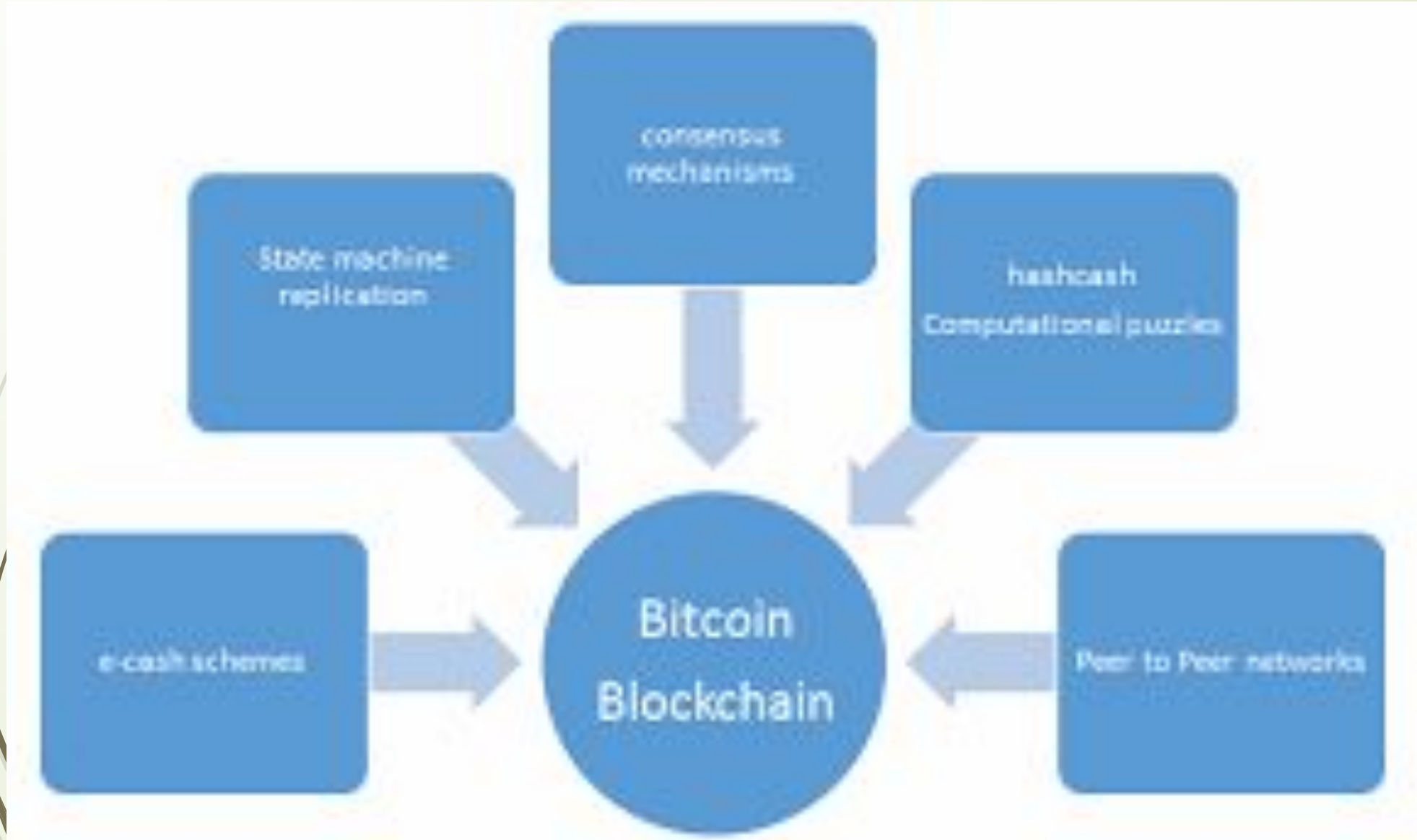
BitGold

- Another similar idea by *Nick Szabo* called **BitGold** was introduced in 2005 and also proposed **solving computational puzzles to mint digital currency**.
- In 2005 *Hal Finney* introduced the concept of **cryptographic currency** by combining ideas from **b-money and hashcash** puzzles
  - but it still relied on a centralized trusted authority.

bitcoin

- In 2009 the first practical implementation of a cryptocurrency named **bitcoin** was introduced;
- For the very first time it solved the problem of **distributed consensus in a trustless network.**
  - It uses **public key cryptography** with **hashcash as PoW** to provide a secure, controlled, and decentralized method of minting digital currency.
- The key innovation is the idea of an **ordered list of blocks** composed of transactions and cryptographically secured by the **PoW mechanism**

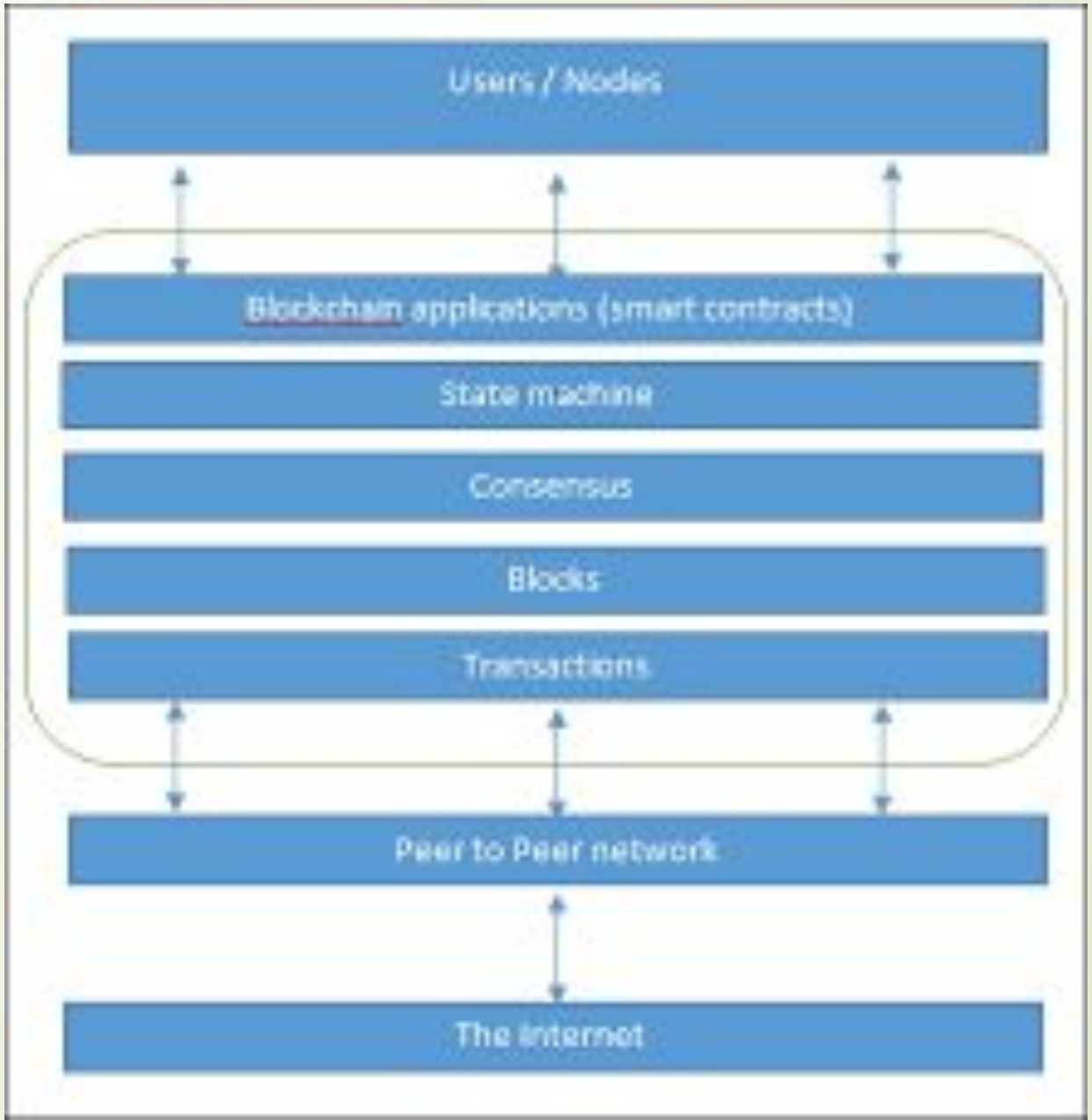
- Ideas and concepts from electronic cash schemes and distributed systems were combined together to invent bitcoin
- Now it is known as blockchain.



Introduction to blockchain

- **Blockchain** at its core is a **peer-to-peer distributed ledger** that is
  - cryptographically secure
  - append-only
  - immutable (extremely hard to change)
  - updateable only via consensus or agreement among peers.
- Blockchain can be thought of as **a layer of a distributed peer-to-peer network** running on top of the Internet

The network view of a blockchain

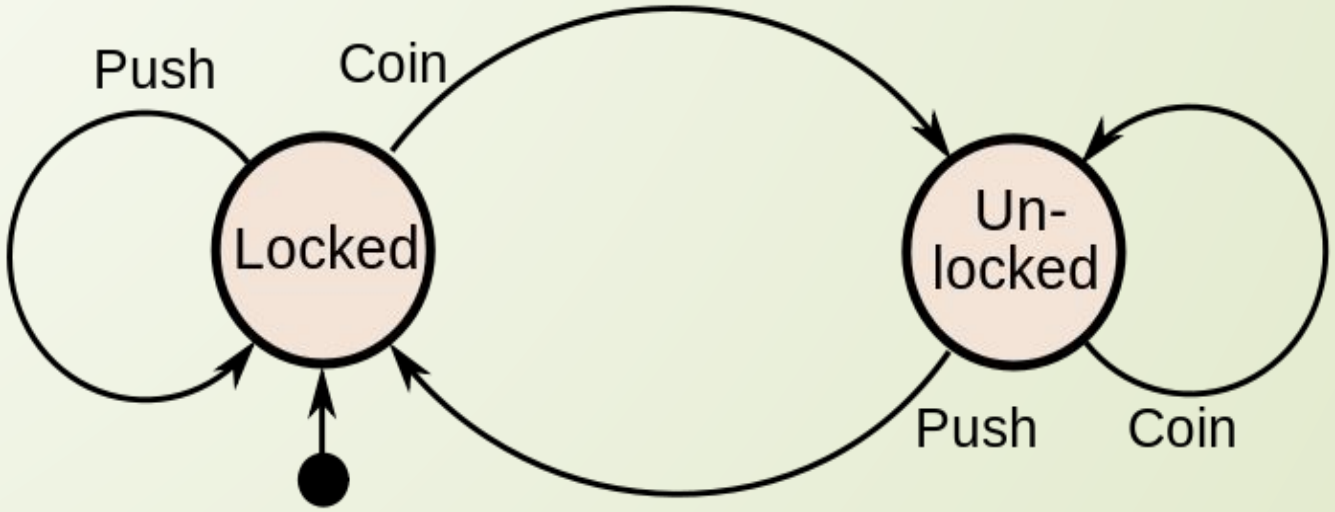


Introduction to blockchain

- From a business point of view a blockchain can be defined as a platform whereby **peers can exchange values using transactions without the need for a central trusted arbitrator**.
  - Blockchain to be a **decentralized consensus mechanism** where no single authority is in charge of the database

State Machines

- State Variables
- Deterministic Commands



Consensus

- Termination
- Validity
- Integrity
- Agreement

- Ensures procedures are called in same order across all machines

Fault Tolerant State Machines

- Implement the state machine on multiple processors.
- State Machine Replication
  - Each starts in the same initial state
  - Executes the same requests
  - Requires consensus to execute in same order
  - Deterministic, each will do the exact same thing
  - Produce the same output.

state machine replication

- state machine replication or state machine approach is a general method **for implementing a fault-tolerant service** by replicating servers and coordinating client interactions with server replicas
  - Make server deterministic (state machine)
  - Replicate server
  - Ensure correct replicas step through the same sequence of state transitions
  - Vote on replica outputs for fault-tolerance

Replica Coordination

- All non-faulty state machines receive all commands in the same order
- Every non-faulty state machine processes the commands it receives in the same order

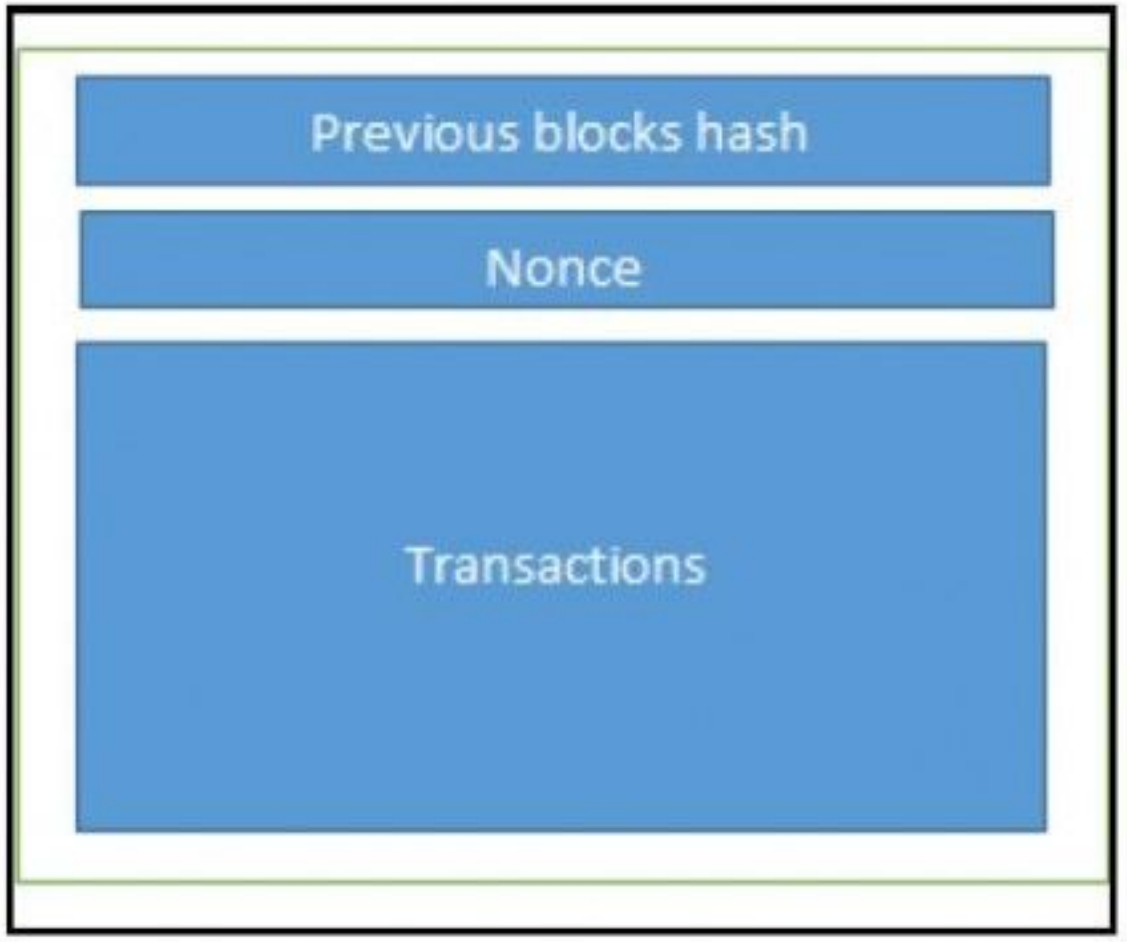
Block

- A block is simply a selection of transactions bundled together in order to organize them logically.
- It is made up of transactions and its size is variable depending on the type and design of the blockchain in use.
- A reference to a previous block is also included in the block unless it's a genesis block.
- A genesis block is the first block in the blockchain that was hardcoded at the time the blockchain was started.
- The structure of a block is also dependent on the type and design of a blockchain

Block

- Generally there are a few attributes that are essential to the functionality of a block
  - Block header
  - Pointers to previous` blocks
  - Time stamp
  - Nonce
  - Transaction counter
  - Transactions
  - Other attributes.

The structure of a block



The structure of a block

Various technical definitions of blockchains – Definition 1

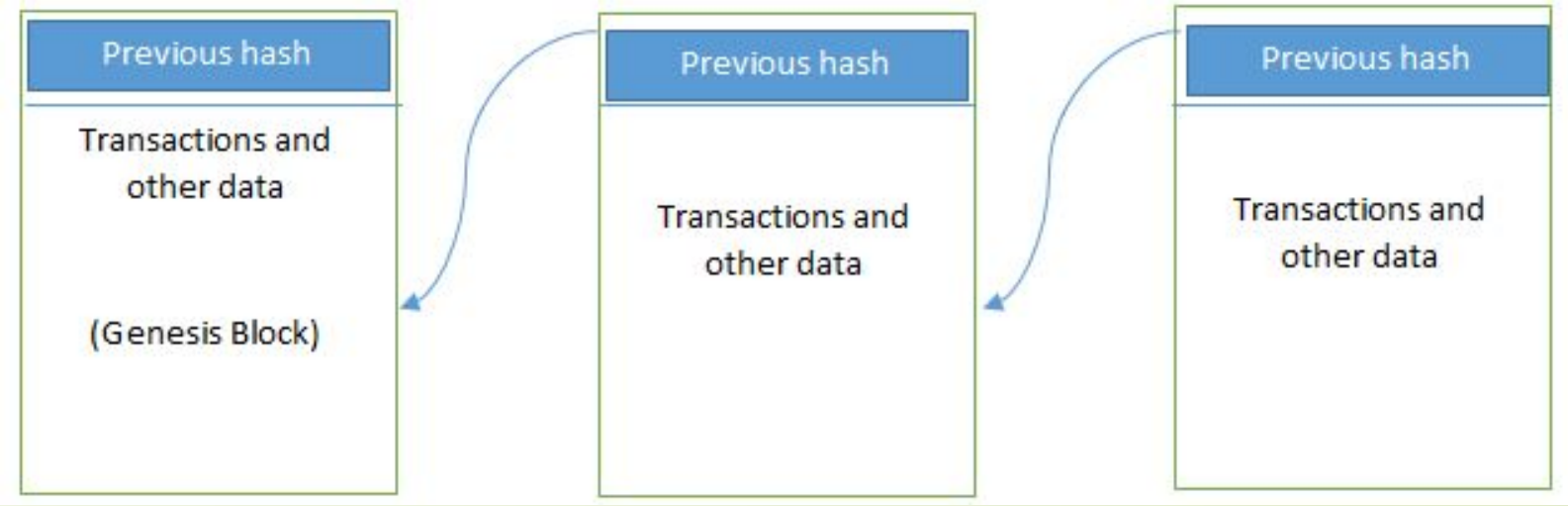
- Blockchain is a decentralized consensus mechanism.
  - In a blockchain, all peers eventually come to an agreement regarding the state of a transaction.

Definition 2

- Blockchain is a distributed shared ledger.
- Blockchain can be considered as a shared ledger of transactions.
- The transaction are ordered and grouped into blocks.
- Real-world model is based on private databases that each organization maintains whereas the **distributed ledger** can serve as a **single source** of truth for all member organizations that are using the blockchain.

Definition 3

- Blockchain is a data structure;
- it is basically a linked list that uses hash pointers instead of normal pointers.
- Hash pointers are used to point to the previous block.



Generic elements of a blockchain - **Addresses**

- Addresses are **unique identifiers** that are used in a transaction on the blockchain **to denote senders and recipients.**
- An address is usually a public key or derived from a public key.
- Addresses can be reused by the same user, addresses themselves are unique.
- **In practice, single user may not use the same address again** and generate a new one for each transaction.

Addresses

- Bitcoin is in fact a pseudonymous(false) system.
  - End users are usually not directly identifiable
  - But some research in de-anonymizing bitcoin users have shown that users can be identified successfully.
- As a good practice it is suggested that users generate a new address for each transaction
  - In order to avoid linking transactions to the common owner, thus avoiding identification.

Transaction

- A transaction is the fundamental unit of a blockchain.
- A transaction represents a transfer of value from one address to another.

Block

- A block is composed of multiple transactions and some other elements such as the previous
  - block hash (hash pointer)
  - Timestamp
  - Nonce.
    - A nonce ("number only used once") is a number added to a hashed block that, when rehashed, meets the difficulty level restrictions.
    - The nonce is the number that blockchain miners are solving for.

Block

- The nonce can be found as a 4-byte field in a block header,
- Its value adjusted by miners so that the **hash of the block will be less than or equal to the current target hash value set by the network**.
- Miners will hash slight variations of the input data, which for the mining process will be the nonce,
- Finding such a hash value during the mining process is known as a golden nonce.

Peer-to-peer network

- Network topology whereby all peers can communicate with each other and send and receive messages.

Scripting or programming language

- This element performs various operations on a transaction.
- Transaction scripts are predefined sets of commands for nodes to transfer tokens from one address to another and perform various other functions.
- Turing complete programming language is a desirable feature of blockchains;

Virtual machine

- A virtual machine allows Turing complete code to be run on a blockchain (as smart contracts)
- Virtual machines are not available on all blockchains;
- Various blockchains use virtual machines to run programs
  - Ethereum Virtual Machine (EVM)
  - Chain Virtual Machine (CVM).

State machine

- A blockchain can be viewed as a state transition mechanism
  - whereby a state is modified from its initial form to the next and eventually to a final form as a result of a transaction execution and validation process by nodes.

Nodes

- A node in a blockchain network performs various functions depending on the role it takes.
- A node can propose and validate transactions and perform mining to facilitate consensus and secure the blockchain.
- Nodes can also perform other functions such as
  - simple payment verification (lightweight nodes)
  - validators,
  - many others functions.

Smart contracts

- Programs run on top of the blockchain
- Encapsulate the business logic to be executed when certain conditions are met.
- The smart contract feature is not available in all blockchains
- It is becoming a very desirable feature due to the flexibility and power it provides to the blockchain applications.
- Self-automated computer programs that can carry out the terms of any contract
- Mostly based on objective conditions precedent
  - “If, then” criteria

101 Blockchains | KEY FEATURES OF BLOCKCHAIN TECHNOLOGY

01 CANNOT BE CORRUPTED

Every node on the network has a copy of the digital ledger. To add a transaction every node needs to check its validity. If the majority thinks it's valid, then it's added to the ledger. This promotes transparency and makes it corruption-proof.

02 DECENTRALIZED TECHNOLOGY

The network is decentralized meaning it doesn't have any governing authority or a single person looking after the framework. Instead, a group of nodes maintain the network making it decentralized.

03 ENHANCED SECURITY

As it eliminates the need for central authority, no one can just simply change any characteristics of the network for their benefit. Also using encryption ensures another layer of security for the system.

04 DISTRIBUTED LEDGERS

The ledger on the network is maintained by all other users on the system. This distributes the computational power across the computers to ensure a better outcome.

05 CONSENSUS

Every blockchain thrives because of the consensus algorithms. The architecture is cleverly designed, and consensus algorithms are at the core of this architecture. Every blockchain has a consensus to help the network make decisions.

06 FASTER SETTLEMENT

Blockchain offers a faster settlement compared to traditional banking systems. This way a user can transfer money relatively faster, which saves a lot of time in the long run.

BLOCKCHAIN FEATURES

Features of a blockchain

- **Distributed consensus**
  - Enables a blockchain to present a single version of truth that is agreed upon by all parties
- **Transaction verification**
  - Any transactions posted from nodes on the blockchain are verified based on a predetermined set of rules
  - Only valid transactions are selected for inclusion in a block
- **Platforms for smart contracts**
  - A blockchain is a platform where programs can run that execute business logic on behalf of the users

- **Transferring value between peers**
  - Blockchain enables the transfer of value between its users via tokens.
- **Generating cryptocurrency**
  - A blockchain can generate cryptocurrency as an incentive to its miners who validate the transactions and spend resources in order to secure the blockchain.
- **Smart property**
  - Linking a digital or physical asset to the blockchain in an irrevocable manner such that it cannot be claimed by anyone else;
  - Full control on your asset and it cannot be double spent or double owned

- **Provider of security**
  - Blockchain is based on **proven cryptographic technology** that ensures the integrity and availability of data
    - Confidentiality is not provided due to the requirements of transparency
  - Other security services such as **nonrepudiation and authentication** are also provided by blockchain as all actions are secured by using private keys and digital signatures.
- **Immutability**
  - Records once added onto the blockchain are immutable.
  - There is the possibility of **rolling back the changes** but this is considered almost impossible to do as it will require an **unaffordable amount of computing resources**

- **Uniqueness**
  - Ensures that every transaction is unique and has not been spent already.
  - Especially relevant in **cryptocurrencies**
- **Smart contracts**
  - Revolutionary feature of blockchain as it allows **flexibility, programmability,** and much **desirable control of actions** that users of blockchain need to perform according to their specific business requirements.

Applications of blockchain technology

- Blockchain technology has a multitude of applications in various sectors including but not limited to **finance, government, media, law, and arts**.
- Almost all industries have already realized the potential and promise of blockchain and have already embarked, or soon will embark

How blockchains accumulate blocks

- A node starts a transaction by signing it with its private key.
- The transaction is propagated (flooded) by using much desirable Gossip protocol to peers
  - Peers validates the transaction based on pre-set criteria.
  - Usually, more than one node is required to validate the transactions.
- Once the transaction is validated, it is included in a block, which is then propagated on to the network.
  - At this point, the transaction is considered confirmed.

How blockchains accumulate blocks

- The newly created block now becomes part of the ledger and the next block links itself cryptographically back to this block.
  - This link is a hash pointer.
  - At this stage, the transaction gets its second confirmation and the block gets its first.
- Transactions are then reconfirmed every time a new block is created.
  - Usually, six confirmations in the bitcoin network are required to consider the transaction final.
  - To be secure against double spending, a transaction should not be considered as **confirmed** until it is a certain number of blocks deep.

How many Bitcoin Confirmations are Enough?

- 0 — Payments with 0 confirmations can still be reversed! Wait for at least one.
- 1 — One confirmation is enough for small Bitcoin payments less than $1,000.
- 3 — Enough for payments $1,000 - $10,000. Most exchanges require 3 confirmations for deposits.
- 6 — Enough for large payments between $10,000 - $1,000,000. Six is standard for most transactions to be considered secure.
- 60 — Suggested for large payments greater than $1,000,000. Less is likely fine, but this is to be safe!

Tiers of blockchain technology

- **Blockchain 1.0**
  - Introduced with the invention of bitcoin and is basically used for cryptocurrencies.
  - Categorize Generation 1 of blockchain technology to only include cryptographic currencies.
  - All alternative coins and bitcoin fall into this category.
  - This includes core applications such as payments and applications.
  - This generation started in 2009 when Bitcoin was released and ended in early 2010.

Tiers of blockchain technology

- **Blockchain 2.0**
  - Generation 2.0 blockchains are used by financial services and contracts are introduced in this generation.
  - This includes various financial assets, for example derivatives, options, swaps, and bonds.
  - Applications that are beyond currency, finance, and markets are included at this tier.
  - Ethereum, Hyperledger, and other newer blockchain platforms are considered part of Blockchain 2.0.
  - This generation started when ideas related to using blockchain for other purposes started to emerge in 2010.

Tiers of blockchain technology

Blockchain 3.0

- Generation 3 blockchains are used to implement applications beyond the financial services industry
- Used in more general-purpose industries such as government, health, media, the arts, and justice.
- Again, as in Blockchain 2.0, Ethereum, Hyperledger, and newer blockchains with the ability to code smart contracts are considered part of this blockchain technology tier.
- This generation of blockchain emerged around 2012 when multiple applications of blockchain technology in different industries were researched.

Tiers of blockchain technology

- **Generation X (Blockchain X)**
  - This is a vision of blockchain singularity where one day we will have a **public blockchain service** available that anyone can use just like the Google search engine
  - It will provide services in all realms of society.
  - This is a public open distributed ledger with general purpose rational agents (Machina Economicus) running on blockchain, making decisions and interacting with other intelligent autonomous agents on behalf of humans and regulated by code instead of law or paper contracts.
  - Machina Economicus is a concept which comes from the field of Artificial Intelligence (AI) and computational economics. It can be defined as a machine that makes logical and perfect decisions.

Types of blockchain-Public blockchains

- Blockchains are open to the public
  - Anyone can participate as a node in the decision-making process.
- Users may or may not be rewarded for their participation.
- Ledgers are not owned by anyone and are publicly open for anyone to participate in.
- All users of the permission-less ledger maintain a copy of the ledger on their local nodes and use a distributed consensus mechanism in order to reach a decision about the eventual state of the ledger.
- Public blockchains are also known as permission-less ledgers.

Types of blockchain-Private blockchains

- Private blockchains as the name implies are private
- Open only to a consortium or group of individuals or organizations that has decided to share the ledger among themselves

Private Blockchain vs Public Blockchain

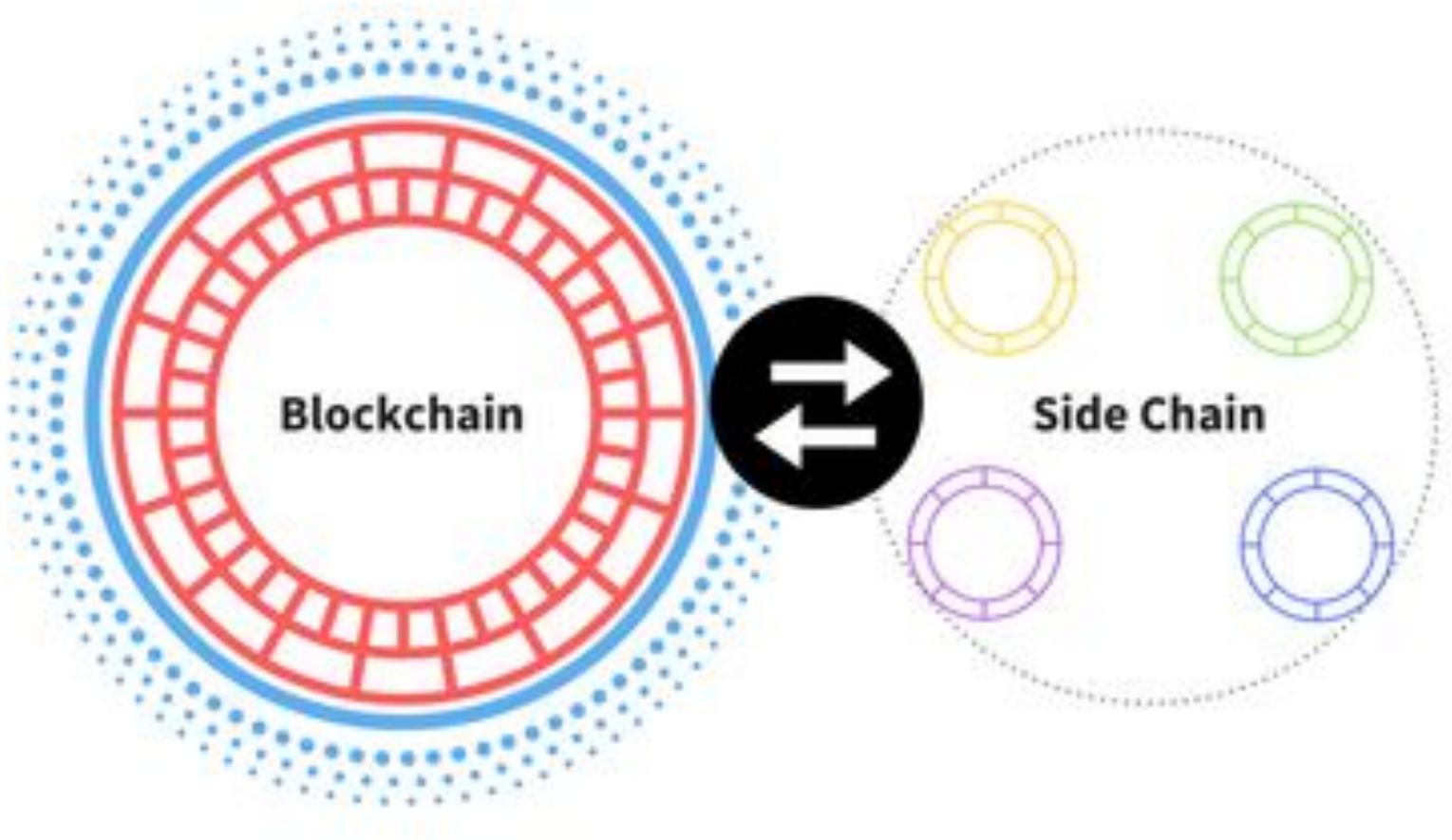
Conditions	Private Blockchain	Public Blockchain
Centralization / Purpose	Semi-Decentralized Business-to-Business	Decentralized Peer-to-Peer
Authentication	Authenticated	Not-Authenticated
Permissions	Permissioned	Permission-less
Advantages	• Support legal entities • Higher performance • Better scalability	• Support anonymity • High immutability • Trustless environment

Types of blockchain-Semi-private blockchains

- Part of the blockchain is private and part of it is public.
- The private part is controlled by a group of individuals
  - whereas the public part is open for participation by anyone

Types of blockchain-Sidechains

- More precisely known as pegged sidechains,
  - This is a concept whereby coins can be moved from one blockchain to another and moved back.
- Common uses include the creation of new altcoins (alternative cryptocurrencies) whereby coins are *burnt* as a proof of adequate stake.
- There are two types of sidechain.
  - The example provided above for *burning* coins is applicable to a one-way pegged sidechain.
  - The second type is called a two-way pegged sidechain, which allows the movement of coins from the main chain to the sidechain and back to the main chain when required.


Blockchain
Side Chain

Types of blockchain-Permissioned ledger

- A permissioned ledger is a blockchain whereby the participants of the network are known and already trusted.
- Permissioned ledgers do not need to use a distributed consensus mechanism
  - instead an *agreement protocol* can be used to maintain a shared version of truth about the state of the records on the blockchain.
- There is also no requirement for a permissioned blockchain to be private as it can be a public blockchain but with regulated access control.

Types of blockchain-Distributed ledger

- This ledger is distributed among its participants and spread across multiple sites or organizations.
- This type can either be private or public.
- The key idea is that, unlike many other blockchains, the records are stored contiguously instead of sorted into blocks.
  - This concept is used in **Ripple**.

Types of blockchain

- **Shared ledger**
  - This is generic term that is used to describe any application or database that is shared by the public or a consortium.
- **Fully private and proprietary blockchains**
  - These blockchains perhaps have no mainstream application as they deviate from the core idea of decentralization in blockchain technology.
  - Nonetheless in specific private settings within an organization there might be a need to share data and provide some level of guarantee of the authenticity of the data.
  - **For example**, for collaboration and sharing data between various government departments.

Types of blockchain

- **Tokenized blockchains**
  - These blockchains are standard blockchains that generate cryptocurrency as a result of a consensus process via mining or via initial distribution
- **Tokenless blockchains**
  - These are probably not real blockchains because they lack the basic unit of transfer of value
  - These are still valuable in situations where there is no need to transfer value between nodes and **only sharing some data among various already trusted parties is required**.

Consensus in blockchain

- Consensus is basically a distributed computing concept that has been used in blockchain in order to provide a means of agreeing to a single version of truth by all peers on the blockchain network.
- Two categories of consensus mechanism exist:
  - Proof-based, leader-based, or the *Nakamoto consensus* whereby a leader is elected and proposes a final value
  - Byzantine fault tolerance-based, which is a more traditional approach based on rounds of votes

Proof of Work

- This type of consensus mechanism relies on proof that enough computational resources have been spent before proposing a value for acceptance by the network.
  - Used in bitcoin and other cryptocurrencies.
- Currently, this is the only algorithm that has proven astonishingly successful against **Sybil attacks**.
  - Sybil Attack : node in the network operates multiple identities actively at the same time and undermines the authority/power in reputation systems.

Proof of Stake

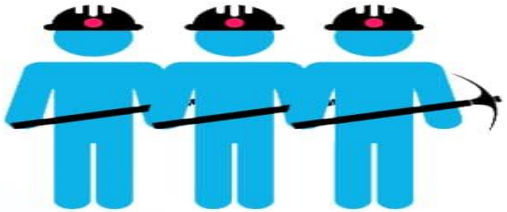
- This algorithm works on the idea that a node or user has enough stake in the system;
  - Example the user has invested enough in the system so that any malicious attempt would outweigh the benefits of performing an attack on the system.
- This idea was first introduced by Peercoin and is going to be used in the Ethereum blockchain.
- Another important concept in **Proof of Stake** (**PoS**) is coin age, which is a derived from the amount of time and the number of coins that have not been spent.
- In this model, the chances of proposing and signing the next block increase with the coin age.

Blockgeeks

Proof of Work vs. Proof of Stake

To add each block to the chain, miners must compete to solve a difficult puzzle using their computers processing power.

In order to add a malicious block, you'd have to have a computer more powerful than 51% of the network.



The first miner to solve the puzzle is given a reward for their work.

There is no competition as the block creator is chosen by an algorithm based on the user's stake.

In order to add a malicious block, you'd have to own 51% of all the cryptocurrency on the network.

There is no reward for making a block, so the block creator takes a transaction fee.

Delegated Proof of Stake

- Delegated Proof of Stake (DPOS) is an innovation over standard PoS
- Each node that has stake in the system can delegate the validation of a transaction to other nodes by voting.
- This is used in the bitshares blockchain

Proof of Elapsed Time

- Introduced by **Intel**
- Proof of elapsed time (POET) is a blockchain network consensus mechanism algorithm that prevents high resource utilization and high energy consumption
- Uses **Trusted Execution Environment** (**TEE**) to provide randomness and safety in the leader election process via a guaranteed wait time.
  - Each participating node in the network is required to wait for a randomly chosen time period,
  - first one to complete the designated waiting time wins the new block.
  - Each node in the blockchain network generates a random wait time and goes to sleep for that specified duration.
  - The one to wake up first – that is, the one with the shortest wait time – wakes up and commits a new block to the blockchain, broadcasting the necessary information to the whole peer network .
- It requires the Intel **SGX** (**Software Guard Extensions**) processor in order to provide the security guarantee and for it to be secure.
  - *Hyperledger* in the context of the Intel Sawtooth Lake blockchain project.

Deposit-based consensus

- Nodes that wish to participate on the network have to put in a security deposit before they can propose a block.
- The protocol governs through the controlling of these security deposits, which implicitly governs the incentives of validators.

Proof of importance

- Proof of importance not only relies on how much stake a user has in the system but it also monitors the usage and movement of tokens by the user to establish a level of trust and importance.
- This is used in Nemcoin.
- Proof importance uses other various metrics in order to evaluate nodes.
  - net transfers
  - amount of vested currency,
  - activity clusters

Federated consensus or federated Byzantine consensus

- Used in the stellar consensus protocol
- Nodes in this protocol keep a group of publicly trusted peers
  - Propagates only those transactions that have been validated by the majority of trusted nodes.

Reputation-based mechanisms

- Leader is elected on the basis of the reputation it has built over time on the network.
- This can be based on the voting from other members.

Practical Byzantine Fault Tolerance

- **Practical Byzantine Fault Tolerance** (**PBFT**) achieves state machine replication
  - which provides tolerance against Byzantine nodes.
- PBFT, PAXOS, RAFT, and Federated Byzantine Agreement (FBA), are also being used or have been proposed for use in many different implementations of distributed systems and blockchains.

CAP theorem and blockchain

- CAP theorem is violated in blockchain,
  - bitcoin,
- In blockchains consistency is sacrificed in favor of availability and partition tolerance.
- In this scenario, **Consistency** (**C**) on the blockchain is not achieved simultaneously with **Partition tolerance** (**P**) and **Availability** (**A**), but it is achieved over time.
  - This is called *eventual consistency*, where consistency is achieved as a result of validation from multiple nodes over time.

CAP theorem and blockchain

- For this purpose, the concept of mining was introduced in bitcoin;
- This is a process that facilitates the achievement of consensus by using a consensus algorithm called PoW.
- Mining can be defined as a process that is used to add more blocks to the blockchain.

Benefits of blockchain

- **Decentralization**
  - no need for a trusted third party or intermediary to validate transactions
- **Transparency and trust**
  - blockchains are shared and everyone can see what is on the blockchain
- **Immutability**
  - Once the data has been written to the blockchain, it is extremely difficult to change it back

Benefits of blockchain

- **High availability**
  - data is replicated and updated on each and every node
- **Highly secure**
  - All transactions on a blockchain are cryptographically secured and provide integrity
- **Simplification of current paradigms**
  - Blockchain can serve as a single shared ledger among interested parties

Benefits of blockchain

- **Faster dealings**
  - Blockchain does not require a lengthy process of verification, reconciliation, and clearance
    - Because a single version of agreed upon data is already available on a shared ledger
- **Cost saving**
  - No third party or clearing houses are required in the blockchain model

Challenges and limitations of blockchain technology

- Scalability
- Adaptability
- Regulation
- Relatively immature technology
- Privacy

References

- Imran Bashir. “Mastring BlockChain”, Packt
- Web Materials