Computer Science Department

University of Massachusetts Dartmouth

Blockchain Demonstration

Senior Design Capstone Project 9

Distributed Consensus Systems Guide

Research Guide for Blockchain, Hashgraph, Hyperledger, Corda, and Tangle Technologies

Client: Bradford T. Doyle, NUWC

Supervisor: Iren T. Valova

Table of Contents

Chapter 1: What Are Distributed Consensus Systems?

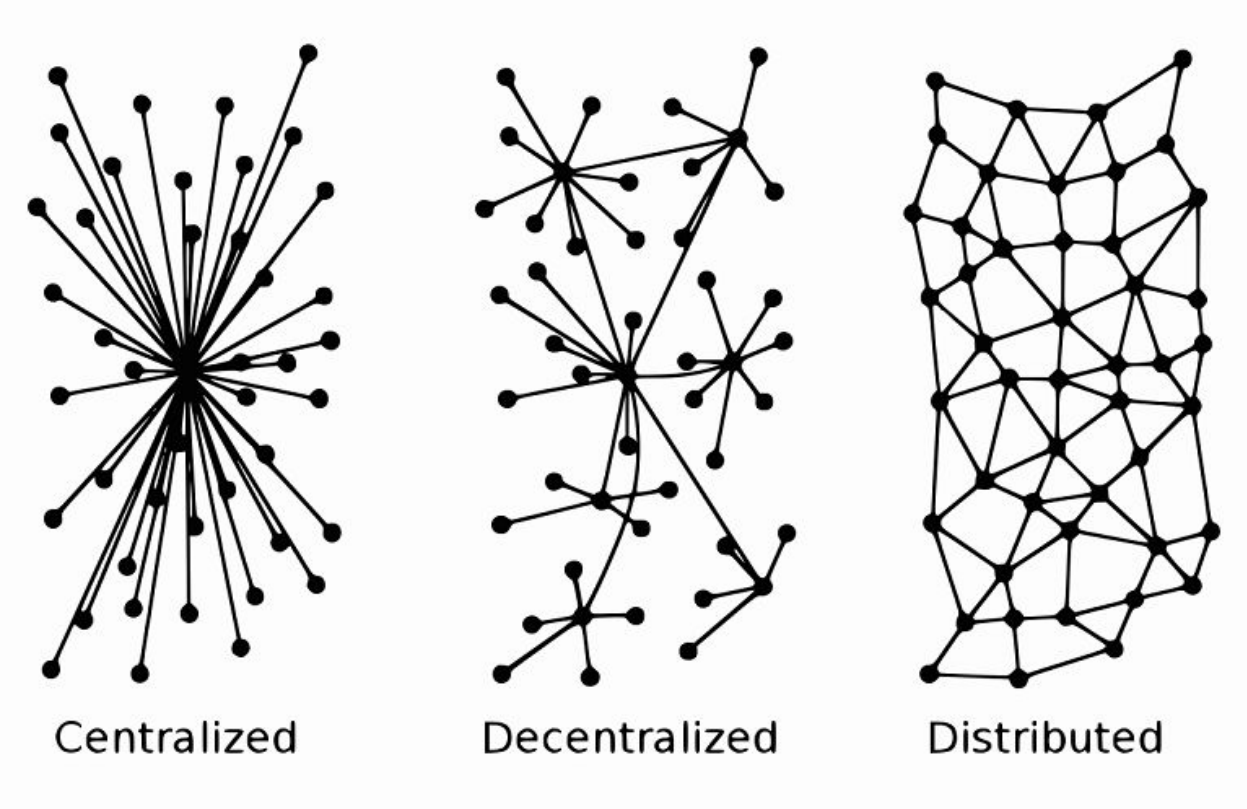
In order to understand what a distributed consensus system is, we must first explain the consensus problem. The consensus problem is an agreement among processes over a single piece of data. This data can be a birth certificate, a vote, or even a user identity. However, some nodes on the network may fail or become unreliable. Therefore, there must be a system put in place to accept the agreement. For example, one consensus mechanism may state that 51% of the nodes on the network must agree on this piece of data before acceptance.

Let's start with the preliminary two-node consensus system. There are two nodes that need to agree on a piece of data such as a transaction or a student record. Once the two nodes agree that this piece of data is valid, a consensus has been achieved on the validity of the data. A distributed consensus system follows the basic protocol of a two-node consensus system. However, it is distributed amongst various nodes on the network instead of two single nodes as with a standard method of consensus. As a result, multiple nodes on the network need to reach an agreement to establish a consensus for a piece of data.

We begin our research with an extensive analysis of Blockchain, Hashgraph, Hyperledger, Corda, and Tangle technologies. We then proceed to an in-depth comparison of these technologies and evaluate them in terms of their ability to prevent data loss and tampering. Finally, we provide various development tutorials for Ethereum private blockchains as well as any methods of troubleshooting potential issues encountered during development. We hope that this guide helps you to get a better understanding of these technologies, the main differences between them, as well as how they can help to prevent data loss and tampering.

Chapter 2: Concepts and Definitions

2.1: Centralization, Decentralization, and Distribution



Source: [81]

Centralization

Centralization is the concept of data being stored in a single location such as a database on an individual server. The main disadvantage of centralization is a single point of failure. If the central server is attacked, then all data will be lost.

Decentralization

Decentralization is the concept of data being stored in multiple locations across all nodes on the network. Some nodes act as servers that provide information to the client nodes. These servers are connected to each other. The main advantage of decentralization is that if one location is attacked and data is lost, then the data still persists on other locations on the network [79].

Distribution

Distribution is the concept of data being stored on all nodes throughout the network. There are no data servers as with decentralization. All nodes on the network are equal and have equal rights in terms of performing data operations [79].

Disadvantages of Centralized Networks [79]

1. Security: If someone has access to the central server that is hosting all of the information, then the data can be added, modified, or even removed. There is a greater risk of data loss and tampering on centralized networks.
2. Reliability: If the server is overloaded with a large number of requests from the clients on the network, then the server can break down and no longer respond.
3. Accessibility: If there is a problem with the centralized storage, then data may become inaccessible for periods of time until the problem is resolved. Also, different clients have different needs and the processes of a centralized network can become inconvenient for a client as a result.
4. Data Transfer Rates: If the nodes on the network are in different countries or continents, then they may have problems connecting with the server.
5. Scalability: Centralized networks are difficulty to scale since server capacity is limited and the network traffic cannot be infinite.

Benefits of Decentralization and Distribution [79]

- Since decentralized networks do not have any centralized storage, they are protected from deliberate attacks or accidental changes to information.
- Decentralized systems are self-sufficient and self-regulating as any changes made to data will be reflected across all nodes on the network and if there are any conflicts, then the data will be backed up to match that of the other nodes.
- Decentralized systems do not have issues with reliability, accessibility, and data transfer rates. All nodes on the network have a copy of the data and all requests are distributed across the network as a result. The pressure doesn't fall on one single node on the network. Therefore, the total network capacity is much larger that that of a centralized network.

- Since the total number of nodes on a distributed or decentralized network is large, Distributed Denial of Service (DDoS) attacks are only possible if their capacity is much larger than that of the network. However, an attack of such magnitude would be very expensive. Therefore, it can be considered that decentralized and distributed networks are safe from DDoS attacks.
- Distributed and decentralized networks also allow for the client to choose the node closest to them and work with the data on that node. Therefore, these networks do not suffer the internet connection issues that a centralized network may face.
- In terms of scalability, centralized networks require all clients to connect to the server and all data is stored on that single server. Therefore, all requests pass through the single server. However, the server has finite resources and can only process a certain number of requests at a time. Decentralized and distributed networks do not face this problem as the request load is shared between several nodes.

2.2: Web 3.0

Web 3.0 is referred to as a third generation of Internet-based services that comprise what could be called the “Intelligent Web.” These services can include machine learning, semantic web, microformats, natural language search, recommendation agents, and artificial intelligence technologies. Web 3.0 is intended to change how websites are created and how people will interact with them [39]. Below is the Abstracted Technology Stack which consists of various services and technologies that will comprise Web 3.0, such as decentralized application browsers, decentralized applications, and consensus algorithms.

Source: [80]

In terms of the Blockchain and other decentralized technologies, distributed computing is a service that will be part of Web 3.0. Web 3.0 is considered to be the new web that has the potential to be the platform for decentralized applications [39]. The transition from our current Web 2.0 to the developing Web 3.0 appears to be from our currently centralized structure, to a partially decentralized, and then to a fully decentralized web [80].

2.3: Byzantine Fault Tolerance

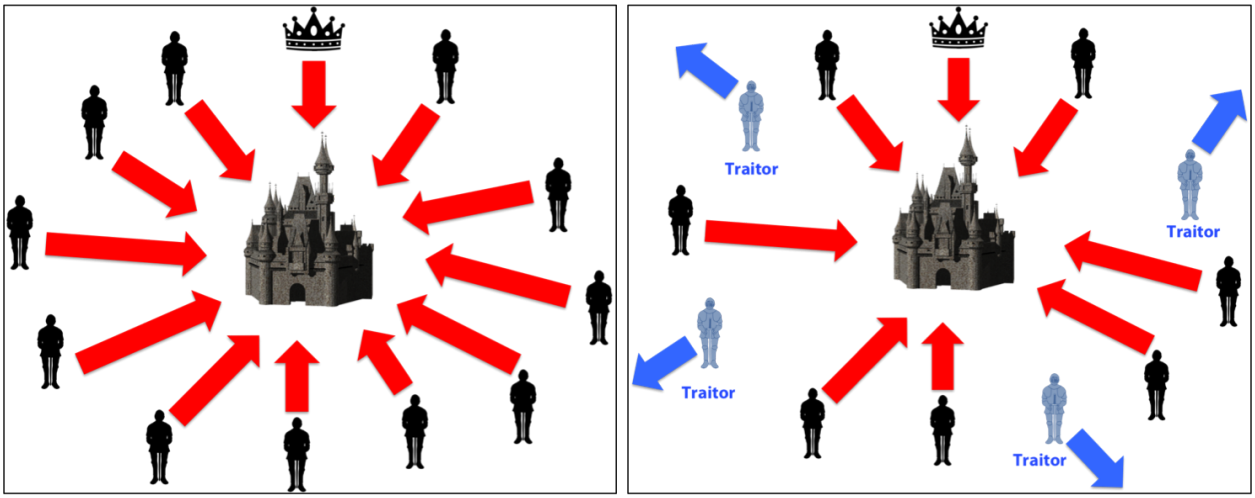
Byzantine Fault Tolerance is the characteristic that defines a system that tolerates the group of failures that belong to the Byzantine Generals' Problem. Byzantine Failure is the most difficult class of failure modes as it implies no restrictions and makes no assumptions of possible node behaviors [32]. To begin understanding the concept of Byzantine Fault Tolerance, we will start with the Byzantine Generals' Problem.

The Byzantine Generals' Problem

The Byzantine Generals' Problem is a generalized version of the Two Generals Problem with a slight modification. The Two Generals Problem is a scenario in which two generals need to agree

on a time to attack their mutual enemy. With the Byzantine Generals' Problem, one or more of these generals can be a traitor and therefore lie about their choice of a time to attack [32].

Let's consider a castle surrounded by several armies that are each controlled by a General. The lead General is indicated by the crown symbol above the castle. The only method of successfully attacking the castle is for every army to attack at the same time. But, there will be command difficulties as there are multiple generals throughout the perimeter of the castle. The Generals will send messages in order to agree on a set time to attack the castle. However, some Generals may be traitors to their armies and relay incorrect attack times in order to prevent a successful attack on the castle. In order to solve the Byzantine Generals' Problem, we must move from a Leader and Follower paradigm in the Two Generals' Problem to a Commander and Lieutenant paradigm. Below is a visual to depict the Byzantine Generals' Problem scenario [32].



Coordinated Attack Leading to Victory **Uncoordinated Attack Leading to Defeat**

Source: [32]

In order to achieve consensus in this scenario, the command and every lieutenant must agree on a mutual decision. This is further explained below.

Byzantine Generals Problem. A commanding general must send an order to his $n - 1$ lieutenant generals such that

IC1. All loyal lieutenants obey the same order.
IC2. If the commanding general is loyal, then every loyal lieutenant obeys the order he sends.

Source: [32]

Building on IC2, even if the commander is a traitor, the consensus must still be achieved. Therefore, all lieutenants take the majority vote. This algorithm can reach consensus as long as ⅔ of the actors are honest. If more than ⅓ of the lieutenants are traitors, then consensus cannot be reached and the armies cannot coordinate their attack [32].

Distributed Consensus Systems and Byzantine Fault Tolerance [32]

In order for a distributed consensus system to be reliable, it must develop a solution to solve the Byzantine Generals' Problem by providing Byzantine Fault Tolerance. Blockchains utilize Proof-of-Work as a solution to providing a reliable Byzantine Fault Tolerance. Proof-of-Work is a consensus protocol that requires nodes that give consensus to calculate a hashing algorithm. This hashing algorithm can only be computed from valid transactions and 51% of the nodes on the network must agree upon the next transaction. This is a functional solution to provide Byzantine Fault Tolerance as a fair amount of computing power is required to generate the correct hash, and in sufficiently large blockchain networks it would require an enormous amount of computing power to disrupt the truth and validity of the blockchain itself. Other consensus algorithms that attempt to provide a solution for Byzantine Fault Tolerance are Proof-of-Stake, Virtual Voting, and IOTA's Tangle consensus.

2.4: Aspects of Distributed Consensus Systems

Decentralization: Distributed technology lets us store data on a network that can be accessed over the Internet. Through the decentralized technology, the owner of the data has direct control through their private key which is directly linked to the data [4]. Anything that happens on the blockchain network is a function of the network as a whole. Therefore, the blockchain is managed by all nodes on its network instead of a single entity as with centralized networks.

Benefits of Decentralization [4]:

- Empowered Users: allows users to keep control of their information and transactions.
- Fault Tolerance: less likely to fail accidentally since they rely on many separate components that are not likely to fail.
- Durability and Attack Resistance: Since the blockchain doesn't have a central point of control and can better survive malicious attacks, decentralized systems are more "expensive" to attack and destroy or manipulate.
- Free from Scams: it is much more difficult for users to cause harm to other users by scamming.
- Removing Third-Party Risks: enables users to make an exchange without a third party as an intermediary which eliminates risks.
- Higher Transaction Rate: transactions can reduce times to minutes and can be processed anytime compared to how transactions are done through banks now.
- Lower Transaction Costs: done through eliminating 3rd party intermediaries and overhead costs for exchanging assets.
- Transparency: changes to public blockchains are viewable by all parties and all transactions are immutable which means they cannot be altered or deleted.
- Authenticity: the blockchain is complete, consistent, timely, accurate, and widely available.

Consensus: For a transaction to be valid, all participants must agree on the validity of the Transaction [5] In order to create a secure consensus protocol, the protocol must be fault tolerant.

Provenance: Participants know where the asset came from and how its ownership changed over time [5].

Immutability: No participant can tamper with a transaction once it has been recorded to the blockchain. If a transaction was made by mistake and a change needs to be made, a new transaction must be used to reverse the error. Both transactions remain on the blockchain [5].

Finality: A single, shared ledger provides one place to go to determine who owns the asset or if a transaction has been completed [5].

Transparency: The data recorded in a blockchain is transparent to each node as well as on updating its data. This allows for more trust in the data being stored [22].

Open Source: The blockchain system is open to everyone. Records can be checked publicly and people can use the blockchain technology to create any application they want [22].

Autonomy: Due to the consensus mechanism, every node on the blockchain network can transfer and update data in a safe manner [22]. No one can intervene in the ability of adding data by bypassing the consensus [22].

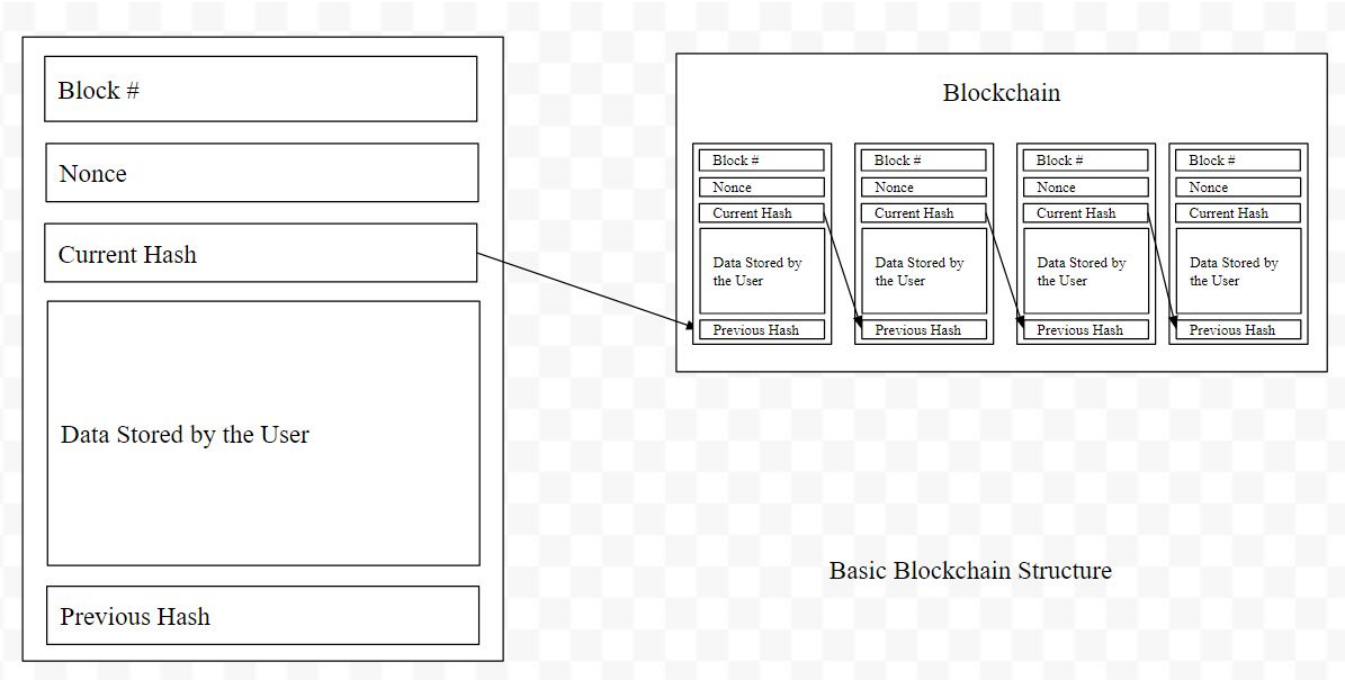
Anonymity: Blockchain technology solves the trust problem between nodes. Data transfers and transactions can be anonymous and only need to know the person's blockchain address [22].

Chapter 3: Blockchain

3.1: What is Blockchain?

Definition of a Blockchain

A blockchain is a shared distributed ledger that allows for the recording of transactions and asset tracking on a network. An asset can be anything from a house, a car, land, or even patents, copyrights, or branding. Anything of value can be tracked and traded on a blockchain network. This reduces the risk and cuts costs for anyone involved in the tracking process. In terms of Bitcoin and the blockchain, think of the blockchain as the operating system and Bitcoin as the application that is running on the operating system. Bitcoin is only the first use case for the Blockchain [5]. Below is a diagram that we created to visually represent the Blockchain structure.



Basic Blockchain Structure

The block on the left consists of the block number, nonce, its current hash, any data stored by the user, and the hash of the previous block. The next block is linked to the previous block through the previous hash. This establishes the connection between all blocks: each hash of each block is calculated using the hash of the previous block.

Blockchains are a new and developing technology that may further protect against data breaches. They were originally devised for Bitcoin, the digital currency, but the technology community is currently studying other potential uses for them. Blockchains allow digital information to be distributed but not copied. Information shared on a blockchain exists as a shared and continually updated database. Information is public, easily verifiable, and is not stored in one single location. Therefore, it makes it extremely difficult for hackers to corrupt the data since no centralized version of this information exists. Since blockchains store blocks of information that are identical across the network, the blockchain can't be controlled by one single entity and it has no single point of failure. Therefore, data breaches may be prevented by potentially storing user information or any other confidential information on a blockchain.

A blockchain is a linked-list-like structure that acts as an immutable ledger. It is distributed among all members of a decentralized network. This ledger tracks and records any arbitrary piece of data in such a way that it cannot be changed. The distributed nature of the blockchain implies that multiple copies exist. The distributed nature is also what lends itself to being immutable. If there are multiple versions of the blockchain on the network, the official ledger is the version that is held by the majority of the peers on the network [18].

Participants on a Blockchain and their Roles [5]

1. Blockchain User: This is typically a business user with permissions to join the blockchain network and perform transactions with other participants. Blockchain users have no awareness of the blockchain technology as it operates in the background. There are typically multiple users on any one business network.
2. Regulator: A blockchain user with special permissions might oversee the transactions on the network. Regulators may be restricted from performing transactions.
3. Blockchain Developer: Programmers who create the applications and smart contracts that allow the blockchain users to conduct transactions on the network. Applications serve as the bridge between users and the blockchain.

4. Blockchain Network Operator: Those who have special permissions and authority to define, create, manage, and monitor the blockchain network. Each business on the network has a network operator.
5. Traditional Processing Platforms: Existing computer systems that might be used by the blockchain to augment processing. This system might also need to initiate requests into the blockchain.
6. Traditional Data Sources: Existing data systems that may provide data to influence the behavior of smart contracts as well as helping to define how communications and data transfer will occur between applications/data on the blockchain.
7. Certificate Authority: An individual who issues and manages different types of certificates required to run on a permissioned blockchain.

5 Key Responsibilities of the Network Node

- Discover fellow peers and compute the genesis block
- Wait for data from a user node to signal the start of the race to compute/mine a block
- Repeatedly randomize or increment the nonce until the block header's hash meets the blockchain's criteria (e.g. a certain number of leading 0's)
- Once a solution is found, share it with the other peers on the network.
- Upon solving their own block, or receiving a block solved by a peer, each node must verify that the block fits onto the blockchain. This is important since nodes do not trust their peers on the network.

The above network structure gives Blockchain its distributed and decentralized properties [18].

Types of Blockchains [24]

Public

Public blockchains are based on the Proof-of-Work (PoW) consensus algorithm. They are open sourced and not permissioned. Anyone can participate on this blockchain without additional permission. Examples of public blockchains include Bitcoin and Ethereum. With public blockchains, anyone can download the code to the blockchain and start running a public node on their own device, validate network transactions, and participate in the consensus process. Anyone can send transactions through a public blockchain network and expect to see them

added to the blockchain as long as the transactions are valid. Public block explorers can be used by anyone to read transactions on the network [24].

Federated/Consortium

Federated blockchains operate under the leadership of a group and do not allow any person with access to the Internet to participate in the transaction verification process. Federated blockchains are faster and provide more privacy in transactions. Consortium blockchains are mainly used in the banking sector. The consensus mechanisms of these blockchains are controlled by a pre-selected set of nodes (e.g. a consortium of 15 financial institutions and each operates a node, 10 must sign every block for it to be valid). The right to read the blockchain can either be public or restricted to the participants on the network [24]. Examples of these blockchains include R3 and Corda.

Private

With private blockchains, write permissions are centralized to one organization. Read permissions can either be public or restricted to a certain extent. Some applications include database management and auditing which are internal to a single company. Private blockchains are a method of taking advantage of blockchain technology by setting up groups who can verify transactions internally. However, this puts you at risk of security breaches as with a centralized system, as opposed to public blockchains secured by incentive mechanisms. However, private blockchains have their use case, especially with scalability and state compliance of data privacy rules [24]. Examples of private blockchains include MONAX and Multichain.

Components of a Blockchain - Hashing and Merkle Trees

Cryptographic Hash Functions

A hash function is a tool that converts input data of variable length to a unique fixed size output known as a digest. Hash functions exhibit pre-image resistance, second pre-image resistance, and collision resistance. Blockchain implementations typically use the SHA-256 or RIPEMD hashing algorithms [18, 98]. With SHA-256, the length of the input does not matter as the output will always be 256 bits in length [97].

Properties of Cryptographic Hash Functions [98, 99, 100]

1. Deterministic: You will always receive the same result no matter how many times a particular input is parsed through the hashing function. This is critical because if a different hashing output was received every time, then it would be impossible to keep track of inputs.
2. Quick Computation: The hashing function should be able to quickly return the hash digest of an input. The system would not be efficient if this process wasn't fast enough.
3. Pre-Image Resistance: Given the hash of an input A (known as H(A)), it is infeasible to determine what the input A is given H(A). It is not impossible to determine the original input given its hash value. However, it should be extremely difficult to do so.
4. Detecting Change: the easiest method of detecting an input change is to compare the message digest of the 2 versions. If they match, then it is guaranteed that no changes have been made to the input.
5. Collision Detection: it is infeasible to find two inputs that will result in the same output.
6. One Way: It is simple to calculate a message digest. However, given the digest, it is near impossible to determine the input.
7. Compression: a large input is essentially compressed into a short string representation of that input.

Hashing Functions in the Blockchain

In order for a Blockchain to function, it has to constantly update itself. All records of transactions and assets must be up-to-date for each member of the network. When updating transactional information, any authenticating system is open to attack. Blockchain uses a cryptographic hashing probability game known as Proof-of-Work to guarantee authenticity [100].

In order for a Blockchain to continue functioning, it must create new blocks. These new blocks are created by the network as a whole to maintain its decentralized structure. The network must arrive at a consensus in order to determine what the new blocks will consist of. To achieve a consensus, miners on the network propose certain blocks, these blocks are verified, and the network chooses a single block to become the next portion of the ledger. However, numerous miners propose identical blocks that are verified.

A specific block is chosen to be the next in the chain by computers competing in a hash game. To win the game, a mining computer has to guess a number known as a nonce. This nonce, in combination with all of the previous data in the blockchain, outputs a certain hash when it is used as input through the SHA-256 hashing function. All of the miners on the network run a similar program searching for the correct nonce which, when it is added to the blockchain data and run through the hashing function, produces a random hash that the blockchain "decides" is the "solution" to the problem. The computers are working together and the hash that has been guessed cannot be guessed again. The first computer to guess the correct nonce wins the right to create the next block and is rewarded in cryptocurrency [100].

This process ensures consensus and prevents attackers from manipulating the system. Since every input produces a completely unique hash output, only a specific nonce combined with the correct previously verified blockchain data will result in the hash that solves the computational equation. If it is inaccurate or fraudulent, then the correct hash cannot be guessed. Therefore, cryptography makes blockchain more secure than any human verifying-bank could be [100].

However, as the number of people dedicating computing power to mining has increased over the years, innovators have designed machines designed only to mine cryptocurrency. These computers can guess hashes at a much faster rate than an average computer, making them able to arrive at correct guesses much faster and mine more cryptocurrency. A new problem arises: as fewer and fewer people can afford computers dedicated to mining, the potential for centralization increases. Blocks are created and cryptocurrency is mined as soon as the solution to the next block is calculated. Someone with enough resources could build a more powerful mining machine than anyone else on the network, and they could mine a large percentage of remaining cryptocurrency much faster than anyone else simply by guessing more hashes in a shorter time [100].

Blockchain's 10 Minute Solution

This computational guessing game self-adjusts to always be difficult enough to be guessed by all computers only every 10 minutes, independent of how powerful the computers are. Certain

criteria is established to determine what the hash function must translate the input into in order to work. Therefore, the more criteria put in place, the more guesses will have to be made to receive that specific output. Essentially, the game gets harder by requiring the right hash to have a certain number of zeros at the beginning [100].

Merkle Trees

A merkle tree is also known as a hash tree and it is a balanced binary search tree. Any parent node is the hash of the concatenation of its children. The tree root is the tallest node and is known as the Merkle Root. A condensed version of all the data stored in the block is the merkle tree. Hashing and merkle trees are the foundation of the immutable ledger [18].

Basic Definitions [97]

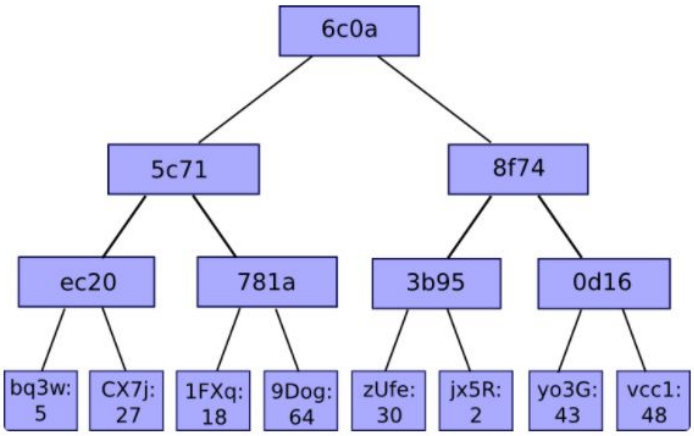
1. Leaf Node: the nodes in the lowest tier of the tree.
2. Child Node: for a single node, the child nodes are the nodes below its tier that are feeding into it.
3. Root Node: the single node on the highest tier.

In terms of the Blockchain, the tree is formed upside-down, beginning with the leaf nodes of the tree. Let's say that we have 4 transactions: A, B, C, and D. The data for each of these transactions is passed through the SHA-256 algorithm 2 times. First, the data is hashed, then the hash of the data is hashed again, resulting in hA, hB, hC, and hD. Once all transactions have been double-hashed, hA and hB are combined and hashed together, and hC and hC are combined and hashed together. Instead of four leaf nodes, we have 2: Hash(hA + hB) + Hash(hC + hD). These two leaf nodes are hashed together to form a single hash that contains all of the data about the original A, B, C, and D transactions. This final hash is called the Merkle root [101].

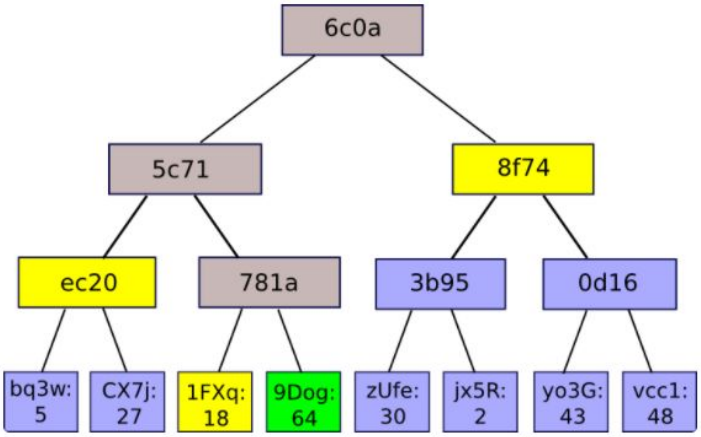
On the Blockchain, every piece of data is linked to the one before it through hashing. All nodes keep a local copy of this blockchain which is constantly updated as new transactions are added. Each new transaction is hashed multiple times into the Merkle tree of an individual block. To validate a new block, nodes will check the hash in the block's header if it contains a reference to the previous block. If the hashes are confirmed, then the new block is added to the blockchain.

Each block contains a link to the block before it through this tree structure, and this links all the way back up to the first transaction, known as the genesis block [101].

A binary merkle tree is the simplest form of a Merkle tree. A bucket always consists of two adjacent hashes. A binary merkle tree is depicted below [93]:



This method of hashing allows for a concept known as Merkle proofs [93]:



A Merkle proof consists of a chunk, the root hash of the tree, and the branch that consists of all hashes going along the path from chunk to root. When reading the proof, someone can verify the hashing for that branch is consistent going up to the tree. Suppose that there's a large database and that the entire contents are stored in a Merkle tree with the root is publicly known and trusted. A user who wants to do a key-value lookup on the database can ask for a Merkle proof. When the proof is received, the user can verify that the value received is actually at the position in the database with that particular root. It allows for a way to authenticate a small amount of

data and this method can be extended to authenticate large databases with a potentially unbounded size [93].

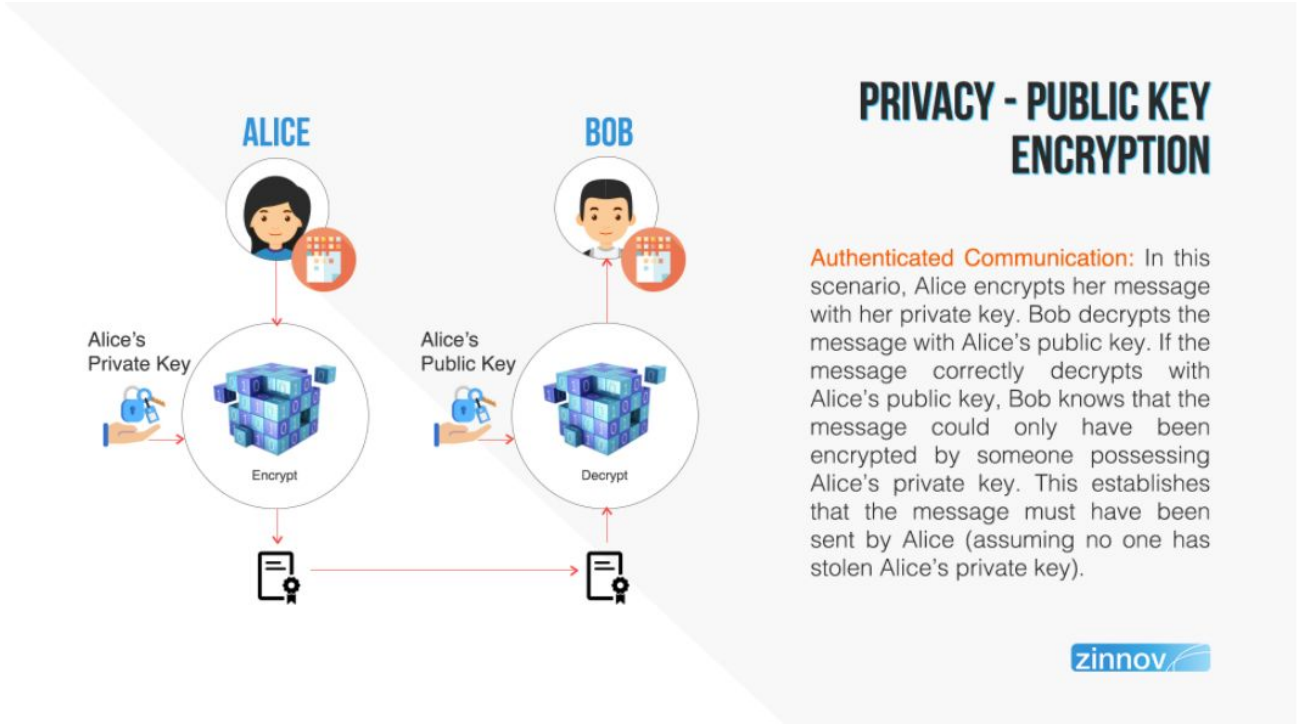
Ethereum Merkle Patricia Tree [92, 93]

Binary Merkle trees are decent data structures for authenticating information in a list format. A state in Ethereum consists of a key-value map with the keys as addresses and the values as account declarations. The state needs to be frequently updated since the balance and nonce of accounts are often changing, new accounts are frequently inserted, and keys are often inserted or deleted. We need a data structure that allows for quickly calculating the new tree root after an insert, update, edit, or delete, without recomputing the entire tree [93].

Ethereum uses a Merkle Patricia Tree which provides a cryptographically authenticated data structure that can be used for storing all (key, value) bindings. Merkle Patricia Trees are also known as Merkle Patricia Tries. Merkle Patricia Trees are fully deterministic which means that a Patricia trie with the same (key, value) binding is guaranteed to be exactly the same and have the same root hash [92]. In the Patricia Tree, the key under which a value is stored is encoded into a "path" that you have to take down the tree. Each node has 16 children, so the path is determined by hexadecimal encoding [93].

Patricia is an acronym that stands for Practical Algorithm to Retrieve Information Coded in Alphanumeric. It is a binary radix trie where a binary choice is made at each node when traversing the trie. Hexadecimal is used in Ethereum. Therefore, nodes in the trie have 16 child nodes and each hex character is referred to as a nibble [96].

Encryption Basics for Understanding the Blockchain [23]



3.2: Applications and Use Cases

The concepts of parallel blockchains and sidechains allow for tradeoffs and improved scalability using independent blockchains which allow for further innovation. Various benefits of blockchains include decentralization, recording and validating every transaction which provides security and reliability, authorization of transactions by miners which make the transactions immutable and prevent hacking threats, and discarding the need for third-party or central authority for P2P transactions [3]. Various companies and institutions are studying blockchains to apply them to various areas such as money transfers, risk management, smart bonds, and cryptocurrencies.

Supply Chain Management

The immutability of the blockchain makes it suitable for tracking goods as they move and change locations in the supply chain. Entries on a blockchain can be used to queue events with a supply chain. Blockchain provides a new way to organize tracking data and putting it to use [12].

Healthcare

Age, gender, and basic medical history data would all be suitable information to be stored on a blockchain in the healthcare industry. None of this information should be able to identify a patient which allows it to be stored on a shared blockchain accessible by numerous individuals without any concerns for privacy. Devices will be able to store data on a healthcare blockchain that can append data to a person's medical record [12].

Blockchain can allow for drug supply chain integrity, patient databases and indexes, claims adjudication, and medical supply chain management. It can also allow for transparency and automation within the patient-to-hospital or patient-to-doctor transactions. Blockchain can provide clinical trial provenance through integrity with an auditable trail of data exchanges. Lastly, Blockchain can provide efficiency, privacy, and ownership of patient health data [82].

Real Estate

The average person sells their house every 5-7 years, and the average person will move almost 12 times in their lifetime. This information could be very useful to store on a blockchain for the real estate industry as it could expedite home sales by quickly verifying finances, reduce fraud as a result of blockchain's encryption, and offer transparency through the selling and purchasing process [12].

Blockchain provides transparency within agreements which can help to verify property information, update and decentralize records, reduce paperwork, digitize transactional processes, as well as recording, tracking, and transferring land titles [82].

Media

Writers and content creators can spread their works on a blockchain and receive immediate payment. Comcast's advanced advertising group developed a new technology to let companies may ad buys on broadcast and over-the-top TV through the blockchain [12].

Blockchain can provide control of ownership rights as well as anti-piracy and preventing copyright infringement. Smart contracts can be used for artist compensation and legal

proceedings. Blockchain can allow for payments processing through cryptographic, security, and anti-3rd party smart contracts which helps to open up content availability internationally [82].

Energy

Blockchains could be used to execute energy supply transactions and provide the basis for metering, billing, and clearing. Other potential applications include ownership documentation, management of assets, guarantees of origin, emission allowances, and renewable energy certificates [12]. Blockchain can bypass public grids to allow for cheaper peer-to-peer energy transfer. It can also allow for smart utility metering [82].

Record Management

National, state, and local governments must maintain records of all individuals such as birth and death dates, marital status, and property transfers. Some of these records only exist in paper form. At times, citizens have to physically go to their local office to make changes which is time-consuming. Blockchain technology can simplify the record keeping process and make it far more secure [12].

Identity Management

Blockchains offer enhanced methods for managing identities as well as digitizing personal documents. Developing digital identity standards is a highly complex process. A universal online personal identity solution requires the cooperation of private entities and government.

Currently, Illinois is experimenting with blockchain technology to replace birth certificates as a form of identification [8]. This would give citizens more control over their data as well as reassurance that their information is more secure than before. The Illinois Blockchain Initiative partnered with identity solutions firm Evernym to create an online ledger that is only accessible to the ID owner and additional granted individuals [8]. This is similar to how the technology could be used to share information between hospitals [8].

Cybersecurity

Cisco believes that the blockchain can play a role in managing networks built on switches, firewalls, and other appliances [7]. Cisco has joined the Hyperledger Project, an open source initiative that is trying to develop cross-industry blockchain technologies. Blockchains can be used to manage switches, routers, firewalls, and internet of things gateways [7]. The blockchain technology is promising where network management is done through a centralized controller [7]. Blockchains could make management tasks possible across multiple vendors by keeping record of an appliance's current state and configuration history which ensures that changes made to a device don't cause a network outage [7].

Blockchain can fight hacking with its immutable ledger as it guarantees validity with data integrity. Since the blockchain has no single point of failure, it can decrease the success of IP-based DDoS attacks [82].

Land Title Registration

Blockchains make record keeping far more efficient and can be applied to recording property titles as well as registered cars.

Connected Cars

Toyota is partnering with MIT to utilize blockchain technology for connected cars. This would allow an unspecified number of people to access data as well as manage payment between cars and charging spots [9]. Blockchains would also allow for accurately preserving vehicle mileage and maintenance records. It can link headquarters to vehicles to manage times and fees for car sharing [9]. This information is key to measuring the value of a vehicle and calculating insurance rates [9]. Blockchains would also allow car-sharing companies to locate any vehicle at any time and determine who to charge for any services [9].

Center for Disease Control

Blockchain technology could allow public health workers respond better to a crisis [10]. The CDC, state and local departments, and other organizations need to routinely share public health data so they can control the spread of various infectious diseases. Blockchains can especially assist with public health surveillance by efficiently managing data during a crisis or allow for better tracking

of opioid abuse [10]. Moving such important data between peers in a secure, compliant, and transparent manner as quickly as possible is key to the business model [10].

For example, let's evaluate the scenario of a pandemic. The CDC has an existing mobile app used by local health workers to log information about patients and determine medications that should be given to various patients. However, personally identifiable information cannot be stored on the cloud. Storing it in an approved manner takes more time. By using the blockchain, storing and sharing data can be done much faster while complying with security and privacy laws [10].

Preventing DDoS Attacks

Hackers use several techniques to instigate an attack such as sending junk requests to a website, increasing traffic until the site can't keep up with requests. The attack goes on until the site gets overwhelmed with requests and crashes. The main difficulty in preventing DDoS is with the existing DNS (Domain Name System). DNS is a partially decentralized one-to-one mapping of IP addresses to domain names and works like an internet phone book [11].

Blockchains would allow for DNS to be fully decentralized and distribute the contents to a large number of nodes which makes it nearly impossible for hackers to attack a network. Domain editing rights would only be granted to domain owners and no one else could make changes. This reduces the risk of data being accessed/changed by unauthorized parties. By using blockchains to protect data, a system can ensure that it is invulnerable to hackers unless every node on the network is wiped clean simultaneously. If current DNS would operate on the blockchain, users would still be able to register domain names but only authorized users could make changes to their domains. Data would be stored on various nodes and every user on the network would have a copy. It would be virtually impossible to hack/destroy it completely [11].

Safeguarding Data

Blockchains allow for the distribution of every piece of data to nodes throughout the system. If someone tries to alter the data, the system analyzes the whole chain and compares them to the meta-packet and excludes any that don't match up. The only way to wipe out the entire blockchain is to destroy every node on the network. Even if just one node remains on the system,

the whole system can be restored despite all the other nodes being compromised [11].

Preventing Fraud and Data Theft

Blockchains prevent potential fraud and decrease the chance of data being stolen/compromised. In order to destroy/corrupt a blockchain, a hacker would have to destroy the data stored on every user's computer in the blockchain network. This can sometimes be millions of computers. It is nearly impossible for a hacker to simultaneously bring down an entire blockchain network. Even if the Blockchain were impacted by an attack, nodes not affected by the attack would continue to run as normal on the network. Bigger blockchain networks with more users have an infinitely lower risk of getting attacked by hackers since the complexity is higher to penetrate such a network [11].

Distributed Storage, Recordkeeping, and Peer-to-Peer Sharing

Users are able to store all of the data in their network on their computer if they choose to do so. This results in earning money for renting "extra" storage space and they can ensure that the chain won't collapse. If a hacker tries to tamper with a block, the whole system analyzes every block of data to find the one that differs from the rest. If the system finds this kind of block, it simply excludes it from the chain and identifies it as false. There is no central authority or storage location. Every user on the network stores some/all of the blockchain and everyone is responsible for verifying data to make sure false data can't be changed and existing data can't be removed [11].

Wills and Inheritances

Smart contracts can be used to determine the validity of a will and allocate inheritances [82].

Automotive

Blockchain can be used to track the truthful and complete history of a vehicle from pre-production to sale. It can also be used for supply chain parts management [82].

Banking, Financial, and Fintech

Blockchain can allow for the streamlining of payments processing with high efficiency, fast, and secure transactions. This can empower global transactions and tear down national currency borders. Blockchain can also minimize the auditing complexity for any financial ledger [82].

Charity

Blockchain can be used to track donation allocation, accountability, and integrity. It can also reduce overhead and the complexity of processing donation payments [82].

Cloud Storage

Blockchain can provide increased security with a shift from a centralized to decentralized network. Blockchain can also lower transactional costs within a decentralized network [82].

Credit History

Blockchain can make credit reports more accurate, transparent, and accessible [82].

Donations

Blockchain can provide an auditable trail for donations to prevent fraud. It can also ensure crowdfunded campaigns receive donations and contributors are compensated [82].

Education

Blockchain can provide digitization and verification of academic credentials. It can also provide a federated repository of academic information specific to class, professor, and student [82].

Forecasting

Blockchain combined with machine learning algorithms can provide a decentralized forecasting tool [82].

Government and Voting

Blockchain can provide major benefits with voting such as reducing voter fraud and inefficiencies with verifiable audit trails. It can minimize government fraud and digitize most processes.

Blockchain has the capability to increase accountability and compliance for government officials. Lastly, Blockchain can provide identity validation and integrity of citizen registration data [82]. Blockchains are fault tolerant, do not allow someone to change the past records, hack the present records, or alter access to the system [12]. Every node on the blockchain with access can see the same results and every vote on the blockchain can be irrefutably traced to its source without sacrificing voter anonymity [12]. End-to-end verifiable voting systems will give the voter the ability to verify that their vote was correctly recorded and counted [12]. Blockchains would especially be useful in the event that a ballot was missing, in transit, or modified and it can be detected by the voter and caught before the election is over [12].

Gun Safety

Blockchain can provide tracking for gun ownership and possession related information as well as the tracking of criminal ID history and attempts made by criminals to purchase guns [82].

Human Resources

Blockchain can allow for an easier method to perform background checks, verification of identity, and employment history. It can also provide payments and benefits process validation through smart contracts [82].

Insurance

Blockchain can improve multi-party contracts, streamline risk contract efficiency and claims adjudication, and reduce disputes through its transparency of shared data [82].

Preventing Data Breaches

One of the main benefits of the blockchain is decentralization which eliminates the risk with data being held in a central location since they store data across its network. Another benefit is recording and validating every transaction which provides security and reliability. Lastly, authorization of transactions by miners makes the transactions immutable and prevents hacking threats. Since blockchains store blocks of data that are identical across the network, they cannot be controlled by a single entity and they have no single point of failure [1]. The blockchain network operates on a consensus mechanism that automatically checks in with itself every 10

minutes and reconciles every transaction that happens in these 10 minute intervals [1].

Blockchains have the ability to improve cyber defense since they can secure and prevent fraudulent activities and detect data tampering based on their underlying characteristics of immutability, transparency, auditability, data encryption, and operational resilience [2]. Computer hackers will no longer be able to exploit central points of vulnerability when networks implement the blockchain for data [1]. Blockchain uses encryption for security with public and private keys as a basis. A public key is a long and randomly-generated string of numbers and is used as the user's' address on the blockchain [1]. Any actions done by the user on that network get recorded as belonging to that specific public key [1]. The private key is used as a "password" that gives its owner access to assets on the network [1].

IOT

Blockchain provides the ability for IoT applications to contribute transactional data and can allow for implications across industries such as trucking, transportation, and supply chain integrity [82].

Law Enforcement

Blockchain can help provide integrity of evidence as well as resistance to the falsification of case data. The blockchain allows for time-stamped documentation and a chronological chain of facts [82].

Legal

Smart contracts with defined rules, expiration, and accessibility through the Blockchain can be created for relevant parties [82].

Marketing

Blockchain can provide more cost-efficient advertising since it allows for bypassing of intermediary parties [82].

Media

Blockchain can provide control of ownership rights as well as anti-piracy and preventing copyright infringement. Smart contracts can be used for artist compensation and legal proceedings. Blockchain can allow for payments processing through cryptographic, security, and anti-3rd party smart contracts which helps to open up content availability internationally [82].

Music Streaming

Blockchain can prevent the illegal downloading of music and can provide compensation for purchased songs to artists [82].

Public Transportation/Ride Sharing

Blockchain can streamline public transportation and provide a more accurate method of payment for ride, gas, and wear and tear [82].
Blockchain can be used to track journey stops and can be paired with IoT to create an immutable ledger of trip data [82].

Travel

Blockchain can allow for a simpler method of passenger identification, boarding, passport, payment, and other documentation digitization and verification. It can also provide loyalty programs digitization and tracking [82].

Current Real-World Applications of the Blockchain [87, 88]

Cybersecurity

- Guardtime is a company that is currently using Blockchain to create keyless signature systems that is being used to secure health records of 1 million Estonian citizens.
- REMME is a decentralized authentication system with the goal of replacing logins and passwords with SSL certificates that are stored on a blockchain.

Healthcare

- Gem is a startup working with the CDC to put disease outbreak data onto a blockchain to increase the effectiveness of disaster relief and response.

- SimplyVital Health is currently developing two health-related blockchain products. ConnectingCare is intended to track the progress of patients after they leave the hospital. Health Nexus aims to provide decentralized blockchain patient records.
- Medrec is an MIT project involving blockchain electronic medical records that are designed to manage authentication, confidentiality, and data sharing.

Finance Services

- ARBA is a cryptocurrency wallet that uses the Bitcoin blockchain to hold and track balances stored in different currencies.
- Bank Hapoalim is a collaboration with the Israeli bank and Microsoft to create a blockchain system to manage bank guarantees.
- Barclays has launched various blockchain initiatives involving tracking financial transactions, compliance and combating fraud.
- Maersk is a shipping and transport consortium that intends to use the blockchain as a solution for streamlining marine insurance.
- Aeternity is currently using smart contracts that become active when network consensus agrees that conditions have been met which will allow for automated payments to be made when parties agree that conditions have been met.
- Augur is using blockchain-based predictions markets for the trading of derivatives and other financial instruments in a decentralized ecosystem.

Manufacturing and Industrial

- The Provenance project aims to provide a blockchain-based provenance record of transparency within supply chains.
- Jiacoin is a cryptocurrency developed by Reliance Industries in India that aims to provide blockchain-based supply chain logistics.
- SKUChain is a blockchain system for allowing tracking and tracing of goods as they pass through a supply chain.
- Blockverify is a blockchain platform being used for anti-counterfeit with initial use cases in diamond, pharmaceuticals, and luxury goods markets.
- Transactivgrid is using blockchain in Brooklyn to allow members to locally produce and cell energy, with the goal of reducing costs involved in energy distribution.

- STORJ.io is a distributed and encrypted cloud storage blockchain that allows users to share unused hard drive space.
- UPS and Fedex have joined the Blockchain in Trucking Alliance (BiTA) to push for increased transparency among all groups involved in the supply chain. BiTA is hoping to develop blockchain standards for the freight industry.

Government

- Dubai is using blockchain to investigate opportunities in health records, shipping, business registration, and preventing the spread of conflict diamonds.
- The Estonian government has partnered with Ericsson for an initiative to create a new data center to move public records onto the blockchain.
- Samsung is currently creating blockchain solutions for the South Korean government to be used in public safety and transport applications.
- Govcoin is a UK Department of Work and Pensions project that is investigating the use of blockchain to record and administer benefit payments.
- Followmyvote.com allows the creation of secure and transparent voting systems to reduce opportunities for voter fraud and increasing turnout through improving accessibility to democracy.

Retail

- OpenBazaar is a blockchain project with the attempt to build a decentralized market where goods and services can be traded with no middleman.
- Loyyal is a blockchain-based universal loyalty framework that aims to allow consumers to combine and trade loyalty rewards.
- Walmart has used blockchain to allow employees to track products back to their origins. After scanning a product, employees can see which location/warehouse the product came from and its current placement in the backroom. This has the potential to streamline the restocking process.

Real Estate

- Ubiquity is a startup that is creating a blockchain-driven system for tracking the legal process during a real estate transfer.

Transport and Tourism

- IBM Blockchain is going public with a number of non-finance related blockchain initiatives in 2018.
- Arcade City is an application that aims to beat Uber by moving ride sharing and car hiring onto the blockchain.
- Webjet is an online travel portal that is developing a blockchain solution to allow empty hotel rooms to be tracked and traded with payment routed to the network.

3.3: Blockchain and Security

Blockchains eliminate the risk with data being held in a central location since they store data across its network. Computer hackers will no longer be able to exploit central points of vulnerability when networks implement the blockchain for data. Blockchain uses encryption for security with public and private keys as a basis. A public key is a long and randomly-generated string of numbers and is used as the user's address on the blockchain. Any actions done by the user on that network get recorded as belonging to that specific public key. The private key is used as a "password" that gives its owner access to assets on the network [1].

Various benefits of blockchains include decentralization, recording and validating every transaction which provides security and reliability, authorization of transactions by miners which make the transactions immutable and prevent hacking threats, and discarding the need for third-party or central authority for P2P transactions (LTP). They have the ability to improve cyber defense since they can secure and prevent fraudulent activities and detect data tampering based on their underlying characteristics of immutability, transparency, auditability, data encryption, and operational resilience [2]. Computer hackers will no longer be able to exploit central points of vulnerability when networks implement the blockchain for data storage [1]. Various companies and institutions are studying blockchains to apply them to various areas such as money transfers, risk management, smart bonds, healthcare, network security, government, and cryptocurrencies.

Our personal information is highly vulnerable when it is stored in online databases. The argument for placing storing our personal information on a Blockchain stems from the belief that it would give us more control over our own information and proper applications would allow us to present just the bare minimum of information needed to identify ourselves [2]. Jerry Cuomo, IBM

Fellow and VP of blockchain technologies, claims that placing our personal information on the blockchain would ensure privacy and trust and allow transactions to be secure, authenticated, and verifiable and endorsed by permissioned users [2].

There are two main types of blockchains: public and private. Public Blockchains are permissionless, meaning that data is publically available to anyone who wishes to participate on the network [2]. Private Blockchains are permission based platforms established by groups or individual firms, or divisions within an organization. Data can only be accessed by those users who are part of the group and are properly authenticated [2]. Thus, blockchains can help with digital identities and maintaining data integrity.

Blockchains use encryption for security with public and private keys as a basis. A public key is a long and randomly-generated string of numbers and is used as the user's address on the blockchain [1]. Any actions done by the user on that network get recorded as belonging to that specific public key [1]. The private key is used as a "password" that gives its owner access to assets on the network [1]. Since blockchains store blocks of data that are identical across the network, they cannot be controlled by a single entity and they have no single point of failure [1]. The blockchain network operates on a consensus mechanism that automatically checks in with itself every 10 minutes and reconciles every transaction that happens in these 10 minute intervals [1]. 51% of users need to agree a transaction is valid before it is added to the blockchain (consensus model protocol). Blockchains are constantly adding new blocks independent of receiving new data in order to prevent against 51% attacks which occur when an attacker is trying to gain ownership of 51% of the blockchain in order to modify data.

Blockchain builds trust through the following attributes [5]:

1. Distributed and Sustainable: The ledger is shared, updated after every transaction, and selectively replicated among participating nodes in near real time. Since it isn't owned by a centralized figure, the blockchain platform's continued existence doesn't depend on an individual entity.
2. Secure, Private, and Indelible: Permissions and cryptography prevent unauthorized access to the network and ensure that all identities of its participants are valid. Privacy is

maintained through cryptography and/or data partitioning techniques to allow selective visibility to certain nodes on the ledger; both transactions and their node identities can be masked. After conditions are agreed upon, participants cannot tamper with the transaction record; they can only create a new transaction to reverse the error.

3. Transparent and Auditable: Participants in a transaction have access to the same records. Therefore, they can validate transactions and verify identities/ownership without the need for a third-party intermediary. Transactions are also time-stamped and can be verified in near real time.
4. Consensus-Based and Transactional: All relevant network nodes must agree on the validity of a transaction. This is achieved using consensus algorithms. Each blockchain network can establish a set of conditions under which a transaction or asset exchange may occur.
5. Orchestrated and Flexible: Since business rules and smart contracts can be built into the blockchain platform, blockchain business networks can evolve as they mature to support end-to-end business processes as well as a wide range of other activities.

The main security components that are evaluated when making software secure are the CIA triad as well as Authentication, Authorization, and Accounting.

CIA Triad

- Confidentiality: Full encryption of blockchain data ensures that it won't be accessed by unauthorized parties while this data is in transit. End-to-End Encryption ensures that only those who have authorization to access the encrypted data i.e. through their private key can decrypt and see the data [2].
- Integrity: The data is verified to be true because it was authorized by a majority of nodes.
- Availability: When it comes to blockchain information availability differs based on the design of each one. But it has been shown to be possible to have no fee, instantaneous transfer of information.

Confidentiality

The CIA triad represents confidentiality, integrity, and availability. Confidentiality is defined as the property that sensitive information is not disclosed to unauthorized individuals, entities, or processes [2]. The main concern of confidentiality is ensuring only interested and authorized parties can access the correct and appropriate data to them [2]. If an attacker can gain access to the blockchain network, then they are more likely to gain access to the data. Therefore, authentication and authorization controls need to be implemented [2]. Various blockchain implementations are starting to address data confidentiality and access control challenges by providing full block data encryption and AAA capabilities. [2]. Full encryption of blockchain data ensures that it won't be accessed by unauthorized parties while this data is in transit.

With public blockchains, there's no requirement to control network access since the chain protocols allow anyone to access and participate in the network [2]. Private blockchains require that appropriate security controls are in place to protect network access. Blockchains can provide advanced security controls by leveraging the public key infrastructure (PKI) to authenticate and authorize parties, and encrypt their communications [2].

Today, if an attacker gains access to the blockchain network and its data, this doesn't always mean that the attacker can read or retrieve the information stored on the blockchain. Full encryption of data blocks can be applied to guarantee its confidentiality. End-to-End Encryption ensures that only those who have authorization to access the encrypted data i.e. through their private key can decrypt and see the data [2]. Using encryption keys along with PKI on a blockchain can provide higher levels of security. Implementing secure communication protocols on a blockchain guarantees that even when an attacker tries to do a man-in-the-middle attack, the attacker won't be able to either forge the interlocutor's identity or disclose any data while in transit [2].

Integrity

Integrity is defined as guarding against improper information modification or destruction. It includes ensuring information nonrepudiation and authenticity [2]. This may consist of data encryption, hash comparison, or digital signatures. Blockchains have built in immutability and traceability that provide means for data integrity. They are secure from the perspective that they

enable users to trust that the transactions stored on them are valid [2]. Blockchains use sequential hashing, cryptography, and have a decentralized structure. These 3 characteristics make it very challenging for any party to tamper with it compared to a standard database [2]. In addition, 51% of users need to agree a transaction is valid before it is added to the blockchain (consensus model protocol).

Every transaction added to a blockchain (private or public) is digitally signed and timestamped. The organization therefore can trace back to a specific time period for each transaction and identify the party through their public address on the blockchain [2]. Nonrepudiation is the assurance that someone can't duplicate the authenticity of their signature on a file or the authorship on transaction that they originated [2]. It increases the reliability of the system (detection of tamper events or fraudulent transactions) since every transaction is cryptographically associated to a user. Any new transaction added to the blockchain will result in a change of the global state of the ledger. The implication is that with every new iteration of the system, the previous state of the system will be stored which results in a fully traceable history log [2]. This provides entities with an extra level of reassurance that data is authentic and hasn't been tampered with.

Availability

Availability means ensuring timely and reliable access to and use of information. The Distributed Denial of Service (DDoS) is the most common attack and can cause the most disruption to internet services as websites are disrupted and applications become unresponsive [2]. DDoS attacks on blockchain networks are costly since they attempt to overpower the network with large volumes of small transactions. The decentralization and peer-to-peer characteristics of the blockchain make it harder to disrupt than client-server architectures. Blockchains are also subject to DDoS attacks but they are able to withstand them a lot better than regular network architectures. Since Blockchains have no single point of failure, this decreases the chances of an IP-based DDoS attack disrupting normal operation. If a node is taken down, data is still accessible via other nodes on the network since they all have a full copy of the blockchain at all times [2].

Bitcoin has withstood cyber-attacks for more than 7 years. Organizations can still face risks from

external events outside of their control despite using blockchains (i.e. global internet outage) [2]. Blockchains have operational resistance due to their combination of peer-to-peer nature and number of nodes on the network, operating in a distributed and 24/7 manner. Organizations can make a node under attack redundant while the rest of the nodes on the network operate as usual. Even if a major part of the blockchain is under attack, it'll operate as normal due to the distributed nature of the technology. Obviously, blockchains aren't "bullet-proof." But, they exhibit various attributes that can help to prevent data breaches on a network [2].

Authentication, Authorization, and Accounting

- Authentication: To be a useful node you must hold a true version of the blockchain on that node. In order to utilize most blockchains you must have an account authenticated with identification information.
- Authorization: More than half of the nodes must authorize a change.
- Accounting: All updates are stored on the blockchain.

Blockchain's security features protect against cybercrime, fraud, and tampering of data. If a network is permissioned, then it allows for the ability to create a members-only network with proof that their members are accurately identified and that goods/assets traded are exactly as represented [5].

The main difficulty in preventing DDoS is with the existing DNS (Domain Name System). DNS is a partially decentralized one-to-one mapping of IP addresses to domain names and works like an internet phone book [11]. Blockchains also help to prevent against DDoS attacks since they would allow for DNS to be fully decentralized and distribute the contents to a large number of nodes which makes it nearly impossible for hackers to attack a network [11]. By using blockchains to protect data, a system can ensure that it is invulnerable to hackers unless every node on the network is wiped clean simultaneously. If current DNS would operate on the blockchain, users would still be able to register domain names but only authorized users could make changes to their domains. Data would be stored on various nodes and every user on the network would have a copy. It would be virtually impossible to hack/destroy it completely [11].

Security Benefits of Permissioned Blockchain Networks [5]:

1. Enhanced Privacy: Through the use of permissions and member IDs, users can specify which transaction details they want other members to be allowed to view. Permissions can be expanded to special users who may need access to further details of the transaction.
2. Improved Auditability: A shared ledger serves as a single source of truth that improves the ability to monitor and audit transactions.
3. Increased Operational Efficiency: Digitization of asset streamlines the transfer of ownership. As a result, transactions can be conducted at a speed more in line with the pace of a business.

Attacks on the Blockchain Network

Consensus Attack

A consensus attack occurs when a majority of the nodes on the network work together to commit a Shallow Block Attack to commit invalid data to the Blockchain. This attack takes advantage of the fact that the official ledger is the one present at a majority of the network's peers [18]. A high number of peers are the best method of countering a possible consensus attack since it is more costly to attack a network with a large number of peers. It would also be beneficial to keep track of who is allowed to become a miner on the network [18].

Shallow Block Attack

A shallow block attack occurs when a single malicious node computes blocks containing malicious data that fit into specific locations on the chain, and then swaps them with actual blocks. This attack is called "shallow" because it typically targets blocks at the top of the chain since too much hashing power would be needed to compute a block that is deep in the Blockchain [18]. There isn't much you can do to counter shallow block attacks. However, in order for these to be effective, they must be coupled with a consensus attack and the counters for that attack are listed above [18].

51% Attack and Double-Spending

If network nodes (miners) receive the majority of the rewards when successfully solving hashing algorithms, then they will be able to control the consensus and include only their data in the blockchain. It is very difficult to change information that has already been added to the blockchain. An attacker can conduct an attack known as the double-spending attack to spend more money than they currently possess. This is done by creating several transactions using the same coins. In theory, the network will consider unnecessary transactions to be incorrect and reject them, and miners won't include them into the chain. If an attacker gets the right to mine a block, then they can include double-spending information into this block. Honest miners will split the blockchain and build a valid parallel branch in response. However, if the attacker has 51% of the network computing power, they can influence the consensus on the network and build their own blockchain with double-spent transactions that will be considered true even though they are actually incorrect [103].

Sybil Attacks

A Sybil attack implies the event in which one node on the network acquires multiple identities. This attack is based on the fact that peer-to-peer networks can't reliably distinguish between members. An attacker may try to fill the network with nodes controlled by themself. This allows them to refuse to transmit and receive blocks by disconnecting other users from the network, execute 51% and double spending attacks, and viewing all transactions using special software tools. A number of heuristic rules are used to Sybil attacks. For example, the system can require that only a limited number of accounts can be created from the same IP within a set time interval. Another option could be to use a trusted certification authority to verify all users on the network. On Bitcoin, Sybil attacks are eliminated by implementing special requirements that rule the generation of new blocks. The ability to generate blocks must be proportional to the processing power of the Proof-of-Work mechanism. Therefore, the attacker cannot fake their own computing power and as thus, cannot execute a Sybil attack [103].

DDoS Attacks

On a Blockchain network, a Distributed Denial of Service (DDoS) attack would be executed by sending a large number of similar requests/transactions. To complicate the clogging of nodes memory, block sizes on Bitcoin are limited to 1MB, and the size of each script doesn't exceed

10,000 bytes. The number of confirmations that each block can process is limited to 20,000 and the number of multi-confirmations is also limited to 20 keys. With these limitations on the network, it would be infeasible to execute a DDoS attack on the network [103].

Quantum Computing

Experts believe that in 10 years, quantum computers will surpass the classical systems in regards to power. Quantum algorithms, in theory, will be able to break RSA-encryption and digital signatures used in blockchain networks. Developers are currently working on solutions to withstand the power of quantum machines. The Quantum Resistant Ledger team is working on cryptographic algorithms that are based on hash functions to make it more complex and stable [103].

3.4: Smart Contracts and Ethereum

Ethereum Virtual Machine and Smart Contracts

Smart Contracts are accounts holding objects on the Ethereum blockchain [6]. A smart contract is a piece of code that is executed on all nodes across the network. All nodes must execute the code and create the same result, verifying the code was executed as it should be. But what is the motivation for nodes to perform this code? If you are a developer and you create a smart contract, then you have to declare the gas price you'll pay computers to execute the code. The gas price is determined by the size of the smart contract. This encourages smart and efficient coding and also encourages miners to continue running nodes [6].

ERC20 Tokens (Ethereum Request for Comments)

ERC20 Tokens, or Ethereum Standard tokens, are used within ethereum smart contracts. ERC20 defines a common list of rules that an ERC20 token must implement, which allows developers to implement coins in a general manner with the Ethereum ecosystem. ERC tokens are one of the most used resources in smart application development [6].

3.5: Performance and Maintenance

Computer Maintenance

To ensure blockchain performance, the blockchain must have multiple nodes constantly verifying the blockchain at all times. These nodes are called miners. Depending on which blockchain platform you use, the specifications required to run a full node and the speed of consensus will differ.

For the Ethereum blockchain using GETH (Go Ethereum), you can sync to the blockchain at three different levels [36]:

- "Full" Sync: Gets the block headers, the block bodies, and validates every element from genesis block.
- Fast Sync: The goal of fast sync is to utilize bandwidth rather than processing power. Instead of processing the entire blockchain like a full sync, it downloads the transaction receipts and pulls the entire recent state database.
    - The fast sync can only be done on a nodes initial sync.
    - The fast sync is susceptible to a sybil attack. An attacker could isolate a single node and feed it a false truth making it believe it is in the correct state. Because fast sync skips the transaction processing, it is easier for an attacker to launch a sybil attack against a node in a fast sync. This is another reason fast sync is allowed only on a nodes initial sync.
- Light Sync: Gets only the current state. To verify elements, it needs to ask to full (archive) nodes for the corresponding tree leaves.

Blockchain Performance

If two blocks are mined at the same time, then the chain will fork until the community can decide on which branch to extend [27]. If the blocks are added slowly, then the community can add to the longer branch and stop growth on the other branch which can be pruned and discarded since it is stale [27]. This is inefficient since some blocks can be mined but discarded anyway. This also means that its necessary to slow down the speed of mining blocks so the community can jointly prune them faster than new branches sprout which is the purpose of proof-of-work: by requiring that miners solve difficult computation problems to mine a block, it can guarantee that the entire

network will have long enough delays between mining events [27]. Proof-of-work blockchains require that electricity be wasted on extra computations and even expensive mining machines may need to be purchased [27].

3.6: Capabilities and Limitations

When utilizing a blockchain network, a large network of users is required to ensure that it is robust [16]. A blockchain with only a few users puts it at greater risk of failure as well as security risks such as a 51% attack. For example, a blockchain with 100 users versus one with 1000 is more likely to experience failures if users begin to disappear or change the blockchain data. The large user base ensures the robustness of the blockchain, as authenticity of transactions is guaranteed if a very large amount of users can confirm it [16].

Consensus Algorithms

Proof-of-Work

Proof-of-Work is the most common consensus algorithm. In Proof-of-Work, miners compete to add the next block by racing to solve a difficult cryptographic puzzle. The first node to solve the puzzle wins and is rewarded in cryptocurrency for their efforts. However, common criticisms include that Proof-of-Work requires a large amount of computational energy, cannot scale well as transactions require 10-60 minutes to be confirmed, and the majority of mining is centralized in the world where electricity is low-cost [97].

Proof-of-Stake

Proof-of-Stake is the most common alternative to Proof-of-Work. Instead of investing in expensive computer equipment to mine blocks, a validator in a Proof-of-Stake network invests in the coins of the system. No mining/coin creation exists in Proof-of-Stake. Instead, all of the coins on the network exist from day one and validators (aka stakeholders since they hold a stake in the system) are paid strictly in transaction fees. Your chance of being picked to create the next block relies on the fraction of coins in the system that you own (the amount of stake that you hold). Once a validator creates a block, the block needs to be stored on the blockchain. Each Proof-of-Stake system handles this process differently depending on its implementation. Some

systems choose to have every node in the system sign off on the block until a majority vote is reached. Other systems may choose a random group of signer nodes [97].

However, Proof-of-Stake is not free of problems. The 'nothing-at-stake' problem occurs when a participant with nothing to lose has no reason to not behave maliciously. This problem arises when there is nothing to discourage a validator from creating two blocks and claiming two sets of transaction fees while the signer signs both of the blocks. One proposed suggestion is to require a validator to lock currency in a virtual vault. If the validator tries to double sign or fork the system, then the coins are slashed in the vault [97].

Proof-of-Activity

Proof-of-Activity was created as an alternative to the incentive structure for Bitcoin. It is a hybrid approach that encompases Proof-of-Work and Proof-of-Stake. Mining begins in a typical Proof-of-Work fashion with miners racing to solve a cryptographic puzzle. Depending on the implementation, blocks mined don't contain transactions, so the winning block will only have the header and the miner's reward address. At this point, the system switches to Proof-of-Stake. A random group of validators to sign the new block is chosen based on the information in the block's header. If some validators are not available to complete the block, then the next winning block is chosen, a new group of validators is chosen, and this is repeated until a block receives the correct amount of signatures. Fees are split between the miner and validators [97].

Proof-of-Burn

In Proof-of-Burn, coins are burnt by sending them into an address where they are irretrievable. This is done as an alternative to having to purchase expensive mining equipment to mine cryptocurrency. By committing their coins to an irretrievable location, a node earns the permanent privilege to mine on the system based on the random selection process. Based on its implementation, miners may burn native cryptocurrency or the currency of an alternative chain. The more coins a node burns, the more likely they are to be chosen to mine the next block. Over time, the node's stake in the system declines, so the burning of coins is recommended to increase one's odds of being selected in the lottery. This is similar to bitcoin's mining process to invest in more computing equipment to maintain hashing power. Critics claim that this mechanism wastes

resources needlessly and that mining power simply goes to those who are willing to burn more money [97].

Proof-of-Capacity

Proof-of-Capacity is done through the node's hard drive space. The more hard drive space present on a node, the better chance the node has to mine the next block and earning the block reward. The algorithm generates large data sets known as "plots" that are stored on the node's hard drive space. The more plots present on the drive, the better chance that node will have of finding the next block in the chain. However, there is still the "nothing-at-stake" problem as with Proof-of-Stake [97].

Proof-of-Elapsed-Time

Intel has designed the Proof-of-Elapsed-Time consensus mechanism which performs similarly to Proof-of-Work but consumes less electricity. The algorithm uses a trusted execution environment to ensure that blocks get produced in a random lottery fashion but without the required work. This is guaranteed on the wait time provided through the environment [97].

Proof-of-Authority

Proof-of-Authority is a modified implementation of Proof-of-Stake. A validator's identity performs the role of stake rather than using monetary value [102].

Limitations

Another factor to think about when considering blockchain limitations is the size of the blockchain. As each new block is added to the blockchain, each node must reflect the addition. This becomes problematic when the blockchain itself reaches unmanageable sizes. Each block on the ethereum blockchain has an average size of 1.225 KB. Over the course of a month, the ethereum public blockchain grows approximately 187 MB. This makes it difficult for an average node run by an independent entity to continuously sync and maintain the blockchain [50].

Network Size

Blockchains respond to attacks and grow stronger. However, this requires a large network of nodes. Smaller Blockchain networks are not able to experience the full benefits of the Blockchain [104].

Transaction Costs and Network Speed

Bitcoin currently has transaction costs of about $0.20 that can only store 80 bytes of data at a rate of 7 transactions per second [104]. This impacts performance on the network if more than 7 transactions are pending approval in one second. The network will become slower, and it will take much longer to process transactions as a result.

Blockchain networks require nodes to run but newer networks don't have the number of nodes required to facilitate widespread usage. This results in higher costs since nodes are seeking higher rewards for completing transactions. This also slows transactions as nodes prioritize transactions with higher rewards, resulting in a backlog of transactions [105].

Human Error

If a blockchain is being used as a database, then the data being entered into the database must be 100% accurate the first time it is entered. If someone enters information that has spelling errors or is inaccurate, then the data is not inherently trustworthy [104].

An Unavoidable Security Flaw

If more than half of the nodes on the network store invalid and untrue data, then the data will be deemed valid. This is known as the 51% attack [104].

Inefficient Spending of Computational Power

Every node on the Blockchain network runs the blockchain in order to maintain consensus. This provides high levels of fault tolerance, ensures zero downtime, and makes data forever unchangeable and censorship-resistant. This is wasteful, however, as each node repeats a task to reach a consensus. This burns electricity and time and makes computations far slower and more expensive than on a single computer. Alternative consensus mechanisms have been developed to provide other options to Proof-of-Work [105].

Block Sizes

Each block added to the blockchain increases the overall size of the database. Each node has to maintain the blockchain to run on the network, and the computing requirements increase with each use. For large public implementations, this has one of two results:

1. Smaller ledgers - Not every node can carry a full copy of the blockchain. This impacts immutability and consensus.
2. More centralized - THere is a high barrier to entry to become a node. This encourages a larger amount of centralization in the network [105].

Hard and Soft Forks

Maky blockchains decentralize their decision making. Bitcoin allows nodes to "signal" support for improvements to the core software that runs their network. This allows the blockchain to avoid centralized decision processes but also presents challenges when communities are unsure of the best course for approving a transaction.

When nodes change their software, there is a "fork" in the chain. Nodes operating the new software won't accept the same transactions as nodes on the old software. This creates a new blockchain with the same history as the one it is built on. Forks create uncertainty as they have the potential to fragment the power of the blockchain network [105].

Immutable Smart Contracts

Once a smart contract is implemented on the blockchain, it becomes immutable and cannot be changed. If there are any security flaws in the code, this can be exploited by hackers and these flaws are permanent. Hackers can exploit code flaws to send smart contract contents to their own accounts. Since the blockchain is immutable, large amounts of value may be lost forever [105].

Sharding

All blockchains face another scalability issue as they can not process more transactions than a single node can. Ethereum's blockchain can only process 7 - 15 transactions a second. As blockchains are adopted and become more mainstream, this number of verified transactions per

second becomes cumbersome and the average time for a transaction to be verified increases. One way to combat this inadequacy is a method called sharding. A shard is a partition of an ethereum's state. An ethereum's state is a set of information that represents the current state of the blockchain. Currently, on most blockchains, each node must stores all states and processes all transactions of the blockchain In a sharded system, each node would only process a certain portion of the states and only a select amount of transactions. Granted there are enough nodes on the system, a blockchain could be verified by verifying a shard rather than a full block.This would decrease the amount of time needed to update the blockchain because it would cost less computing power, thus it can be done quicker. It also retains its innate security [51].

3.7: Alternatives to Ethereum

Monax

Monax is an open platform for developers to build, ship, and run blockchain applications for business ecosystems. Monax also sells legally compliant smart contract-based SDKs to speed up the development process to be able to market sophisticated ecosystem applications [89].

HydraChain

Hydrachain is the most useful extension of the Ethereum platform that supports the creation of scalable Blockchain based applications. The platform is 100% compatible on an API and contract level. HydraChain is open source with an infrastructure for developing smart contracts in Python. It is fully compatible with the Ethereum protocol and supports sub-second block times [84].

MultiChain

Multichain helps organizations to build and deploy blockchain applications. It was created by Coin Sciences and its goal is to free banks from the rigid options offered by competitors by allowing them to build their own blockchain [90].

OpenChain

Openchain is an open source distributed ledger technology that allows for issuing and managing digital assets in a highly secure and scalable way. Openchain follows a divided consensus system where every new Openchain has one authority for authenticating transactions. It leads to a

client-server architecture that they believe is more effective and reliable than peer-to-peer. Every transaction is digitally signed and the administrator of the instance defines the rules on the ledger [84].

BigChainDB

BigChainDB is an open source system that starts with a big data distributed database and adds characteristics of the Blockchain such as decentralized control and the transfer of digital assets. Developers and enterprises can use BigChainDB to deploy Blockchain proof-of-concepts, platforms and applications. BigChainDB allows for permissions settings at the transaction level as well as a federal consensus model [84].

Lisk

Lisk is a Blockchain platform and project based out of Zug, Switzerland. It allows any developer to create, manage, and distribute decentralized applications. Custom tokens can be created with Lisk and developers can deploy their applications on their own sidechains that are linked to the main Lisk network. Lisk developers chose to create this platform because they saw there was a large lack of developer adoption in terms of Blockchain. Their goal was to make the technology more accessible while bringing complexities to a minimum [83].

Stratis

Stratis allows developers to use C# and Microsoft .NET to create decentralized applications. This allows enterprises and individual developers to easily develop decentralized applications. Smart Contracts will run on sidechains to allow the main Stratis chain to experience less load and be more secure [83].

NEO

NEO is previously known as Antshares and is a Chinese Blockchain that was developed for creating smart contracts, decentralized applications, and ICOs. The transactions are fueled by NeoGas. Both NEO and ETH are open source, have massive communities, represent full transparency, and are Turing complete. With NEO, users can register, exchange, and circulate a variety of assets. Digital and physical assets can be linked together through digital identity and are

therefore validated and protected by national law. Smart contracts on NEO use C#, Java, and other popular languages [83].

Cardano

Cardano is a project with the main goal of changing how crypto assets are created and developed. Cardano applications can be used by any organization, government body, or individual around the globe. It uses the Proof-of-Stake protocol known as Ouroboros that is backed by a rigorous cryptographic model, is flexible, and grants the ability to compose protocols to enhance functional flexibility. Ouroboros allows for sidechains, subscribable checkpoints, better light client data structuring, delegation, random number generation, and synchronization assumptions. Decentralized applications using Cardano will operate fully on a Blockchain and the platform runs on the Haskel programming language. Cardano also has its own crypto wallet known as Daedalus [83].

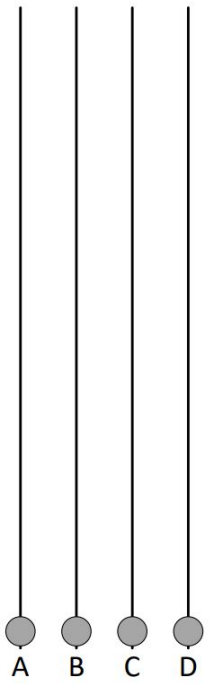
QTUM

QTUM is an open source hybrid Blockchain project that combines a bitcoin core fork with an Account Abstraction layer. It allows for multiple virtual machines and can fully support the Ethereum Virtual Machine as well as Proof-of-Stake consensus models. QTUM is designed specifically for industry, smart contract, and application building use cases. It positions itself to be a public Blockchain for business environments, specifically the mobile telecommunications industry, finance, logistics, manufacturing, and counterfeit protection [83].

Chapter 4: Hashgraph

4.1: What is Hashgraph?

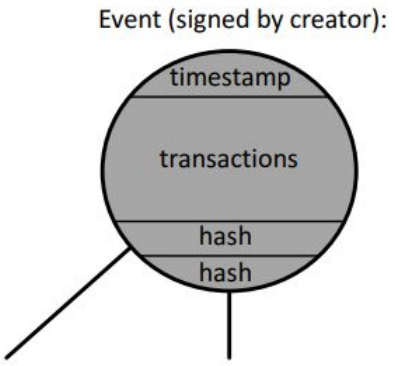
Hashgraph is a data structure and consensus algorithm that is fast, secure, and fair [25]. It grows upward over time and every participant has a copy of the hashgraph in its memory [26]. Each member (full node) on the Hashgraph starts by creating an event which is a small data structure that is stored in memory [26]. The goal of the Hashgraph consensus algorithm is for all members of the community to come to an agreement on the order of the events and the order of the transactions in the events and to agree on a timestamp for each event and transaction [26]. Using this system, it should be relatively difficult for attackers to prevent a consensus on the Hashgraph, to force different members to come to a different consensus, or to unfairly influence the order and timestamps that are agreed upon amongst the members of the Hashgraph [26]. To the right is a preliminary Hashgraph structure. This is similar to the genesis block of the Blockchain.



Core Concepts

Gossip about Gossip

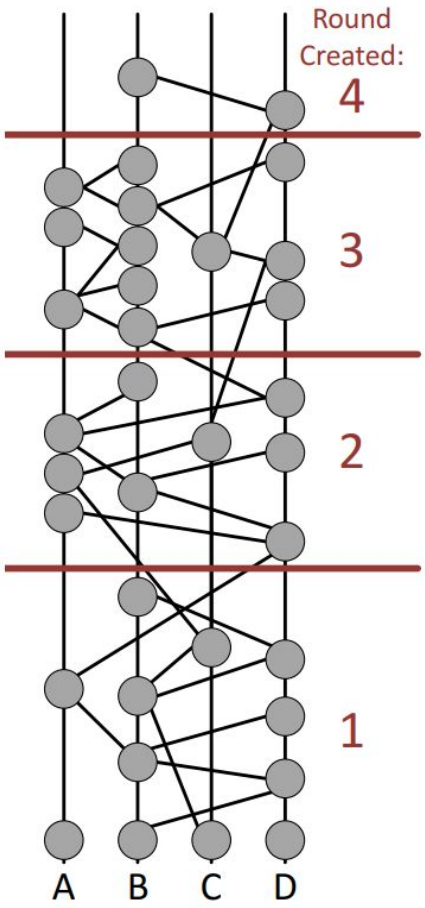
The members in the Hashgraph community run on a gossip protocol which basically means that members repeatedly call others at random to sync with them [26]. Members can avoid sending events to other members that already know about them by exchanging how many events they know about that were created by each member and exchanging events that are missing between them [26]. An event is a data structure that contains the two hashes of the two events below it (its self-parent and its other-parent) [26].



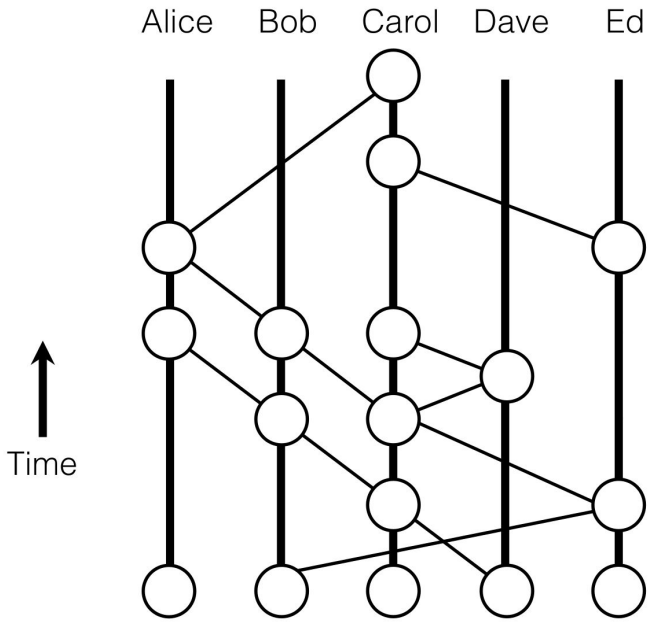
The graph formed by these gossips and transactions continues forever and therefore grows a directed acyclic graph upwards forever [26]. The graph is connected by cryptographic hashes thus leading to its name, Hashgraph. Every event contains the hashes of the events below it and is digitally signed by its creator [26]. The entire graph of hashes is cryptographically secure so it can always grow, but the older parts are immutable, as strong as the cryptographic hash, and a signature system is used [26].

Round Created

A child never has a round created before one of its parents [26]. Therefore, a round can only stay the same or increase. The round created for an event is R or R+1 where R is the maximum of the round created of its parents. It is only R+1 if and only if it can strongly see a supermajority of round R witnesses [26]. A witness is the first event that is created in each round [26].



It is possible for a member to cheat by forking which consists of creating two events with the same self-parent [26]. As a result, there might be two witnesses in the same round created by the same member [26]. A famous witness is determined by considering the witnesses in the next round. If the witness is seen by many of the nodes in the next round then it is considered to be famous [26]. An election occurs when each of those witnesses vote to determine if the witness is famous [26]. There are separate elections for every witness. To strongly see a witness, there must be enough different paths to it so that the paths go through a supermajority of the population [26]. A supermajority is any number that is more than ⅔ of the population of members [26].



Distributed databases are often required to be replicated state machines with Byzantine fault tolerance. For swirlds hashgraph, Byzantine is used in a strong sense: up to 1/3rd of members can be an attacker without compromising the performance of the hashgraph.

Byzantine Fault Tolerance

The hashgraph consensus algorithm is entirely asynchronous, nondeterministic, and achieves Byzantine agreement with probability one [27].

Hedera Hashgraph

On March 13th, 2018, Hedera Hashgraph was announced to the public. It provides a new form of distributed consensus that allows people to securely collaborate and transact online without a trusted intermediary. Hedera is fast, secure, fair, and doesn't require proof-of-work like Blockchain technology. The Hedera Hashgraph Council will provide governance for a decentralized public ledger built on Hashgraph's consensus algorithm. Governance is provided by a council of up to 39 known and reputable global organizations [27].

Governance

Hedera Hashgraph is comprised of Council Governance and Consensus. Council Governance is used for the management of the business of the council. Consensus is used to determine the consensus order of transactions.

Council Governance Model

Hedera will be governed by up to 39 leading organizations in their respective fields. Membership criteria are designed to reflect a range of industries and to have highly respected brands and trusted marketing positions. The terms of the governance ensure that no single member will have control and no small group of members will have too much influence over the body as a whole [27].

Consensus Model

Each node casts one vote for each coin of the hashcap cryptocurrency that they own. New nodes join the network and will be compensated for their services in maintaining the hashgraph structure [27].

4.2: Performance and Maintenance

The Hashgraph consensus mechanism is similar to the blockchain in which the "chain" is always branching, but without pruning, no blocks are ever stale, and each miner can mine many blocks per second without proof-of-work and with 100% efficiency [27].

Hashgraph also has improved security since it is Asynchronous Byzantine Fault Tolerant. This means that no member can prevent the community from reaching an agreement and they also cannot change the consensus once it has been reached [28]. Also, no transactions ever become stale since every container is used and none are discarded [28]. Hashgraph is inexpensive since it avoids the Proof-of-Work mechanism. Every member can create transactions and containers whenever they want [28].

The hashgraph distributed consensus algorithm provides near-perfect efficiency in bandwidth usage and can therefore process hundreds of thousands of transactions per second in a single

shard. A single shard is defined as a fully-connected, P2P mesh of nodes in a network. Consensus latency is measured in seconds [27].

Fair [29]

- No individual can manipulate the order of transactions.
- No individual can stop or delay a transaction from entering the system.

Provable [29]

- Once an event occurs, within a couple of minutes every member in the community will know where it should be placed in history.
- Every member will know that every other member is aware of the event.
- Members don't need to remember old transactions or blocks which shrinks the amount of storage needed.

Byzantine [29]

- No single member or group of members can prevent the community from reaching a consensus.
- No member or group of members can change the consensus once it has been reached.
- Each member will eventually reach a point where they know with 100% certainty that a consensus has been reached.

ACID Compliant [29]

- Atomicity, Consistency, Isolation, Durability
- Applies to the hashgraph when it is used as a distributed database: a community of members uses it to reach a consensus. Once a consensus is reached, each member feeds the transactions to their local copy of the database.
- The community as a whole can be said to have a single, distributed database with the properties of ACID if the local database has all of those standard properties.

Inexpensive [29]

- Avoids proof-of-work mechanism which can impact performance

- Does not need to waste computations to slow itself down for blocks to be properly mined.

Timestamped [29]

- Every transaction is assigned a consensus time which is the median of the times that each member first received it.
- If a majority of the participating members are honest and have reliable computer clock times, then the timestamp itself will be honest and reliable.
- Consensus timestamping is useful for smart contracts since there will be a consensus on whether an event happened by a deadline.
- Timestamps are resistant to manipulation by an attacker.

Non-permissioned [29]

- A permissioned system is one where only trusted members can be part of the community and participate in transactions.
- An open system is not permissioned and allows anyone to participate.
- Hashgraph can be designed to work in various ways such as proof-of-stake.

4.3: Capabilities and Limitations

Hashgraph eliminates the need for massive computation and unsustainable energy consumption that is present in Blockchain systems [28]. Hashgraph is only limited by bandwidth at 250,000+ transactions per second pre-sharding while Bitcoin is limited to 7 transactions per second [28]. Hashgraph is more fair since no individual can manipulate the order of the transactions.

4.4: Hashgraph and Security

Hashgraph consensus does not use a leader and is therefore resilient to the Denial of Service (DoS) attacks on small subsets of members [27]. Hashgraph achieves the “gold standard for security” by using asynchronous Byzantine Fault Tolerance (aBFT). Other platforms tend to be vulnerable to Distributed Denial of Service attacks (DDoS). Hashgraph is resilient to DDoS and

achieves the theoretical limits of security defined by aBFT. Hashgraph ensures both Fair Access and Fair Ordering.

DoS Resistant [29]

- Distributed structure resists such attacks.
- An attacker might flood one member/miner with packets to temporarily disconnect them from the internet but the community as a whole will still operate as usual.
- An attack on the system as a whole would require flooding a large fraction of the population with packets which is much more difficult.
- Hashgraph avoids the problem of an attacker freezing the entire system by attacking one leader at a time.

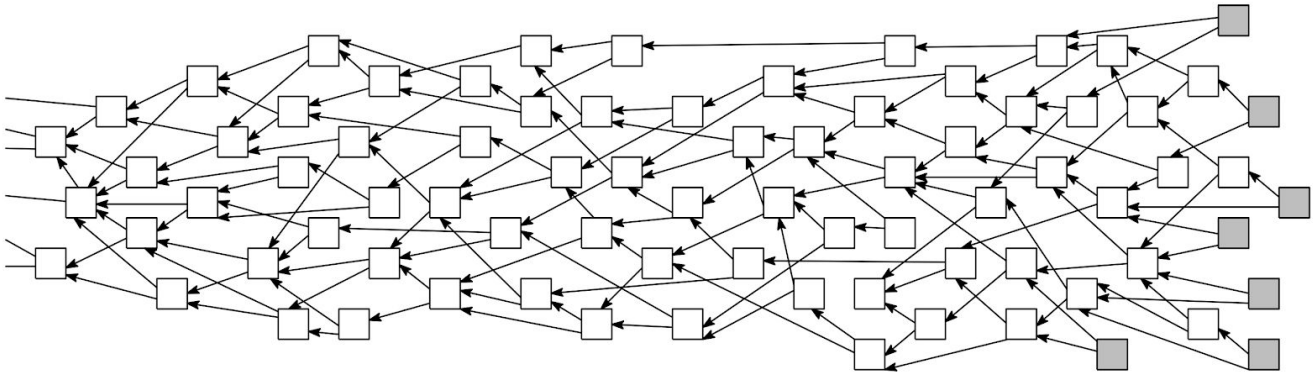
Sybil Attacks

- Consensus algorithm provides a mathematical guarantee that these attacks cannot succeed as long as less than ⅓ of the population is dishonest.

Chapter 5: Tangle

5.1: What is Tangle?

The Tangle is a Directed Acyclic Graph (DAG) which can be used for storing transactions as well as distributed consensus. The Tangle succeeds the blockchain by nature as it offers features required to establish a machine-to-machine micropayment system. The Tangle was developed for IOTA, a cryptocurrency for the Internet-of-Things industry [30].



Source: [30]

A Tangle is made of sites and nodes. Sites are transactions that are represented on the graph and nodes are the issuers of the transactions. Sites can contain one or more transactions that relate together. Tangle eliminates the need for miners as the responsibility of approving new transactions belongs to all nodes and brings equality back to all users. The first transaction in the Tangle is known as the genesis transaction. It is indirectly and directly approved by all other transactions and sends tokens from an address that holds all tokens to other founder addresses [94]. In the early stage of a Tangle, there is no rule for choosing transactions to approve. The nodes are assumed to follow some reference rules since they are usually local devices in the same geographic region [94].

Relationships between transactions can be formed directly or indirectly. A direct transaction consists of a transaction A being directly approved by transaction B. An indirect transaction consists of some transactions being placed between transactions A and B that allow A and B to be indirectly related, such as adding a transaction x and y to form A → X → Y → B [94].

Core Concepts

Transactions [30]

In order to issue a transaction, a node must accomplish the following:

1. The node chooses two other transactions to approve according to an algorithm. These transaction may or may not coincide.
2. The node determines if those transaction are conflicting or not, and does not approve the conflicting transaction.
3. The node must then solve a cryptographic puzzle, similar to blockchains proof of work. It does this by solving a nonce such that its hash is concatenated with the data of the approved transactions.

It is important to note the asynchronicity of IOTA's Tangle. It is also important to note that the tangle may in fact contain conflicting transactions, however these transactions are left on the tangles 'edge' and as more transactions are verified conflicting transactions become 'orphaned', and are prevented from being indirectly approved from incoming transactions. The tangle does this using its *tip selected algorithm*[30]. If a node doesn't approve enough transactions, it will be given status of 'lazy' and surrounding nodes will drop it. In this way, nodes are motivated to participate in approving transactions else they could never make a transaction themself [30].

Weights, Tips, and Scores

The weight of a transaction is proportional to the effort put in by its user. A cumulative weight is the sum of the transaction's own weight plus the sum of the weights of the following transactions that indirectly or directly approve it.

A tip is a newly issued transaction that hasn't received approval. An algorithm called tip selection algorithm is used to solve conflicts between tips by running many times to check which transaction is more likely to be approved by a selectec tip [94].

The score is the transaction's own weight plus the sum of the weights of all previous transactions approved by this current transaction. The height of the Tangle is the length of the longest oriented path to the genesis transaction. The depth of the Tangle is the length of the longest reverse-oriented path to specific tips [94].

5.2: Performance and Maintenance

The Tangle is asynchronous and can therefore tolerate any conflicting transactions. It believes that any incorrect transaction would automatically be orphaned or erased as the Tangle progresses in size [94]. In the event that a node shows "laziness" toward propagation of transactions, it will be dropped by its neighbor. The incentive keeps all nodes working even if they don't issue transactions as frequently [94].

Cutsets

The cutset is a group of tips that are alive between a specific time and the average time of issuing a transaction. It must have the following definition [94]:

- Any path from a new transaction to the genesis transaction needs to go through the cutset.
- The smaller the size, the higher the chance that a new transaction will be approved. A new transaction has fewer competitors.
- It is used as a checkpoint for DAG pruning and other actions taken to control the development and progression of the Tangle.

Choosing Transactions

Transactions can be chosen randomly or random among the top section. Choosing transactions at random is not advisable as it does not encourage approving tips. Choosing transactions at random among the top section, which is the section near the tips, is suggested as tips have a higher probability of being selected and approved [94].

Low Load and High Load

The load of the Tangle determines its security, efficiency, and feasibility. A low load is considered to be a small number of tips, usually 1 or 2, and results in a weak inflow of transactions. This occurs when there is low network latency and great computational power. A high load is a larger number of tips that depend on the strategy of choosing the transaction. It results in a strong inflow of transactions and occurs when there is a high network latency and low computational power. This forces new transactions to turn to approve the same tips. The main disadvantage of a high load is that some transactions might need to wait a long period of time before gaining approval [94].

There are a few methods of solving this problem. As suggested in whitepaper v1.2, the owner of the transaction can reattach/broadcast the transaction while knowing that the tokens won't be sent until the transaction gains approval. Any duplicate copies will be discarded later [94]. Another solution suggested by whitepaper v0.6 states that the owner of the transaction can issue an empty transaction referencing the missed transaction to help it obtain approval. Another average solution involves picking random tips and approving the top ones to increase the chance of the missed transaction being approved [94].

Tangle's Growth Speed

The speed of growth for a transaction's cumulative weight can be examined in the context of high load and low load. With low load, the growth will be at a constant speed. However, with high load, the growth will be increasing in speed during its adaptation period, and return to a constant speed after this period [94].

5.3: Capabilities and Limitations

The Coordinator

IOTA relies on a coordinator to make it possible for the network to grow and for protection against certain attacks. The coordinator checkpoints valid transactions that are then validated by the network as a whole. The coordinator is run by the IOTA foundation in a multi-signature and

cannot go rogue as it is being checked and validated by the network. It also issues period milestones that reference valid transactions [31].

5.4: Tangle and Security

IOTA's Tangle has implemented many attack prevention algorithms to defend from a variety of threats. The Tangle is not Byzantine Fault Tolerant.

Potential Attacks

Large Weight Attack

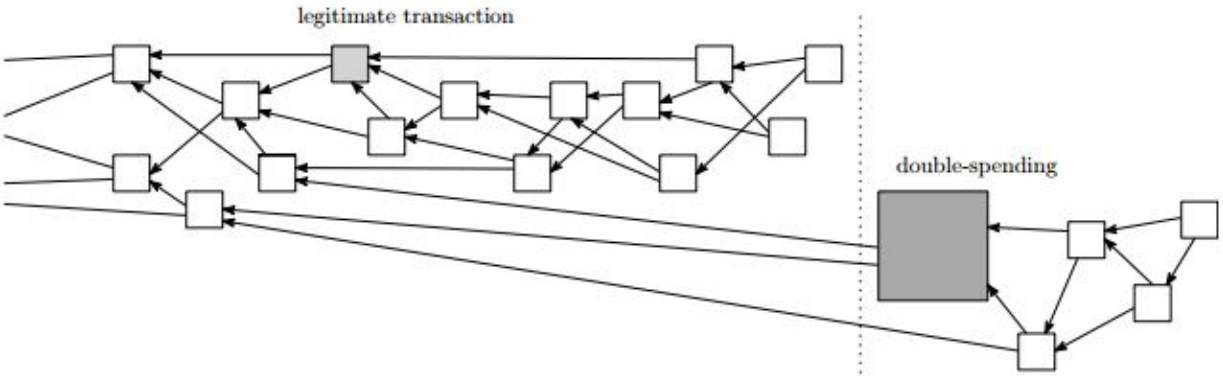
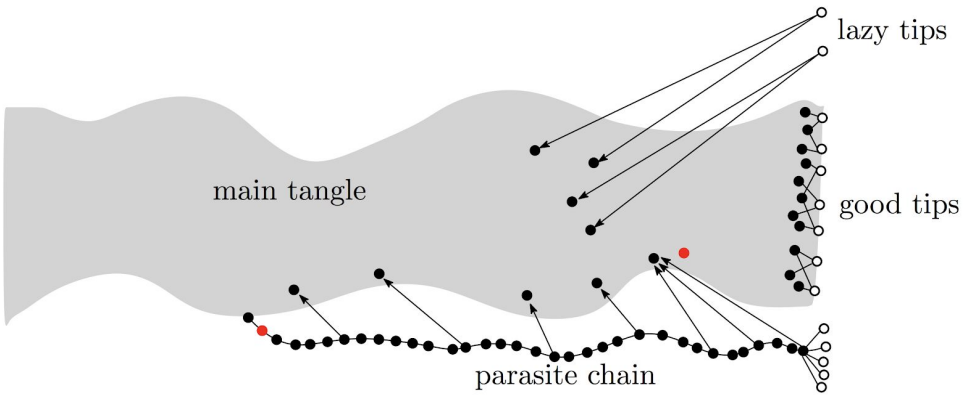


Figure 5: The "large weight" attack

Source: [30]

In a large weight attack, the attacker tries to give a large weight to the double-spending transaction so it would outweigh the legitimate subtangle in order for the attacker to be able to double-spend. This would be detrimental to a network in the case where the allowed weight of a transaction is unbounded. To solve this attack, one may decide to limit the weight of a transaction or set it to a constant value [30].

Parasite Chain Attack



Source: [30]

The Parasitic Chain attack consists of an attacker secretly building a subtangle to occasionally reference the main tangle to gain a higher score. The score of honest tips is about the sum of all of its weights on the main tangle. The score of the attacker's tips would also contain the sum of all of its weights in the parasite chain. If the attacker uses a computer that is sufficiently strong, then they may be able to give more height to the parasite tips since network latency isn't an issue for a single subtangle. The attacker can also artificially increase their tip count by broadcasting new transactions that approve transactions issued earlier on the parasite chain. This helps the attacker to gain an advantage in the case where honest nodes use a selection strategy that involves a simple choice between available tips [30]. To defend against this attack, a MCMC algorithm is used to select two tips to reference by using the fact that the main Tangle is intended to have more hashing power than the attacker.

Splitting Attack

The splitting attack was proposed as a potential attack against the Tangle that is using the MCMC algorithm. It suggests that in a high load regime, an attack could potentially split the main Tangle while retaining a balance between the two halves. The attacker does this by placing two conflicting transactions at the point of the split to prevent honest nodes from joining the network. Then, the attacker hopes that the branch will be roughly split in half so half of the network contributes to each branch similarly. If the attacker succeeds, the attacker could issue the same spending transactions on both chains. To defend against this, the tangle employs a sharp threshold rule that makes it extremely difficult for a split Tangle to retain balance between both of them [30].

Quantum Computing Resistance

A sufficiently large quantum computer can be used to efficiently solve problems that rely on trial and error for solutions. A large weight attack would be much more efficient on a quantum computer since it is 17 billion times more efficient at mining the Bitcoin blockchain than a classical computer. However, capping the weight of the Tangle would prevent a quantum computer attack since the number of nonces that one needs to check to find a suitable hash isn't unreasonably large. The algorithm used in IOTA is structured in a way that the time to find a nonce isn't much larger than the time needed for other transaction-dependent tasks. Capping the weights makes the Tangle much more resistant to quantum computing and gives the Tangle much more protection against an attacker when compared to the Blockchain [30].

Chapter 6: Hyperledger

6.1: What is Hyperledger?

Hyperledger is an open source community that is dedicated to helping the advancement of technology be being an “umbrella” for developer communities to build open source blockchain and related technologies [5]. The vision of Hyperledger is to provide robust and efficient standards for the blockchain ledger technology to facilitate mainstream commercial adoption [5].

Various Hyperledger projects include [5]:

1. Hyperledger Fabric: IBM contributes to this open source project which has become their biggest/highest profile project.
2. Sawtooth Lake: Intel has contributed to this project which is a blockchain technology that introduces the Proof of Elapsed Time consensus.
3. Iroha: A C++ blockchain platform.
4. Cello: Provides rapid blockchain development to cloud platforms.

Hyperledger Fabric

Hyperledger Fabric is a blockchain framework implementation that has a modular architecture with pluggable/interchangeable services that use container technology.

Various benefits of Hyperledger Fabric include [5]:

- Supports a wide variety of industry use cases with different requirements
- Complies with today’s statutes and regulations
- Supports verified identities and private and confidential transactions
- Supports permissioned and shared ledgers
- Supports performance, scaling, auditability, identity, security, and privacy
- Reduces costly computations involved with proof-of-work consensus

Hyperledger Fabric fulfills all 4 key elements of a blockchain for business [5]:

- Permissioned Networks: Collectively defined membership and access rights for nodes on a business network.
- Confidential Transactions: Gives a business the flexibility and security to make transactions visible to selective parties with the proper encryption keys.
- Doesn't Rely on Cryptocurrencies: Doesn't require mining and expensive computations to validate transactions.
- Programmable: Leverages the embedded logic in smart contracts to automate business processes on a business network.

Fabric's understanding of a consensus mechanism is broad and encompasses the entire transaction process from start to finish. Nodes take on different roles and tasks during this consensus process. This is different to Ethereum's mechanism where roles and tasks of nodes participating in the consensus are identical [20].

Nodes are differentiated based on their role: client, peer, or orderer [20]. A client acts on behalf of the end-user by creating and invoking transactions and also communicate with peers and orderers [20]. Peers maintain the ledger and receive ordered update messages from orderers that wish to commit new transactions to the ledger [20]. Endorsers are a special kind of peer that is responsible for endorsing a transaction by checking if they fulfill the necessary and sufficient conditions [20]. Orderers provide a communication channel to clients and peers where messages containing transactions can be broadcasted [20]. The channels ensure that all connected peers are delivered the same messages in the same logical order [20].

There are problems that arise with Fabric's mechanism of consensus as well. For instance, faults in the delivery of messages may occur when many mutually untrusting orderers are put in place [20]. A consensus algorithm has to be used in order to reach a consensus despite faults [20]. The algorithm put in place by Fabric is "pluggable", meaning that various algorithms can be used depending on specific application requirements [20]. Fabric's channels partition the flow of messages so clients only see the messages and transactions of the channels they are connected to while remaining unaware of other channels [20]. This allows for restricting access to

transactions to involved parties only with the consequence that consensus only has to be reached at the transaction level and not at the ledger level as with Ethereum [20].

Role of Nodes in Transaction Flow [20]

- Client sends transaction to connected endorsers to initiate ledger update.
- Endorsers must agree on the proposed transaction. A consensus has to be reached on the ledger update proposal.
- Client collects approval of all endorsers.
- Approved transaction is sent to connected orders to reach consensus.
- Transaction is forwarded to peers holding the ledger for committing the transaction.

Hence, Fabric provides fine-grained control over consensus mechanisms and restricts access to transactions which provides improved performance scalability and privacy.

Hyperledger is an organization focused on the development and research of blockchains while providing a platform for projects to collaborate. Hyperledger plans on proving a set of guidelines that architecture of blockchains can follow. Hyperledger consists of multiple projects who all follow a common design philosophy. The architectural design that projects follow is based off of the following components [56]:

- Consensus Layer
- Smart Contract Layer
- Communication Layer
- Data Store Abstraction
- Crypto Abstraction
- Identity Services
- Policy Services
- APIs
- Interoperation

Their main focuses are on a modular architectures with high security that is natively not based on cryptocurrency.

Hyperledger provides some core functionality that consensus must follow. It must confirm the correctness of the transactions proposed by on endorsement and policies. All nodes must agree on order and correctness to execute the transaction. Lastly, it must interface with the smart-contract layer to verify the correctness and order of transactions. Hyperledger states that consensus processing is based on an environment of partial trust, much different than Bitcoin or ether who rely on anonymous nodes. This is because Hyperledger is based on a business sense where nodes will most likely be trusted as they will be within a closed secure system. Consensus can be approached in multiple ways, but Hyperledger focuses on three: permissioned lottery-based, permissioned voting-based, and stand proof of work. However, based on the modularity of Hyperledger framework aims to provide any other methods of consensus can possibly be implemented. Hyperledger essentiality gives a platform that provides interface between modular aspects so they may be modified without affecting how other pieces work. Hyperledger fabric is just one architectures that follows these guidelines, there is also Hyperledger Sawtooth, Burrow, Iroha and Indy. All of which have unique features that allow them to function different from one another but all follow Hyperledger's standards [56].

Hyperledger Fabric [68]

Hyperledger Fabric uses a consensus algorithm known as Kafka and approaches consensus as permissioned-voting based. The consensus is broken into 3 phases, Endorsement, ordering and a validation phase. The endorsement phase is controlled by a policy such as having m out of n signatures participate in the transaction. Once the transaction is endorsed, the ordering phase then accepts it and commits it to the ledger. The validation phase then verifies the correctness of the transactions. Based on hyperledgers modularity, these three phases are pluggable, meaning that different ways of endorsement, ordering, and validation can be implemented based on the users needs. For example, the ordering service offers many API plugins that create a variety of ways the services works.

The hyperledger fabric model consisted of some key features. They are assets, chaincode, ledger features, privacy, security, and consensus. Assets are essentially what is part of the transaction, from a tangible item being sold or a contract. These assets are modified by the next feature, being the chaincode. The chaincode is the rules enforced for modifying assets. This is essentially what

needs to be done in order for a transaction to take place as well as the execution and writing to all peers. The ledger features are a main part of what sets Hyperledger Fabric apart from others. The blockchain consist of many channels, each of these channels has its own ledger as well as special rules in members. For example, a peer can have access to a channel involving every other peer and a channel involved only 2 peers. This brings up the privacy feature within Hyperledger fabric. Based on the channel system a channel can be made private between chosen peers, or a channel can be open to everyone. This can further allow obscuring data within a transaction to only the peers allowed, but also displaying to all that a transaction took place. Hyperledger fabrics security is also a promptly based on a trusted level with business use in mind. Each member has a known identity which essential for conducting business transactions where the user would need to know who they having a transaction with.

Consensus in hyperledger fabric emcompasses the lifecyle of a transaction. Through the process of a transaction, there are multiple checks that take place to ensure that the transaction is correct. The extends between endorsements and that the correct chaincode policies were met. There final check involves a versioning check to make sure the ledger is correct before appending the new transaction. Consensus within fabric is based around the whole transaction from start to finish.

Hyperledger Sawtooth [57, 58]

Hyperledger Sawtooth once again aims to create a platform where customized applications are built off of a system core. This separation of application from core allows developers to create customized applications, whether they be smart contract VMs much like ethereum or business logic like Hyperledger fabric, to run on this core. Sawtooth even allows for multiple types of applications to exist in the same blockchain network. Sawtooth aims to be used in private networks, and the blockchain as a whole stores who are participants, their roles, and identify which is available to other members within the network to view.

One of the distinct features of sawtooth is the parallel scheduler which can split transactions into parallel channels to be executed while maintaining the changes to the blockchain. This gives

sawtooth a potential to have greater performance compared to traditional transaction processing.

Consensus in sawtooth allows the probability of multiple types as well as allowing different types on the same blockchain. Sawtooth currently supports a few consensus types, Proof of Elapsed Time(PoET), PoET simulator, and Dev mode. Proof of Elapsed Time is designed to support large networks, while PoET simulator is based on hardware. Dev mode is primarily used for testing and is a simple random-leader algorithm. The default consensus is the lottery based PoET, which is lottery leader is selected based on the guaranteed wait time. This way of consensus is based on fairness, investment and verification. Fairness should distribute a leader across all participants, investment should correlate the cost controlling the leaders election to the value of it, and verification that all participants verify the leader is legitimate. Sawtooth builds off of Intel's Software Guard Extensions to ensure that the leader is selected safely, randomly, and is cost efficient. This in return mainly means the necessary computing power required in a proof of work blockchain is not needed. Leaders are chosen based on a request for a random time to wait, whichever has the shortest time to wait is elected leader and so on.

Sawtooth also provides transactions families, there are data models that reflect the requirements of the ledger. These families are highly customizable and can be made to a users needs but Sawtooth provides some example families. These families declare what is essentially stored within the state. This correlates to consensus, for example PoET requires a target wait time setting which can be implemented into the family setting and adjusted to the users desire.

Hyperledger Burrow [67]

Hyperledger burrow is designed to execute permissioned Ethereum smart-contracts. Burrow consists of a consensus engine where transactions are executed in order based on a byzantine fault-tolerant Tendermint protocol. Transactions are validated and applied based on the order the consensus engine finalized them.

Hyperledger Iroha [65, 66]

Hyperledger Iroha is aimed to provide reusable components to compliment other projects such as Hyperledger fabric and Hyperledger Sawtooth. The project plans to implement these components in C++ and provide a long term and robust library of features developers can take advantage of. Irohas main goals are to provide a C++ environment for hyperledger, provide mobile application support, and provide an environment for new components (Such as other consensus algorithms) to potentially be incorporated into hyperledger. Iroha has a major focus on C++ design and mobile applications, two environments lacking in other hyper ledger projects.

Irohas architecture provides a number of useful features. The system will work on a peer to peer basis, currently all peers are validating peers but there are plans to implement client peers, validating peers, and normal peers together. Iroha will feature a membership service where $2f+1$ signatures are needed to add or remove a node. There are future plans to implement membership groups and more features to how membership permissions are given. Iroha provides a default cytophrahy through SHA-3. Iroha also supports chaincode, implementing domains and assets, and running transactions with these elements. Iroha stores transactions in the form of a merkle tree where each transactions represents a leaf.

Consensus through Iroha is done through a Byzantine fault tolerant consensus algorithm called Sumeragi. Consensus through this method consist of validate peers of at least $2f+1$ peers to sign the transaction. First the client submits the transaction to a validating peer, who then signs it and broadcasts it to the $2f+1$ peers. The servers then verify the transaction from the broadcast then commit once the $2f+1$ signatures are met. Iroha also features a peer reputation system as a way of establishing trust. This is called the hijiri reputation system. The system revolves around a set of rounds that validating peers run that test data throughout, version, computation, and consistency.

Hyperledger Indy [64]

Hyperledger Indy is focused around a decentralized identity system. Indy features a system of trust where users can exchange verifiable claims with a strong focus on privacy. A major part of this privacy is Indys use of decentralized identifiers. These are the primary keys to the ledger which do not require a centralized registry service. These keys are verified through cryptography which creates a “web of trust”. Indy also does not write any personal data to the ledger, instead it

is all exchanged through peer-to-peer encrypted connections. Indy also has support for zero-knowledge proofs to avoid any unnecessary disclosure of identities.

6.2: Performance and Maintenance

Hyperledgers projects can offer a solution to one of blockchains biggest issues, scalability. With traditional blockchains every node has a copy of the entire chain. Because of this, when the chain becomes very large and there are many nodes, the size of data storage needed drastically becomes bigger. Each hyperledger architecture processes consensus and data storage differently, however mainly provide a way of storing a ledger that is much more efficient to many blockchains. As mentioned in Hyperledger fabric, the architecture revolves around channels. Essentially when a transaction is done, everyone in the system can see that a transaction was made between 2 entities while only disclosing some of the information about this transaction. (Hyperledger allows the developers to decide what information is shared to everyone). The 2 parties involved however are the ones who store their copy of the transaction and all of its parts. This major change in data storage means that what would be considered wasted data is separated from what everyone needs access to, in turn improving scalability [56].

Hyperledger architecture often include a membership service for maintaining nodes and usually a fail safe in case of an issue with a transaction. Hyperledger's membership services can range from single entities who oversee and validate new nodes, to consensus validation through all the nodes to add or remove a node. Because of hyper ledgers projects modularity, whatever the needs of the system can easily be met to maintaining the nodes [56].

6.3: Capabilities and Limitations

Hyperledger projects are capable of a wide range of functions that separate it from other blockchains. Their main feature being its design around modularity. Hyperledger's goal is to create a platforms that can be manipulated to the users needs. This extents from functionalities such as consensus, and validation to how entities are allowed onto the blockchain. Hyperledger's performance capabilities are highly based on how it is implemented and what it is planned to be used for. They are capable of giving even more options into the architecture by offering systems

like Iroha that provides C++ and mobile applications, or like barrow which implements an ethereum VM. There is a wide range of capabilities and limitations to hyperledger projects, because of its modularity many of the limits will be controlled by how they are implemented [56].

6.4: Hyperledger and Security

Security in hyperledger is largely designed around permisioned based systems. Because of the large focus around business needs, hyperledger plans to implement systems that require the entities to be permissioned identities. This is often done by some membership system that approves who is allowed on the blockchain. The idea is that the members using will the system will be conducting business together, often needing to know who they are doing business with and validating the authenticity that it is them. These membership services create a private closed blockchain where trust is inherit because each member was approved to be a member. Although membership is implemented, once again because of hyperledgers modularity, this can change to the users needs. We see in hyperledger Iroha that the default system of allowing membership is based on a number of nodes approving a member be added. Users can decide how private they wish their system to be, and what method they wish to be conducted when giving membership [56].

Hyperledger also offers many methods of consensus security. Hyperledger fabric offers consensus that can be based on only a few nodes. For example user A can trade user B and may only need user C, who is a bank, to approve that the transaction is legitimate. This way of consensus can also be changed depending on the need. We see in hyperledger sawtooth consensus is done through a lottery based system [56].

Hyperledger continuously demonstrates its modularity and choice when it comes to security. A developer of a system can not only choose their method of consensus to validation and membership, but also fine tune the methods chosen. We see this throughout all of hyperledgers projects, as they each offer different security methods, but the main focus on security is through a private closed blockchain [56].

Chapter 7: R3 Corda

7.1: What is Corda?

Corda is a technology that uses distributed ledgers as a form of financial services in order to create transactions can be processed seamlessly and kept secure. The vision of Corda is a “global logical ledger”, which will allow all agents to interact in their own agreements while being secure, reliable, private, and authoritative. Essentially, any agent within the network will be able to create secure transactions while deciding who is involved with this transaction. This means that unlike a blockchain, every node will not need to store every transaction on the network. Corda’s implementation is based off of many principles including [70]:

- Records are accepted as evidence and legally bound to the parties involved.
- Records finalized on the ledger cannot be changed, another transaction is needed to fix the error.
- The only parties who have access to transactions are those who need to know. Example: Two parties of a transaction and their banks.

Corda also has a large vision for the future use of this technology. Its goals include reduction of cost, risk, and maintenance as well as establishing new services. They also wish to gain adoption through many financial communities. Corda is designed for regulated financial institutions and has many key features that make it useful for this environment. It is designed for the recording of agreements between parties while adhering to all legalities and existing regulations. The network allows seamless workflow between groups as well as supporting consensus between individuals instead of a whole as well as a variety of consensus methods. This allows for transactions to be validated between only those who are involved and restricts access of the agreement to those verified to see it [70].

7.2: Performance and Maintenance

Corda's design separates it from the limitations that many blockchains have, specifically with scaling and TPS (Transactions per Second). Since Corda is designed in a way that only the members part of a transaction are required to have the data involving said transaction, a massive amount of wasted space that traditional blockchains use is eliminated. This is because instead of having every transaction saved to every member, only the members involved in the transaction save the information. This will, in turn, save space but also increase scalability with a large number of transactions taking place. Corda's second consensus service of uniqueness can cause bottlenecking. All transactions are determined uniqueness by a notary system, essentially specific systems that determine the uniqueness of all transactions. However, if all transactions are accommodated by a single system, a bottleneck may occur. By having multiple systems spread out to deal with transactions, the risk of a bottleneck can be lessened [70].

7.3: Capabilities and Limitations

Cordas main source of limitation is its notary system of uniqueness. As mentioned above, this uniqueness system is prone to bottlenecking. Essentially, Corda's speed is limited to how many transactions are handled by a single notary system. This bottleneck can be accommodated for by a having many systems handle the input of transactions. The more systems, the less likely of a bottle neck.

7.4: Corda and Security

Consensus in Corda is performed using two methods: transaction validity and transaction uniqueness. Transaction validity is when the parties involved agree to consensus. This means only these parties can see the transaction. At least two parties are needed in order to record a transaction to the ledger. Transaction uniqueness can be implemented in a few ways. Corda allows for this service to be "pluggable" meaning that different consensus methods may be used, such as a Byzantine Fault Tolerance algorithm or a simple single machine system. The uniqueness systems are not required to see the information within the transaction as well, they are only there to confirm that the state has not been consumed before [70].

Chapter 8: Comparison of Distributed Consensus Systems

Comparison of Blockchain, Hashgraph, Tangle, Hyperledger, and Corda

Characteristic	Blockchain	Hashgraph	Tangle	Hyperledger	Corda
Description of Platform	Decentralized Tamperproof Ledger	Directed Acyclic Graph (DAG)	Directed Acyclic Graph (DAG)	Blockchain Framework	Distributed Ledger for Financial Services
Governance	Satoshi Nakamoto	Swirlds	IOTA	Linux Foundation	R3
Mode of Operation	Private or Public	Private or Public	Public	Private	Private
Consensus Mechanism	Proof-of-Work, Proof-of-Stake, Proof-of-Authority	Gossip about Gossip, Virtual Voting	Proof-of-Work	Broad understanding of consensus that allows multiple approaches at the transaction level.	Specific understanding of consensus (i.e. notary nodes) at the transaction level.
Byzantine Type	Not Byzantine	Asynchronous Byzantine Fault Tolerant	Not Byzantine	Byzantine Fault Tolerant (Some Projects)	Not Byzantine
Smart Contracts	Solidity	Java and Solidity	Not Used	Go, Java	Kotlin, Java, Legal Prose
Currency	Varies by Platform (e.g. Ether for Ethereum)	None	None	None	None
Attack Resistance	Denial of Service (DoS), Consensus Attack, Shallow Block Attack	Denial of Service (DoS), Distributed Denial of Service (DDoS), and Sybil Attacks	Giant Weight Attack, Parasitic Chain Attack, Splitting Attack, and Quantum Computing	Denial of Service (DoS), Consensus Attack, Shallow Block Attack	Denial of Service (DoS), Consensus Attack, Shallow Block Attack

			Resistance		

Table 1.1: Comparison of Blockchain, Hashgraph, Tangle, Hyperledger, and Corda

Chapter 9: Frequently Asked Questions

9.1: Blockchain

Q: What is a blockchain?

A: A blockchain is a digitized, distributed, immutable ledger that allows for the secure storage of data. It helps to prevent against data loss and tampering through its consensus mechanisms, distributed structure, encryption, and hashing.

Q: What is the difference between centralization and decentralization?

A: Centralization involves storing all of the data in a central location such as a server. Decentralization is the process of storing data in multiple centralized locations in a client-server structure. Decentralization is much more suitable than centralization as it eliminates the risk of a single point of failure.

Q: What are the main benefits of using a blockchain on a network?

A: The main benefits of using a blockchain include decentralization, immutability, and transparency. The blockchain provides a high degree of security for digital assets on a transparent platform.

Q:What is the difference between Bitcoin and the blockchain?

A: Bitcoin is a cryptocurrency that upholds all of its transactions on a blockchain. Bitcoin is a use case of blockchain - blockchain is the technology that allows bitcoin users to exchange true and valid transfers of digital assets (bitcoin) without an intermediate party.

Q:What can a blockchain be used for?

A: Various use cases of the blockchain include record management, voting, Internet of Things, cybersecurity, information asset storage and transfer (e.g. cryptocurrency), and Peer to Peer sharing (e.g. media platform).

Q: What is a distributed consensus system?

A: Let's start with the preliminary two-node consensus system. There are two nodes that need to agree on a piece of data such as a transaction or a student record. Once the two nodes agree that this piece of data is valid, a consensus has been achieved on the validity of the data. A distributed consensus system follows the basic protocol of a two-node consensus system. However, it is distributed amongst various nodes on

the network instead of two single nodes as with a standard method of consensus. As a result, multiple nodes on the network need to reach an agreement to establish a consensus for a piece of data.

Q: What is the difference between Blockchain and Ethereum?
A: The blockchain is an inherent structure that stores data. Ethereum is a platform that utilizes the blockchain structure for the creation of decentralized and distributed applications.

Q: Does Blockchain have any performance and maintenance requirements?
A: A blockchains performance requirements depends on the specific use case that blockchain is being used for. Maintenance can be done on the blockchain in several ways, namely pruning and forking.

Q: Does Blockchain have any special capabilities or limitations?
A: The biggest problem with the blockchain is the larger it gets the slower and clunkier it becomes. There are several methods of working on fixing scalability issues such as sharding, pruning, lightning network, and linking with other consensus systems like tangle.

Q: What are some special use cases/applications for Blockchain?
A: Various use cases include voting, gun tracking, cryptocurrency, supply chain management, personal identification storage, peer to peer sharing, and file storage.

Q: Are there any hardware requirements for using the Blockchain?
A: It is all dependent on the application of the blockchain itself. It is recommended to have a minimum of 2GB RAM for each node.

Q: Is there a minimum number of nodes required to have a Blockchain?
A: A minimum of one node is required to store the blockchain. However, it is not advised to maintain one node on the network since that node will be able to mine all blocks and earn all of the gas on the network. The larger the network, the more secure it is as more nodes will participate in the consensus algorithm and it will be harder for an attacker to gain control of 51% of the network to perform attacks on the data and network.

Q: What is the difference between the Blockchain and an Excel spreadsheet?
A: An excel spreadsheet is a centralized and plaintext document. Using an excel spreadsheet poses various risks to tampering, data loss, a single point of failure, and lack of availability to intended users. These risks are mitigated through the use of a blockchain.

Q: What are the main advantages of using the Blockchain?

A: Main advantages of the Blockchain include:

- Distributed structure that eliminates the risk of a single point of failure.
- SHA encryption which secures data.
- Availability is guaranteed since all nodes maintain a copy of the blockchain.
- Consensus mechanisms guarantee that data is not malicious and that all nodes agree on the addition of the data.

Q: What is a smart contract and how does it work on Ethereum?

A: A smart contract is a piece of code that will run on nodes connected to a blockchain. On ethereum, these smart contracts can be deployed on the blockchain through the Ethereum Virtual Machine (EVM). As a node executes a smart contract, they are rewarded with gas, which can be converted to Ether. A smart contract usually initiates a transaction. Once this transaction is executed, the rest of the nodes on the Ether blockchain must verify its validity and truth.

Q: Does Blockchain utilize smart contracts? If so, what languages do they support?

A: Some blockchains do utilize smart contracts. The blockchain running on the Ethereum platform supports smart contracts coded in Solidity.

Q: What is gas?

A: Gas is a currency that is rewarded to nodes that use their computing power to execute the smart contract. The more complex the smart contract the more gas it must pay the node executing the code.

Q: Can smart contracts be malicious?

A: Yes. A smart contract, if programmed to do so, can move tokens without authorization. ERC20 tokens merely get assigned to an address within an internal ledger with the contract where ownership rights are assigned to the said address. To combat this make sure that any smart contract you use, is verified by etherscan.

Q: What is an ERC20 token?

A: The Ethereum Token Standard (ERC20) is used for ethereum smart contracts. ERC20 defines a common list of rules that an Ethereum token has to implement. This gives developers the ability to program how new tokens will function within the Ethereum ecosystem. ERC stands for Ethereum Request for Comments.

Q: What type of encryption is used by a blockchain?
A: SHA256 (Secure Hash Algorithm)

Q: How does the Blockchain ensure confidentiality, integrity, and availability of its data and transactions?
A: Confidentiality is guaranteed since all of the transactions are only accessible to the nodes on the network. Integrity is guaranteed as all data is encrypted and cannot be altered once added to the blockchain. Availability is guaranteed as the blockchain cannot be taken down and data cannot be lost since the blockchain is distributed and all nodes on the network maintain an active copy of the blockchain.

Q: Is the Blockchain Byzantine Fault Tolerant or Asynchronous Byzantine Fault Tolerant?
A: This is dependent on the platform/company that is utilizing and implementing the blockchain. There are blockchain consensus protocols that are Byzantine Fault Tolerant.

Q: How does Blockchain prevent against data loss and tampering?
A: The blockchain prevents against data loss and tampering through its consensus mechanism, distributed structure, and encryption/hashing algorithms.

Q: What attacks is Blockchain resistant from?
A: Sybil attacks, double spending, 51% attack, DDoS, and cryptographic attacks.

Q: What is the difference between proof-of-stake, proof-of-work, and proof-of-authority?
A: Proof-of-work, proof-of-stake, and proof-of-authority are all consensus algorithms that can be implemented on a blockchain to verify transactions. The most commonly used proof algorithm is proof of work.

Proof of work's consensus algorithm says that more than half of the nodes connected to the distributed system must perform complex computations (hashing algorithms) in order to verify a change within the distributed system. The larger the distributed system the more secure proof of work becomes, but also the slower it is to reach consensus.

Proof of stake consensus algorithm gives the consensus power to the nodes connected to the distributed network with the most invested assets. This algorithm makes it difficult for malicious entities to exploit the distributed system because they would need to have a very large amount of resources to sway the

consensus. It operates under the impression that those with a large ‘stake’ in the distributed system won’t want to exploit it because they will lose assets.

Proof of authority is another consensus algorithm for distributed systems that is specifically useful within private businesses and corporations. In this algorithm, there are designated ‘authority nodes’. This allows for a business or corporation to uphold a distributed system without relying on anonymous users to verify.

Q: What is mining? How is it done on a Blockchain?
A: Mining is the process of a node performing work to validate transactions on the blockchain. As miners validate transactions, they are rewarded with a digital asset. When a transaction is sent on the blockchain, every node on that network validates it through some validation rules set by the creators of that blockchain. Blocks are connected to each other by hashes that are computed from all of data within each previous block. The work done to compute this hash is referred to as mining.

Q: How is reporting done on Blockchain?
A: Once you have made a transaction a third party application will give you confirmation of your transaction (such as Mist Wallet, IOTA Wallet, Exchange sites, and Metamask)

Q: How does the Blockchain maintain a transaction history?
A: The transaction history is stored on full nodes upholding and mining the blockchain.

Q: What is Ethereum?
A: Ethereum is an open-source, public, blockchain based distributed computing platform and operating system featuring smart contract functionality. The Ethereum Network employs an Ethereum blockchain that has a digital asset called Ether (ETH). This platform allows for users to create applications that utilize consensus technology in order to create solutions to a myriad of problems.

Q: What proof system does Ethereum use?
A: Ethereum uses a Proof Of Work consensus protocol. This protocol can successfully handle byzantine failures, given the blockchain is sufficiently large enough. In a proof of work system, nodes connected to the blockchain must generate blocks using a hashing algorithm which takes a fair amount of computing power to do. In order for a malicious entity to exploit the blockchain they would have to use an enormous amount of computing power to sway the consensus.

Q: What is the Ethereum Virtual Machine?

A: The Ethereum Virtual Machine is what you use to connect to the Ethereum blockchain. When you connect to the EVM, you can start mining Ethereum for Ether and Gas. It is through the EVM that smart contracts will interact with your computer and through EVM that you yourself can deploy smart contracts. The EVM is what creates the P2P connection that is inherent on the blockchain.

9.2: Hashgraph

Q: What is a hashgraph?
A: A hashgraph is a data structure and consensus algorithm. It is fast, secure, and fair.

Q: How does a Hashgraph work?
A: A hashgraph uses Gossip about Gossip and Virtual Voting techniques to achieve fast, secure, and fair consensus among nodes.

Q: What is Gossip about Gossip?
A: Calling any random node and telling that node everything you know that it doesn't know. It refers to attaching a small additional amount of information that consists of two hashes that contain the last two people the node communicated with (gossiping about the information gossiped).

Q: Does Hashgraph use a cryptocurrency?
A: There is no public ledger or cryptocurrency for Hashgraph. This technology is currently being implemented on permissioned/private networks.

Q: What does Byzantine Fault Tolerance guarantee?
A: We know when we'll reach a consensus, have a guarantee that we'll reach that consensus, and have math proofs that make no assumptions about the Internet speed which could be impacted by firewalls, DDoS, viruses, botnets, etc. There is also fair ordering and time stamping on every event.

Q: How fast is a Hashgraph?
A: A Hashgraph is very fast and has a low consensus latency. This lets it have a large range of use cases and applications. Performance tests are currently being conducted on this new technology.

Q: What is Byzantine Fault Tolerance (BFT)?
A: BFT is a consensus algorithm that guarantees a moment in time that all participants reach a consensus, know that a consensus has been reached, and they are never wrong. With Blockchain's Proof-of-Work,

participants slowly become more confident of consensus with every block confirmation. There are different levels of BFT which depend on network assumptions and transmissions of messages. The strongest form of BFT is asynchronous.

Q: What is Asynchronous BFT?
A: Asynchronous BFT allows for malicious actors controlling the network and deleting/slowing down messages of their choosing. The only assumptions made are that less than 1/3 of the nodes on the network are attackers and some messages are transmitted over the internet. Some systems are partially asynchronous and are only secure if the attackers don't have too much power and don't manipulate message timing as much.

Q: How does hashgraph prevent Sybil attacks?
A: A Sybil attack is an attempt to compromise the network by creating large numbers of identities that are directed to act in collusion to inappropriately impact the network. Protecting against Sybil attacks can be done by appropriately allocating and weighting votes of different nodes.

Q: How does Hashgraph prevent against DDos attacks?
A: A Distributed Denial of Service (DDoS) attack occurs when its possible to disrupt transaction flow for an entire network by targeting a single or a few computers. Hashgraph doesn't use Proof-of-Work or have a leader that will solve a hash and publish a block like with Blockchain. Leader-based systems similar to the Blockchain give special permissions to a node and are very susceptible to DDoS since the current leader is a bottleneck and can be targeted. So, Hashgraph provides DDoS resilience without the inefficiency and cost of Proof-of-Work.

Q: How is a Hashgraph fair?
A: Fairness is the ability to prevent ordering of transactions from being unduly manipulated. Hashgraphs are fair because they serialize all transactions with cryptographic timestamping. This is different from the Blockchain since Blockchain miners determine the order in which transactions are placed in each block. Hashgraphs order transactions according to the median timestamp of when the node population received them to ensure that they are recorded fairly.

Q: What is the Hedera hashgraph platform?
A: The Hedera hashgraph platform provides a new form of distributed consensus; a way for people who don't know or trust each other to securely collaborate and transact online without the need for a trusted intermediary. The platform is lightning fast, secure, and fair, and, unlike some blockchain-based platforms,

doesn't require compute-heavy proof-of-work. Hedera enables and empowers developers to build an entirely new class of distributed applications never before possible.

Q: What is the Hedera Hashgraph Council?

A: The Hedera Hashgraph Council will be the governing body of the Hedera hashgraph network. The council will consist of up to 39 leading organizations and enterprises in their respective fields, with membership designed to reflect a range of industries and geographies, to have highly respected brands and trusted market positions, and to encourage a wide variety of perspectives. The Governing Members will elect the Governing Board and also contribute expertise through subcommittee membership. Hedera's governance terms ensure no single member will have control, and no small group of members will have undue influence over the body as a whole.

Q: Is the Hedera hashgraph network decentralized?

A: All Governing Members of the Hedera Hashgraph Council will have equal voting rights and all except Swirlds, Inc. will be limited to a three year term with a limit of two consecutive terms. (Swirlds, Inc., the owners and licensor of the hashgraph technology, will retain a permanent seat on the council.) The highly distributed network will expand to many millions of nodes voting on the order of transactions across at least five continents. This separation of governance from consensus is designed to ensure continued decentralization over time.

Q: Is there a Hedera hashgraph cryptocurrency?

A: The Hedera hashgraph network will have a native cryptocurrency, which is a utility token that grants token holders access to distributed applications on the platform. The token may also be "staked" and used to run a node (that is, adding CPU to the Hedera public network), thereby providing the network security within the public ledger. We expect the token to act as a unit of value to motivate responsible use and governance of the platform.

Q: Is the source code open source?

A: The Hedera hashgraph source code will be open review, which provides transparency and allows for contribution, while also bringing stability to the network by preventing forking. Forking, or splitting, is a prevalent issue with some public ledger platforms as it artificially inflates supply.

Q: Is the Hedera hashgraph platform patented?

A: Yes, Swirlds has granted Hedera hashgraph a license to the hashgraph consensus algorithm.

Q: What role does Swirlds have in the Hedera hashgraph platform?

A: Swirlds is the licensor of the underlying hashgraph technology that enables the Hedera hashgraph platform, and will continue to develop the technology. Swirlds is a member of the Hedera Hashgraph Council and will have the same voting rights as every other Governing Member. Prior to formal launch of the platform and the council, Swirlds may retain control of governance and network development.

Q: Do I require a license to develop on the Hedera hashgraph platform?

A: No, you do not require a license or need permission to use, or develop on, the Hedera public network. Applications that are developed on the platform can implement any licensing model.

Q: Are there transaction fees?

A: Yes, token holders will pay transaction fees to access distributed applications on the Hedera hashgraph platform. These transaction fees are paid to Hedera and the nodes in the network that contribute to the consensus mechanism.

Q: What programming languages are available for developers?

A: The Hedera hashgraph platform will support Java™ and Solidity™.

Q: Who can run a node?

A: The Hedera hashgraph platform will start with a small number of nodes during the testing phase, but we anticipate this will become available to anyone who wants to host a node (and meets basic requirements for bandwidth, CPU, and storage) in the future.

Q: Is there a test network?

A: There is a test network currently running for Hedera Hashgraph Council members.

Q: When will the Hedera hashgraph platform go live?

A: We expect the beta will go live in 2018.

Q: What is bank-grade consensus?

A: Hashgraph is the only bank-grade consensus algorithm as a result of the following properties: Mathematical proof of asynchronous Byzantine fault tolerance; Resilience to DDoS attacks, network partitions, sybil attacks and firewall/virus attacks; and Mathematical proof of fairness of ordering, access, and timestamps.

Q: Why is Hashgraph patented?

A: Hashgraph is currently only available on a private network so its patents allow for market advantage in enterprise / commercial applications. This is not designed to stifle creativity or expansion of the emerging ecosystem, but to protect technological innovations that took years to develop.

9.3: Tangle

Q: What is IOTA?

A: IOTA is an open-source distributed ledger protocol launched in 2015 that goes 'beyond blockchain' through its core invention of the blockless 'Tangle'. The IOTA Tangle is a quantum-resistant Directed Acyclic Graph (DAG), whose digital currency 'iota' has a fixed money supply with zero inflationary cost. IOTA uniquely offers zero-fee transactions and no fixed limit on how many transactions can be confirmed per second. Scaling limitations have been removed, since throughput grows in conjunction with activity; the more activity, the more transactions can be processed and the faster the network. Further, unlike blockchain architecture, IOTA has no separation between users and validators (miners / stakers); rather, validation is an intrinsic property of using the ledger, thus avoiding centralization. IOTA is focused on being useful for the emerging machine-to-machine (m2m) economy of the Internet-of-Things (IoT), data integrity, micro-/nano- payments, and other applications where a scalable decentralized system is warranted.

Q: What is a seed?

A: A seed is a unique identifier that can be described as a combined username and password that grants you access to your IOTA. Your seed is used to generate the addresses and private keys you will use to store and send IOTA, so this should be kept private and not shared with anyone. If anyone obtains your seed, they can generate the private keys associated with your addresses and access your IOTA.

Q: What is a non reusable address and how does it work for IOTA's Tangle?

A: Contrary to traditional blockchain based systems such as Bitcoin, where your wallet addresses can be reused, IOTA's addresses should only be used once (for outgoing transfers). That means there is no limit to the number of transactions an address can receive, but as soon as you've used funds from that address to make a transaction, this address should not be used anymore.

Q: How does Tangle reach consensus?

A: Apart from the data structure, the other major difference is how IOTA achieves consensus and how transactions are made. As mentioned previously, there are no miners. What this means is that each

participant in the network that wants to make a transaction has to actively participate in the consensus of the network by approving 2 past transactions. This attestation on the validity of two past transactions ensures that the network achieves consensus on the current state of approved transactions, and it enables a variety of unique features that are only seen in IOTA.

Q: How is Tangle scalable?
A: IOTA can achieve high transaction throughput thanks to parallelized validation of transactions with no limit as to the number of transactions that can be confirmed in a certain interval.

Q: How does decentralization work for Tangle?
A: IOTA has no miners. Every participant in the network that is making a transaction, actively participates in the consensus. As such, IOTA is more decentralized than any Blockchain.

Q: How is Tangle quantum-immutable?
A: IOTA utilized a next generation trinary hash function called Curl-p, which is quantum immune (Winternitz signatures).

Q: What makes Tangle different from the Blockchain?
A: Both are decentralized consensus systems, but with different consensus protocols. IOTAS Tangle is set up as a directed acyclic graph (DAG) and the blockchain is set up as a chronological list of connected merkle trees. DAG consensus differs from Blockchain in the same way that linked lists differ from arrays; they are two different data structures. The Tangle achieves consensus by confirming nearby transactions, where the blockchain confirms the validity of all transactions and all nodes must agree on that.

Q: How does Tangle prevent against data loss and tampering?
A: Like blockchain, the tangle has a consensus protocol which protects the truth of the ledger.

Q: Does Tangle have any performance and maintenance requirements?
A: The Tangle receives snapshots as maintenance for when the Tangle gets too large. See our guide for more information.

Q: Does Tangle have any special capabilities or limitations?
A: The Tangles capabilities are as boundless as human imagination but most DAG consensus protocols are specially designed to thrive in an Internet of Things industry. It is limited by adoptance; the less adopted the less valuable.

Q: Are there any projects under Tangle that are Byzantine Fault Tolerant?

A: No, not at this time.

Q: Are there any projects under Tangle that are Asynchronous Byzantine Fault Tolerant?

A: No, not at this time.

Q: Is it possible to create a private testnet environment with Tangle?

A: You can create a private testnet using IOTA, which is what utilizes the tangle technology.

Q: Does Tangle have its own cryptocurrency?

A: You can use the tangle to create a tangle-based cryptocurrency. The most prominent today is IOTA. (There are other companies using DAG, like Byteball and Constellation).

Q: What attacks is Tangle resistant from?

A:

- Sybil Attacks
- Large Weight Attacks
- Double Spending Attacks
- Parasite Chain Attacks
- Splitting Attacks

Q: Does Tangle utilize smart contracts? If so, what languages do they support?

A: DAG technology can utilize smart contracts. ByteBall is one such company utilizing smart contracts and DAG to perform transactions with conditions. It doesn't have a coding language for the smart contracts, rather it has predefined contracts that users can use as they please.

Q: What are some special use cases/applications for Tangle?

A: Digital Twins in a Machine based economy. Basically, every machine has a digital representation of themselves, and that is their digital twin. You can keep record and make transactions in a machine based economy using this idea.

Q: Does Tangle utilize mining? If so, how is that done?

A: No, there is no mining in Tangle. It's consensus algorithm does not call for it.

Q: How can someone search, add, or view their data using Tangle?

A: They would have to set up a permanent node and a storage place for every update of the tangle.

Q: How does Tangle ensure confidentiality, integrity, and availability of its data and transactions?

A: Similarly to all other decentralized consensus systems, the tangle is innately tamperproof and valid. To break the tangles confidentiality, integrity, or availability a malicious entity would need to have an unreasonable amount of CPU resources.

Q: How do you know if your transaction is verified on the Tangle?

A: To send a transaction using Tangle technology you will need to use an application, for example the IOTA wallet. Once a transaction is received and verified, it will show up in the history of the IOTA wallet. You can run a full node to view the full history of the tangle, as the light node will only report your history.

Q: Are there blocks on Tangle?

A: No, there are only 'transactions'. Each transaction is connected to other transactions, verifying them and being verified as well. These generate the links of the DAG, all which are hashed together similarly to blockchains hashing.

Q: Are there any hardware requirements for using Tangle?

A: 2 GB Ram recommended for hosting a full node. The more peers connected to the full node the less hardware requirements are needed as the verifying transaction resource cost is split amongst the nodes.

Q: How does Tangle mitigate any failure on the network?

A: IOTAs Tangle uses a consensus algorithm to protect from the various attacks such as quantum attacks, double spending, parasite attacks, sybil attacks, and splitting attacks.

Q: How does Tangle maintain a transaction history?

A: The Tangle maintains a transaction history across all full nodes, and it is viewable in applications such as IOTA's wallet.

Q: What is a snapshot? When and why do they happen?

A: The concept is to prune the history of nodes but save the balance, easing the burden of many nodes and retaining the validity of the DAG. This will make each transaction start fresh from a valid data set. (Note; IOTA is developing permanodes to keep the entire history of the DAG which will not be affected by snapshots). A snapshot occurs to prevent IOTAs Tangle from getting to large. IOTA has plans to do

automated snapshotting in accordance with the Tangles growth.

Q: What happens if the Tangle structure grows quickly? How is the data stored?
A: The data is stored on full nodes and permanodes. As the data structure grows, theoretically we have more transactions being verified then coming in (as one transaction must verify two), thus mitigating the risk of the Tangle getting uncontrollably large. If the Tangle is getting two large in terms of the amount of data being stored on each node is too much, then a Snapshot can occur which will reduce the size of the Tangle.

Q: How does pruning happen on the Tangle? What is pruning?
A: Pruning occurs in Snapshots. Basically, each full node is pruned and data pertaining to transaction history is erased, with balances of each seed being saved and retained. Pruning refers to trimming down, and in this case it refers to trimming down the data required for each full node to store. (note: permanodes are never pruned)

Q: How will the tangle be sharded?
A: There are no plans for IOTA's tangle to be sharded.

Q: What causes the Tangle to split into multiple subtangles? How are they managed?
A: A subtangle is an offline tangle created by a full node. This subtangle can then become verified by a transaction on the main tangle, which in turn verifies every previous transaction in that subtangle. These are managed with the tip selection algorithm which favors newly created transactions (a subtangles transaction is reference on the transaction it last verified before the subtangle was built).

Q: How does IOTA protect against an attack that would continuously split the Tangle?
A: See the whitepaper on the Tangle, specifically the section regarding the splitting attack.

Q: What advantages does tangle have over the blockchain?
A: The Tangle has is very scalable and claims to have fee less, almost instantaneous transactions. The biggest advantage is that as the Tangle grows, speed does not suffer. Where as with blockchains, the larger the blockchain gets, the more slow it becomes. Because of this speed, the Tangle is more prone for use in an internet of things environment.

Q: What is IOTA's main goal?
A: IOTA's main goal is to become a leader in the micro transaction and Internet of Things industry.

Q: How high will the proof-of-work have to be for a transaction in order for the network to be secure against attackers?

A: The Tangle does not use Proof of Work to protect against attackers; it uses a tip selection algorithm and a weight based transaction system to defend against attackers. The proof of work does help mitigate spammers.

Q: Is the Tangle quantum-resistant?

A: Yes.

Q: Does each full node store the entire Tangle since the last snapshot?

A: Once the newest snapshot is saved, that's the only thing in the Tangle. A Full node can however store previous snapshots on external hard drives if so desired.

Q: What does the coordinator do on a Tangle?

A: The coordinator acts as a pair of training wheels to protect it from outside attacks in its early stages of infancy. Because the Tangle isn't sufficiently supported to be able to withstand all attacks, the Coordinator acts as it's guardian. Once the Tangle is sufficiently supported, IOTA will get rid of the Coordinator (in fact the coordinator is not needed currently, but stays because the risk is still too significant).

Q: Where does the complete ledger persist if a user transaction is responsible for verification to further the network? Do they exist on persistent server nodes and if so, what is the incentive to run this type of node?

A: Full nodes is where the complete ledger resides, and light nodes assist by connecting to full nodes. There is no incentive for running a full node other than contributing to the safety and authenticity of the network.

Q: How does verification work? What kind of search or filter algorithm is used to make sure that all the transactions of an address are collected and the total sum/difference of their balance is properly calculated?

A: To issue a transaction, that node must verify two other transactions.Verification differs from DAG to DAG but the Tangle uses a tip selection algorithm to determine which new transactions are chosen to be verified .

Q: How does Proof of Work in IOTA work? Does it search for the nonce for the right number of leading

zeros like Bitcoin?

A: The proof of work is minimal. You need low proof of work for internet of things devices, and there is no mining in IOTA, thus there is no need for a proof of work that is adjustable (there is no incentive to mine). The only proof of work that is there is just the work required to send out and verify transactions.

Q: Is the IOTA node software open source?

A: Everything, except the coordinator, is open source.

Q: If there is no mining in IOTA and the supply is set, who owns them?

A: There was an Initial Coin Offering for IOTA. After that, the coins have been moved from people trading with each other.

Q: Does the lack of a Blockchain make Tangle insecure?

A: No. The blockchain and the tangle are two different consensus systems entirely.

9.4: Hyperledger

Q: Where can I learn more about hyperledger?

A: You can visit hyperledgers homepage at: https://www.hyperledger.org/

Q: Who created Hyperledger?

A: It was created by the Linux Foundation in December 2015.

Q: What is Hyperledger?

A: Hyperledger is a organization whose foundation is based around improving blockchain like technologies. Hyperledger incubates blockchain projects and provides a set of guidelines their architecture should follow. Projects are open source and collaborate together.

Q: What is Hyperledger's main goal?

A: Their main goal is to provide a platform for blockchain technologies to grow, collaborate, and standardize components and design necessary for the technology.

Q: Why are there different projects of hyperledger?

A: Hyperledger incubates projects as they are proposed to the organization. Each project is its own architecture, however they all follow the architectural standards that hyperledger decides necessary. Each

project is built upon these standards but may have very different uses.

Q: What are the Hyperledger Projects?
A: There is currently Hyperledger Fabric, Sawtooth, Burrow, Iroha, and Indy. As well as tools being Hyperledger Cello, Composer, Explorer, and Quilt.

Q: What is Hyperledger Fabric?
A: Fabric is an architecture focused on a permission based blockchain with high modularity across its components.

Q: How does Hyperledger Fabric compute consensus?
A: Fabric uses permissioned-voting and breaks consensus into 3 phases: Endorsement, ordering, and validation. A transaction is endorsed by a set policy such as a number of signatures. Ordering then commits the transactions, and validation verifies it.

Q: What makes Hyperledger Fabric unique?
A: Fabric incorporates a channel system where data can be made obscur by having it only shared by nodes within a channel. Fabric can also be adjusted to determine what data is shared within the channels, and what is shared to all nodes.

Q: What is Hyperledger Sawtooth?
A: Sawtooth is an architecture built around the idea of a core for applications to be built off. Essentially sawtooth can run different types of blockchains and even multiple blockchains on the same network.

Q: How does Hyperledger Sawtooth compute consensus?
A: Sawtooth supports multiple types of consensus to be used, ranging between Proof of elapsed time(PoET), PoET simulator, and a Dev mode.

Q: What makes Hyperledger Sawtooth Unique?
A: Sawtooth offers the ability to run multiple blockchains off the same core. Meaning multiple applications can run on the same system. Sawtooth also features a parallel scheduler that maintains the blockchain so transactions can run simultaneously.

Q: What is Hyperledger Burrow?
A: Hyperledger Burrow is a client that is designed to execute Ethereum smart-contracts.

Q: What is Hyperledger Iroha?

A: Hyperledger Iroha is aimed to create reusable components and provide an API to support C++ as well as mobile environments.

Q: How does Hyperledger Iroha compute consensus?

A: Iroha computers consensus through a Byzantine fault tolerant algorithm called Sumeragi. This means that a validation peer must sign the transaction and then broadcast it to at least 2f+1 peers to sign.

Q: What makes Hyperledger Iroha Unique?

A: Iroha offers a unique implementation of C++ and mobile application support. Iroha also offers a peer reputation system to enhance trusting of peers.

Q: What is Hyperledger Indy?

A: Hyperledger Indy is a decentralized identity system to be incorporated into blockchains. It provides tools and components where independent identities can be created and maintained on a ledger.

Q: What is Hyperledger Cello?

A: Cello is a tool project aimed to reduce the effort of creating, managing and terminating blockchains.

Q: What is Hyperledger Composer?

A: Composer is a set of tools that allows for simple implementation of blockchains for business owners and developers.

Q: What is Hyperledger Explorer?

A: Explorer is a tool aimed to create a user-friendly web applications to interact with blockchains.

Q: What is Hyperledger Quilt?

A: Quilt is an implementation of interledger protocol, which is used to make transactions across different ledges.

Q: How does Hyperledger prevent against data loss and tampering?

A: Hyperledger states that all projects require cryptographic tools designed around confidentiality and privacy are made available. Hyperledger projects also inherently have blockchain security features through consensus and issuing protocols for each of its architectural layers. This includes privacy services,

P2P protocol that works with internet firewall, proxies, and other security features, as well as secure smart-contracts.

Q: Do Hyperledger projects have any performance and maintenance requirements?
A: Each project is a very different architecture from each other and offers varying ways to implement a blockchain. This makes performance and maintenance based on the implementation of the project.

Q: What are the required features hyperledger implements?
A: All projects must offer Private transactions and confidential contracts. Provide a concepts of identity and security for these identities. All projects muster have interoperability and portability. Hyperledger also has a major focus on modularity.

Q: What are the architectural features hyperledger states?
A:
Identity Services to manage participants.
Policy Services to provide configuration for component, access control and privacy.
Blockchain Services to provide p2p protocols, distributed ledgers and consensus.
Smart-Contract Layer to to provide security decentralized transactions.

Q: Do Hyperledger projects have any special capabilities or limitations?
A: Each project is designed around high modularity and customizability. Almost all projects can be fit to the users needs.

Q: Are there any projects under Hyperledger that are Byzantine Fault Tolerant?
A: Hyperledger Indy, Iroha, and Sawtooth all provide Byzantine fault tolerance.

Q: Is it possible to create a private testnet environments through Hyperledger projects?
A: Yes, almost all Hyperledger projects are intended to be used within a private system.

Q: Does Hyperledger have its own cryptocurrency?
A: Hyperledger does not natively have a cryptocurrency.

Q: What attacks are Hyperledger projects resistant from?
A: Hyperledger projects are inherently resistant to data tampering and data loss through decentralization. Hyperledger protocol also accounts for cryptography and tools to keep data and identities private when

needed. Many Hyperledger projects are designed for private networks and most require a doorman to grant verified nodes entry to the network, improving trust between nodes.

Q: Do Hyperledger projects utilize smart contracts? If so, what languages do they support?
A: Yes, Fabric, Sawtooth, Burrow, and Iroha all support smart contracts/chaincode. Each features different languages to implementing, from Fabric enabling any language, to Iroha using java.

Q: What are some special use cases/applications for Hyperledger projects?
A: Use cases are highly diverse from banking and financial services to supply chain support. A list of current hyperledger use cases can be found here:
https://wiki.hyperledger.org/groups/requirements/use-case-inventory

Q: Does Hyperledger utilize mining? If so, how is that done?
A: Hyperledger Sawtooth using a mining consensus called Proof of Elapsed Time, this method requires very little computational power and provides a fair consensus through a lottery.

Q: How can someone search, add, or view their data using Hyperledger projects?
A: Each hyperledger projects offers a different way of storing and sharing data, this includes customizability in how what data is shared among nodes, and what is kept private. Throughout the projects, data is held within the ledger and can be access by a queset and keys, based on the permission of the node attempting to access the data. Based on the consensus used within the project data is added differently and entirely depends on the use case of the system.

Q: How does Hyperledger ensure confidentiality, integrity, and availability of its data and transactions?
A: Hyperledger offers a wide range of standard cryptography and privacy policies for the projects to follow. The projects allow for data to be private and confidential. The ledger technologies and consensus algorithms ensure integrity, and the data that is deem available to a node is able to be accessed by this node. These features follow true for all projects within hyperledger.

Q: How is validation done through Hyperledger projects?
A: Hyperledger projects offer a consensus module by design that is responsible for confirming transactions. This module is pluggable and configurable in order to offer different policies and edit the policies of the module used. Validation is based upon what type of consensus is used and the policies of the system, it is decided by the developer of the system.

Q: What are the main differences between Hyperledger's framework projects?

A:

Hyperledger Burrow is built specific to the Ethereum Virtual Machine.

Hyperledger Fabric offers configuration of nodes, smart contracts, consensus and membership services so it may be adjusted to the use case needs.

Hyperledger Iroha is focused on mobile applications and C++ support

Hyperledger Sawtooth uses a different consensus method called Proof of Elapsed Time and is focuses on running multiple blockchains on a single core.

Q: Are blocks mined in Hyperledger?

A: No, many projects rely on a validating peer and do not mine blocks.

Q: Are there any hardware requirements for using Hyperledger projects?

A: Hardware requirements must be assessed by the needs of the use case and what framework will be used.

Q: How does Hyperledger mitigate any failure on the network?

A: Because of distributed ledger technology the data is stored throughout the nodes of the network, if a node is lost, the data is not.

Q: How do Hyperledger projects maintain a transaction history?

A: Many hyperledger projects rely on storing transactions within a ledger, however different from many other blockchains. Hyperledger projects offer the ability for nodes to maintain their own private ledger as well as deciding what information is shared to other nodes. Because of hyper ledgers core feature of modularity, the way transaction history is stored depends on the needs of the use case and can change to what is necessary.

Q: What are the main differences between Hyperledger projects and other Blockchains?

A: Hyperledger focuses on private use cases, with high modularity, and no native cryptocurrency.

Q: Why are Hyperledger projects better than existing Blockchains?

A: Hyperledger projects are specifically designed for business needs with no inherent cryptocurrency. This means that hyperledger can focus on the needs that many existing blockchains fail to meet by diverting their focus into improving the distributed ledger technologies. These Projects are also designed around Hyperledgers standard features and provides a collaborative environment. This means these projects are

highly customizable, scale better, and computer transactions must fast, and can be adjusted to the use cases needs.

Q: Is Hyperledger quantum-resistant?
A: Hyperledger is not quantum-resistant, however many of its frameworks are designed around private systems unlike cryptocurrencies. Gaining access to this system by breaking through non-quantum resistant crypto barriers may still be a threat.

Q: How many peers in the network need to endorse a transaction?
A: The number of peers required to endorse a transaction is driven by the endorsement policy that is specified at chaincode deployment time.

Q: Does an application client need to connect to all peers?
A: Clients only need to connect to as many peers as are required by the endorsement policy for the chaincode.

Q: How do I ensure data privacy?
A: There are various aspects to data privacy. First, you can segregate your network into channels, where each channel represents a subset of participants that are authorized to see the data for the chaincodes that are deployed to that channel. Second, within a channel you can restrict the input data to chaincode to the set of endorsers only, by using visibility settings. The visibility setting will determine whether input and output chaincode data is included in the submitted transaction, versus just output data. Third, you can hash or encrypt the data before calling chaincode. If you hash the data then you will need to provide a means to share the source data. If you encrypt the data then you will need to provide a means to share the decryption keys. Fourth, you can restrict data access to certain roles in your organization, by building access control into the chaincode logic. Fifth, ledger data at rest can be encrypted via file system encryption on the peer, and data in-transit is encrypted via TLS.

Q: Do the orderers see the transaction data?
A: No, the orderers only order transactions, they do not open the transactions. If you do not want the data to go through the orderers at all, and you are only concerned about the input data, then you can use visibility settings. The visibility setting will determine whether input and output chaincode data is included in the submitted transaction, versus just output data. Therefore, the input data can be private to the endorsers only. If you do not want the orderers to see chaincode output, then you can hash or encrypt the data before calling chaincode. If you hash the data then you will need to provide a means to share the

source data. If you encrypt the data then you will need to provide a means to share the decryption keys.

Q: How do application clients know the outcome of a transaction?
A: The transaction simulation results are returned to the client by the endorser in the proposal response. If there are multiple endorsers, the client can check that the responses are all the same, and submit the results and endorsements for ordering and commitment. Ultimately the committing peers will validate or invalidate the transaction, and the client becomes aware of the outcome via an event, that the SDK makes available to the application client.

Q: How do I query the ledger data?
A: Within chaincode you can query based on keys. Keys can be queried by range, and composite keys can be modeled to enable equivalence queries against multiple parameters. For example a composite key of (owner,asset_id) can be used to query all assets owned by a certain entity. These key-based queries can be used for read-only queries against the ledger, as well as in transactions that update the ledger. If you model asset data as JSON in chaincode and use CouchDB as the state database, you can also perform complex rich queries against the chaincode data values, using the CouchDB JSON query language within chaincode. The application client can perform read-only queries, but these responses are not typically submitted as part of transactions to the ordering service.

Q: How do I query the historical data to understand data provenance?
A: The chaincode API GetHistoryForKey() will return history of values for a key.

Q: How to guarantee the query result is correct, especially when the peer being queried may be recovering and catching up on block processing?
A: The client can query multiple peers, compare their block heights, compare their query results, and favor the peers at the higher block heights.

Q: Does Hyperledger Fabric support smart contract logic?
A: Yes. We call this feature Chaincode. It is our interpretation of the smart contract method/algorithm, with additional features. A chaincode is programmatic code deployed on the network, where it is executed and validated by chain validators together during the consensus process. Developers can use chaincodes to develop business contracts, asset definitions, and collectively-managed decentralized applications.

Q: How do I create a business contract?

A: There are generally two ways to develop business contracts: the first way is to code individual contracts into standalone instances of chaincode; the second way, and probably the more efficient way, is to use chaincode to create decentralized applications that manage the life cycle of one or multiple types of business contracts, and let end users instantiate instances of contracts within these applications.

Q: How do I create assets?

A: Users can use chaincode (for business rules) and membership service (for digital tokens) to design assets, as well as the logic that manages them. There are two popular approaches to defining assets in most blockchain solutions: the stateless UTXO model, where account balances are encoded into past transaction records; and the account model, where account balances are kept in state storage space on the ledger. Each approach carries its own benefits and drawbacks. This blockchain technology does not advocate either one over the other. Instead, one of our first requirements was to ensure that both approaches can be easily implemented.

Q: Which languages are supported for writing chaincode?

A: Chaincode can be written in any programming language and executed in containers. The first fully supported chaincode language is Golang. Support for additional languages and the development of a templating language have been discussed, and more details will be released in the near future. It is also possible to build Hyperledger Fabric applications using Hyperledger Composer.

Q: Does the Hyperledger Fabric have native currency?

A: No. However, if you really need a native currency for your chain network, you can develop your own native currency with chaincode. One common attribute of native currency is that some amount will get transacted (the chaincode defining that currency will get called) every time a transaction is processed on its chain.

9.5: Corda

Q: Where can I learn more about Corda?

A: You can visit Corda at https://www.corda.net/

Q: Who created Corda?

A: It was developed by a company called R3.

Q: What is Corda?

A: Corda is an open source distributed ledger project designed around business needs with functionality similar to blockchains.

Q: What are the key features of Corda?

A: Corda features a network, this network consist of nodes. Each node manages a ledger of transactions, and these ledgers are verified through a notary consensus system.

Q: What is Cordas network?

A: Corda is a network of authorized peer to peer nodes. Meaning that all nodes within the network were approved to be in the network. Corda also shares data on a need-to-know basis and has no global broadcasting.

Q: What is a Corda node?

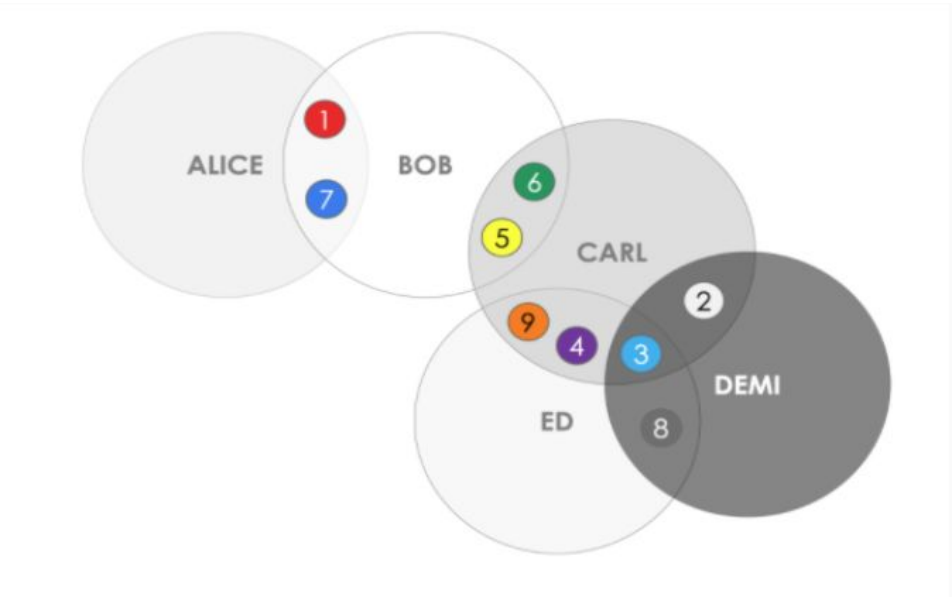
A: Each node has a unique identity as well as layers to store data of transactions and interfacing with the network to connect to other nodes.

Q: How does Corda store data?

A: Corda features a distributed ledger that each node maintains the data it knows as facts. If node A has separate transactions with node B and C, node B will only see transactions between A and B, and node C will only see transactions between A and C.

Q: How does Corda store data?

A: Corda features a distributed ledger that each node maintains the data it knows as facts. If node A has separate transactions with node B and C, node B will only see transactions between A and B, and node C will only see transactions between A and C.



Q: What is Cordas Notary System?

A: The notary system provides consensus to the network through validation of uniqueness. This prevents double spending on a transaction. The notary system can support multiple types of consensus, as well as many notaries running at the same time.

Q: What makes Corda unique?

A: Corda offers a different approach to distributed ledgers than blockchains. Unlike blockchains, the ledger only shares the information between the entities that need to know of the transaction. This separates transactions from being consented as a whole blockchain, but rather between peers.

Q: What makes Corda different from the Blockchain?

A: Corda stores conducts transactions different from blockchains. As mentioned, corda links the states of transactions between nodes involved. This is different from global blocks shared to all entities.

Q: How does Corda prevent against data loss and tampering?

A: Corda has a “Vault” that can be compared to a wallet in a blockchain. It contains the relevant data for that node. Data is stored per node, if a node loses its private ledger, that data will be lost. There are plans for a recovery system but it has yet to be implemented. Corda prevents tampering by having a system of trusted nodes issuing transactions through its notary consensus system.

Q: Does Corda have any performance and maintenance requirements?

A: Corda features a notary system that runs on dedicated system. If this system gets overloaded a bottleneck may occur and more notary systems will be needed to account for the excess.

Q: Does Corda have any special capabilities or limitations?

A: Corda is capable of significantly faster transactions per second and the ability to scale better than most blockchains.

Q: Is Corda Byzantine Fault Tolerant or Asynchronous Byzantine Fault Tolerant?

A: Yes, corda is byzantine fault tolerant through its notary service.

Q: Is it possible to create a private testnet environment with Corda?

A: Yes, corda is designed to be used in a private system.

Q: Does Corda have its own cryptocurrency?

A: No, corda is designed around distributed ledgers rather than blockchains and does not have a cryptocurrency.

Q: What attacks is Corda resistant from?

A:
Sybil Attacks
Double Spending
Data Tampering

Q: Does Corda utilize smart contracts? If so, what languages do they support?

A: Yes corda utilizes smart contracts, they are written in Java and Kotlin.

Q: What are some special use cases/applications for Corda?

A: Corda specializes use cases in financial institutions and security.

Q: Does Corda utilize mining? If so, how is that done?

A: Corda does not mine, as consensus is done peer to peer and verified by a notary system.

Q: Does Corda use a Blockchain structure? If not, what structure do they use?

A: Corda does not use a blockchain structure. Instead they rely on a network of peers who commit transactions between each other and store their own ledger.

Q: How can someone search, add, or view their data using Corda?

A: Each peer has access to their own ledger and any other transactions made available to them. They are able to simple query their vault. In order to add data to the ledger a peer must form a transaction with another peer.

Q: How does Corda ensure confidentiality, integrity, and availability of its data and transactions?

A: Corda ensures confidentiality by keeping ledger data private between the entities that need to know. Integrity is kept by the consensus methods and notary system. Finally data known by a node is inherently available to them.

Q: How is validation done on Corda?

A: Validation on corda is done through two parts. The first is the use of trusted peers on a private network to ensure that transactions are authentic by nature. The second is through the use of a notary system in order to verify the transaction and prevent double spending.

Q: Are there blocks on Corda?

A: No there are no blocks in corda. Instead it links together states between peers.

Q: Are there any hardware requirements for using Corda?

A: Corda states that a single or dual core cpu, 4GB RAM and 25GB disk is the minimum requirement.

Q: How does Corda mitigate failure on the network?

A: Because corda is a decentralized ledger, if a node is disconnected the other nodes will not be affected.

Q: How does Corda maintain a transaction history?

A: Each node maintains its own copy of its need to know ledger.

Q: What are the main differences between Corda and the Blockchain?

A: Corda stores data differently than blockchains. Instead of blocks shared throughout the chain, states are only shared between peers that need to know or are involved with the transaction.

Q: Why is Corda better than the Blockchain?

A: Corda is better in the fact that transactions can be completed quickly and separate from each other. Corda drastically saves space as data is not shared through all nodes and is able to compute transactions significantly faster with no extra cost.

Q: What is Corda's main goal?

A: Cordas main goal is to provide distributed ledger technology that is cheap, eliminates time and effort, and can be used on a global scale throughout many industries to act as a global database.

Q: Is Corda quantum-resistant?

A: Yes.

Q: Does the lack of a Blockchain make Corda insecure?

A: The lack of a blockchain does not make corda insecure, however it does increase the risk of data lost.

Q: How does Corda prove that a party owns the cash or securities?

A: Corda lets you provide the full history of the money back to when it was issued or you can have the issuer re-sign/countersign every time that money is moved so the issuer signature stays with the asset, only ever having to reach back far enough to find a signature from the issuer. Once you're happy that you each own what you're saying you own, you both sign it and the transaction becomes valid, and private at that point.

Q: How does Corda solve the double spending problem?

A: Two transactions for the same thing can't happen (double spend) because Corda distinguishes between, is this transaction valid (an acceptable transaction) and is it confirmed (no double spend by notarization)? It sends a request to a notary cluster that along the lines, says, please confirm these assets haven't been spent. The notary then replies yes or no. This is operated either by a cluster using a Byzantine Fault-Tolerant algorithm or by a CSD. You can have several notaries in the system. When the transaction is confirmed, it is important to know it's confirmed with finality and you cannot go back. The notary clusters – likely in different jurisdictions – will likely be regulated and covered by things like settlement finality obligation, controls, etc. Therefore, Corda can allow several notaries on the system, so that you can use a regulated one when you need to and a non-regulated one when appropriate.

Q: What do you mean by physical separation of ledgers?

A: It means there should be physically distinct nodes for certain client segments.

Q: Where are smart contracts stored?

A: In Corda, every state object contains the identity of the Smart Contract that controls it. If I'm in a cash contract, I will point to the Smart Contract that requires payment. The code is written in Java and is

identified by a hash. The code can move around the network in that hash. Imagine it flowing across the network wherever the agreement goes. You can check by the hash that it is the correct version of the agreement, and hasn't been tampered with.

Q: How does Corda validate the amount of the transaction?

A: You can put a requirement in the verify() function of the contract of the output state stating that the sum of the value of all cash inputs must be of a certain value, that there are no other inputs, and that there is a single output.

Q: What is the role of homomorphic encryption in the financial services industry?

A: Homomorphic encryption have a role to play but as part of a broader overall strategy. Platforms that start by sharing data indiscriminately and inappropriately and then later encrypting it seems like the incorrect way to do it. For example, the equivalent would be if we do a loan, and you lend me the money. I'll then make 1,000 copies of the loan note and share it with everyone. That doesn't make sense. But what if I make the 1,000 copies and put them in 1,000 safes and then share those. Not everyone should be able to see it. But why share it if everyone can't see it? However, in our system, some data does have to be shared to prove it's correct, and then that creates a privacy leak. It's a balancing act. We've made it as secure as we can without advanced encryption. We have pluggable consensus – you can apply any style of verification to prevent double spend for each state. It is a small part called the verification step which in the future can be converted into an efficient zero knowledge proof. We have done work to allow us to use zero knowledge proofs in the future.

Q: How are documents related to transactions handled in Corda?

A: The proposed application can be configured to support uploading of documents for every individual trade. The documents so uploaded may be stored within the transaction, on public or private cloud, and also may be sent as part of notifications to the registered email id. The said documents would be accessible only to the relevant parties. Electronic documents can be easily uploaded to Corda nodes and then references to these documents referred to in the relevant Corda states. Any node that has access to the states can then request access to these documents which are then automatically made available. Corda supports all file types as they are saved as in a binary format. External Documents which are not stored as part of Corda can be referred to as corda textual references within corda state objects.

Q: What capability is present to bind legal clauses/documents to an instance of a workflow?

A: Approved and accredited contract code that runs on Corda nodes has references directly to the text (and/or a secure hash) of the legal documents that underpins the transactions on which it applies to.

Recall that in the Corda model, “code is not law”, but merely Corda catalyzes the processing between two parties. Additionally, at any time, if the relevant parties agree, the programmatic code can be upgraded in order to better reflect the underlying contract. A reference to the underlying contract must exist in every corda contract code written, or the developer will not be able to physically compile the code and it will not be approved for certification for running on an R3 approved network. In the case of dispute, the programmatic code will always defer to the legal clause and documents on which it was written for, and Corda will provide the option for parties to use the upgrade path mentioned in order to resolve inconsistencies.

Q: How is consensus achieved on Corda?

A: We designed Corda from first principles. We took the bits from “blockchain” that work and rejected the rest which were inappropriate for financial services use cases. Consensus: With Corda, the validity of a transaction is established only by the parties to the transaction in question. This is consensus over state validity. However, consensus over state validity does not prevent two seemingly valid transactions being incompatible, for example, the same state is used as input to two different transactions a “double spend”. The uniqueness service guarantees that the same state (which represents an asset or agreement) cannot be used for more than one transaction. When a transaction agreement is reached, the states used as inputs to the transaction are submitted to the uniqueness service for timestamping. Once timestamped, the uniqueness service ensures that any subsequent transactions cannot use those same states. The uniqueness service could be implemented in multiple ways; By a standalone centralised server, federated network or decentralised network By a regulator or consortium of participants Corda is designed to operate in a substantially different environment, for instance: All parties using Corda are assigned a cryptographic identity which will be tied to a “real world” identity, such as a legal entity identifier. It is envisaged that to join the Corda ecosystem, participants must undergo KYC procedures in order to verify these identities. All smart contracts include legal prose (such as an International Swaps and Derivatives Association agreement) which can be relied upon in the event of a dispute. In effect, all transactions are legally binding through the use of digital signatures.

Q: What are the different encryption and hashing techniques used?

A: Corda uses the following signature algorithms:

- RSA ECDA signature scheme using the secp256k1 Koblitz curve ECDA signature scheme using the secp256r1
- NIST P-256 curve EdDSA using ed25519 twisted Edwards curve
- SPHINCS-256 hash-based signature scheme (secure against post quantum attacks)

Corda combines the above signatures with SHA256 and SHA512 for hashing. Implementation of HMAC

and SHA3-512 are planned for in near future. Additionally data in flight is secured using SSL/TLS. Since Corda is built on a Relational database, it can leverage the capabilities provided by enterprise grade RDBMs e.g. Oracle to secure data at rest.

Q: Are there blocks in Corda?

A: Blocks are required in Bitcoin and other similar DLs to incentivise participants to validate transactions by pooling them together. Participants in proof of work blockchains must incur a cost to validate each other's transactions. At a minimum, validator's costs need to be offset. Given that Bitcoin supports micro payments, it may be the case that if transactions were not pooled into blocks then transaction validation fees may exceed the value of a payment. Therefore transactions are pooled into blocks, aggregating transactions fees, lowering the cost of validation for each individual transaction. In Corda, individual contract states reference prior contract states. We have no notion of proof-of-work. Instead notary services perform "uniqueness consensus" and will operate with their own SLAs in place, likely being compensated a fee for their services.

Q: How does Corda address privacy issues?

A: Corda was built with privacy concerns at the core. On Corda, data is only shared with those who need to know – those who are a party to the transaction and those with a need to see. Direct Quote – "Corda is designed to record and manage legal agreements between parties." Ledger is the union of all of these facts; on Corda there is consensus at the agreement level. Once there is consensus, the next question is who are you going to rely on to stamp and validate the transaction? You need a Notary to say yes that transaction occurred and has been committed. With Corda: Each fact is recorded separately - we specify who receives it, who approves changes, and when it's accorded a point of finality. We have digital signatures and business logic (contract code), which is similar to others. We also have differences – the key difference being data sharing. The all to all is for where there isn't a stable identity system and you cannot rely on trusted parties to memorialize transactions or perform services; it requires everyone to check everyone's work because there is anonymity, no regulation, and no trusted party.

Q: How does Corda scale in terms of number of banks & trade parties?

A: As the Corda transaction model transmit transactions on a peer to peer mode. This reduces the workload on any given node as it effectively partitioning the overall workload across the network. Thus, there is no "global speed limit" in Corda due to the architecture, so an increase in overall trade volume has no effect on two parties transacting. Persistence is achieved through the use of enterprise grade RDBMS which already include many scalability features e.g. partitioning, clustering etc. The notaries, which need to sign every transaction can be clustered or multiple notaries could be used. Additionally, Corda uses

standard technology components, including message queues (Apache Artemis Broker is included, but any AMQP-compliant queueing service may be used), and relational databases (via a JDBC interface). Apache Artemis, an embeddable message queue broke provides journaling, load balancing, flow control, high availability clustering, streaming of messages too large to fit in RAM, automatic delivery retries with backoff etc. Building on enterprise technology components allows Corda to scale up as the trade volumes go up. Addition of new banks/trading parties can easily be achieved by either adding the Bank/trading party as a node on the network which is running Corda and securely identifies and authenticates itself to the address and access resources of the network before being allowed to communicate with other peer nodes. As data transmission is done on a point to point basis, the increased number of active nodes has no effect (either on the volume of traffic or latency) on the communication between existing nodes apart from an entry in an address lookup table which is provided by the naming and address provider supplied by Corda.

Q: Is R3 actively researching legal implications of distributed ledger technology?
A: We have explicitly considered the legal implications of DLT in our design for Corda. In particular, agreements managed in Corda can explicitly link to overarching legal prose. This enables applications in Corda to bridge from the technical/DLT world to the legal world, providing the possibility that validly signed transactions on Corda can be considered legally binding. These considerations are a direct result of our painstaking, year-long process of architecture and design performed with our large, engaged and diverse membership. It explains why Corda is so unique: it is the only platform that has considered questions such as this from the start. We are researching legal implications of use of DLT generally, including data privacy, settlement finality, and identity. We are also actively researching compliance with existing regulations for specific use cases. In both efforts, we collaborate with several international and regional law firms as well as lawyers from our consortium members. We foresee the new technology in many ways fitting neatly into today's legal structure. However, there will be legal issues that need to be addressed, for example, settlement finality. Such issues may require a clarification or legislative change.

Q: How is reporting done on Corda?
A: The reports are run on a slave copy of the relational database (EnterpriseDB), thereby not risking to impact the OLTP performance. Since the master-slave replication of the database is done in real-time, any reports run will always depict the latest status and won't show outdated information. From a technology point of view, both online and batch reporting will be delivered as Microservices running on top of Spring Boot and leveraging opens source Java reporting frameworks like JasperReports, BIRT or Pentaho to speed up the development. For the estimates and based on the current insights JasperReports has been selected as the reporting framework. However, the final decision to select which reporting framework will be done

only during the technical design and after the first reports are defined and clear requirements regarding output formats and look and feel are defined. Since underpinning the reporting Microservices Docker is used to scale the overall solution, the number of container instances available for reporting functionality can be easily and dynamically scaled if required. So for example if at 23:00 extra compute resources are required to run the end of day reports, this can be easily catered for without having to scale other components. For the scheduling of the various batches including offline reports, like for example end of day reports, the Quartz scheduler will be leveraged.

Q: What are the failure modes of Corda?

A: The proposed solution takes a robust approach when it comes to redundancy and so all the proposed components are running in a clustered, high available configuration. Docker is leveraged to provide the underlying clustering, failover and scaling functionality all components are running in a multi container configuration. The following setup is selected for the various components: API Gateway is running as a software cluster and incoming traffic is being load balanced before it reaches the API gateway. HSM is running as an appliance cluster. Database is running as a master-slave configuration in which, in case of failure of the master, the slave will take on the role of master. All the microservices (business logic) are running on at least two containers per service and leveraging Docker Swarm to take care of load balancing. Corda is running in an active-active node configuration. Besides the above measurements, the underlying storage is also synced to the other data centre, if applicable, so that in case of disaster recovery the other datacenter can quickly be fully operational and take over the workload. In case the bank is hosting the solution on premise this same high availability can be achieved however it is up to the bank to achieve that. If for example a single node HSM is selected instead of a clustered approach there would be the introduction of a single point of failure.

FAQ Sources: [33], [34], [51], [53], [54]. [55], [59], [70]

Chapter 10: Blockchain Tutorials

Below is a list of tutorials to install and configure the programs you will need before starting development.

Installation Guides

Installing Bash for Windows 10

Installing Cygwin for Windows 7 and 8.1

Configuration Tutorials

Truffle Configuration

Lighttpd Configuration

Configuring Github for Visual Studio Code

Introductory Guides

Introduction to NPM

Command References

Linux Command Reference

Truffle Command Reference

Github Command Reference

Writing your First Solidity Contract

Writing a Solidity contract is very similar to writing in Javascript. If you aren't already familiar with Solidity, then refer to our Appendix for API and language documentation for Solidity to get started.

To begin, create a file with your desired filename and the extension .sol. Save this file. To start programming, we are going to declare the version pragma which is used to reject

compilation in future compiler versions. This helps to avoid the introduction of incompatible changes. For our solidity contract, we stated the version pragma at the start of the .sol file as follows:

```
pragma solidity ^0.4.4;
```

Next, we construct the contract body. To declare the empty contract, we wrote the following:

Contract addBlock {

}

Since this contract will be used to add blocks to the blockchain, we are going to need a structure to hold the block data. We define a new block structure within the contract body as follows:

```
struct NewBlock {
    uint8 age;
    uint id;
    bytes32 name;
   uint blockNum;
}
```

This is the structure that will hold any new data sent from the user on our application. We also need to define an id for each block to be used in the mapping. This can be done with the following line:

```
uint blockid;
```

Now, we need to map this contract to our blockchain. We do this as follows:

```
mapping(uint256 => NewBlock) public blockchain;
```

Next, we functions in this contract to take the block data and send it to the blockchain. We created the function called addData that takes the data from the user and adds it to the blockchain.

```
function addData (bytes32 data, uint8 uage, uint uid) public payable {

    //iterates key to indicate new struct
    blockid++;
    //"pointer" to new data structure
    var n = blockchain[blockid];

    //sets the data
    n.age = uage;
    n.id = uid;
    n.name = data;
      n.blockNum = block.number;
}
```

Notice that the function is of type “public payable.” This is due to the fact that nodes adding blocks to the blockchain will be paid in the form of gas known as Ether on our Ethereum private blockchain. This function takes the name, age, and ID entered by the user into our application, increases the number of block ids by 1, creates a new pointer which will hold the block data, and sets the block’s name, id, and age to the data entered by the user.

Next, we need to create getters that will obtain the name, id, age, and block number data from the block based on the key given by the function. These are defined as follows:

```
//getters using key passed in from function
function getID(uint256 key) public view returns (uint) {
    return blockchain[key].id;
}
```

```
function getAge(uint256 key) public view returns (uint) {
    return blockchain[key].age;
}
function getName(uint256 key) public view returns (bytes32) {
    return blockchain[key].name;
}
function getBlockNum(uint256 key) public view returns (uint) {
    return blockchain[key].blockNum;
}
```

Save this file and proceed with the Truffle configuration tutorial linked above.

Congratulations. You have created your first Solidity contract!

Configuring Your Cygwin Development Environment

For users on Windows 7 and 8.1, Cygwin is the best option for development in place of the Bash for Windows environment that is only enabled for Windows 10 users. To start development using Cygwin, you must first install nodejs.

1. Install nodejs
    a. Install the appropriate node.js version, then confirm that is installed by typing $node -v in your terminal.
    b. Note: Npm should be installed with nodejs. Check with npm -v in the console. Skip step 2 if version was installed.
2. Install npm via terminal if needed.
    a. $npm install --global.
    b. You can check your version of npm by typing
        $npm -v
3. Install git
    a. Follow install instructions and keep default selections. Once the installation is complete, restart your terminal and type:
        $git version
4. Install web3.js via terminal

a. Web3.js is the main ethereum javascript library that comes packed with useful API. Type in either of the two commands below to install.

```
$npm install web3
$npm install web3@^0.20.0
```

5. Install truffle via terminal
   a. $npm install -g truffle
   b. To check if you have truffle installed, type

      ```
      $truffle
      ```

      This will give you a list of the commands you can use in truffle.
6. Install ethereum testnet via terminal
   a. $npm install -g ethereumjs-testnet

These are the main tools you need to begin blockchain development. For developing your web application, we recommend using Visual Studio Code.

Application Development Using Truffle

Once you have everything installed and downloaded (node.js, npm, git, web3.js, truffle, and ethereum testnet), you are now ready to create your first blockchain project. First, open up your terminal and navigate to your desired workspace. If you use Cygwin, the terminal will open up in C:\cygwin64\home\UserName. To create a folder within that workspace, type

```
$mkdir SD_Blockchain
```

The name of my project in this instance is SD_Blockchain. To navigate into this workspace type

```
$cd SD_Blockchain
```

If you are familiar with Linux, then you will recognize many of these commands. That's because Cygwin is a terminal meant to emulate a Linux environment. Once you are in your workspace, you can check which files are in your terminal with the command

```
$ls
```

Once you are in your workspace, there should be no files present for the first time To begin your first contract you will use a truffle command to initialize the truffle environment.

```
$truffle init
```

This will initialize truffle within that workspace and create three folders and two Javascript files. The folders are [contracts] [migrations] [test] and the files are truffle.js and truffle_test.js.

[contracts] will be where you create all of your contracts. To create a .sol file within the contracts folder, navigate into the contracts folder.

```
$cd contracts
```

You will notice there is already a contract in there called migrations. This file is used when you migrate any new contracts after compilation. Now you want to create a new .sol file. To do so,

```
$touch FileName.sol
```

This will create a new file with filetype .sol.

Next you will have to add to the [migrations] folder. You will see there is already a javascript file in it called 1_initial_migration. You will want to create a new .js file

```
$touch 2_filename_migration.js
```

Installing Ethereum

Use the following commands:

```
sudo apt-get install -y software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install -y ethereum
```

Configuring an Ethereum Node

First, you need to install Geth.

Once geth is installed, you need to create a test account. You can use the following command to do so:

```
geth --datadir ~/mychain/data account new
```

Replace “/mychain/data” with your desired filepath. Next, the command terminal will prompt you with a passphrase. Enter the passphrase and enter it once more to confirm. The terminal will give you an address. Save the passphrase and the address as you will need it later. Next, you’ll be creating the genesis block.

Creating the Genesis Block

Use the following commands:
mkdir genesisblock - make the directory
cd genesisblock - navigate into the directory
nano genesis.json - make the file

Copy and paste the following genesis block into your file. This is our genesis block.:

```
{
   "config": {
      "chainId": 15,
      "homesteadBlock": 0,
      "eip155Block": 0,
      "eip158Block": 0
   },
   "difficulty": "0",
   "gasLimit": "2100000",
   "alloc": {
      "7df9a875a174b3bc565e6424a0050ebc1b2d1d82": { "balance": "300000" },
      "f41c74c9ae680c1aa78f42e5647a62f353b7bdde": { "balance": "400000" }
   }
}
```

Under the alloc section, replace the addresses we have with the address that you saved from the terminal. Save this new file and you have now created your first genesis block!

Next, you will initialize your local private node using the following command:

```
geth --identity "LocalTestNode" --rpc --rpcport 8080 --rpccorsdomain "*" --datadir ~/mychain/data/ --port 30303 --nodiscover --rpcapi db,eth,net,web3,personal --networkid 1999 --maxpeers 0 --verbosity 6 init ~/mychain/CustomGenesis.json 2>> ~/mychain/logs/00.log
```

Replace the file path "~/mychain/CustomGenesis.json" with the file path that contains the genesis file you just created. For more information about the parameters included in this command, please see this link.

To start the node in your Javascript (Truffle) console, enter the following command:

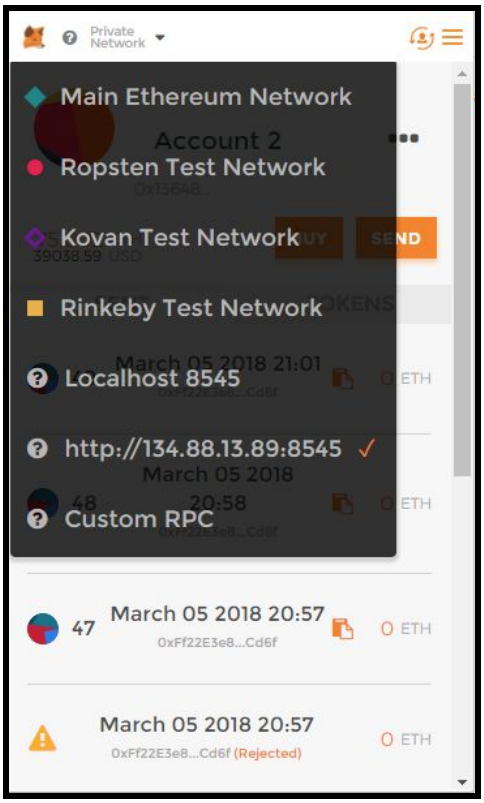
```
geth --identity "LocalTestNode" --rpc --rpcport 8080 --rpccorsdomain "*" --datadir ~/mychain/data/ --port 30303 --nodiscover --rpcapi db,eth,net,web3,personal --networkid 1999 --maxpeers 0 console
```

Now, you will be presented with the prompt ">" to enter commands. To check the balance of your account, enter eth.getBalance(eth.accounts[0]);

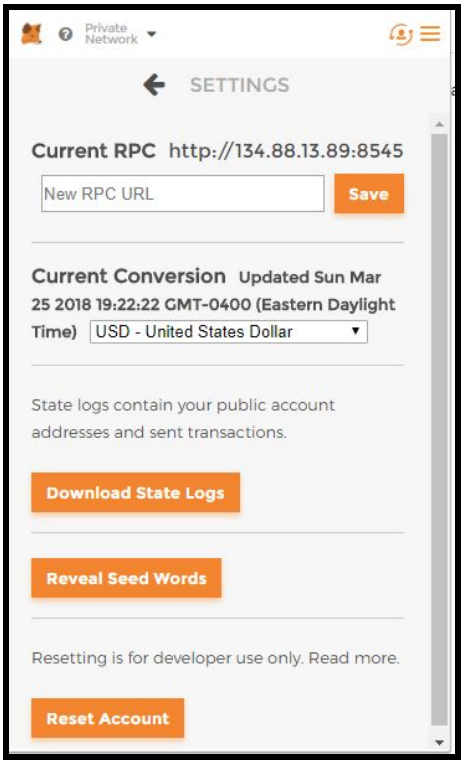
Congratulations! You have configured your first Ethereum private node!

Configuring Metamask

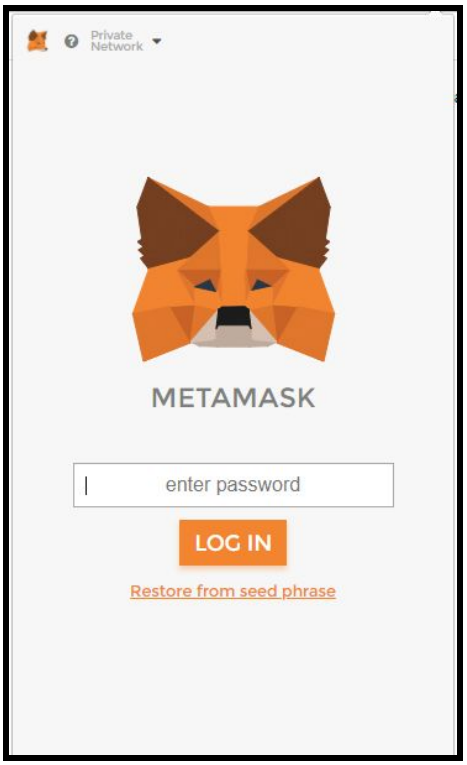
Select the Network Menu button in the upper left corner to display all of the networks that you can connect to. Currently, the options are the Main Ethereum network, Ropsten, Kovan, Rinkeby, your Localhost, or a custom RPC. Click the CustomRPC option to link to your private blockchain.



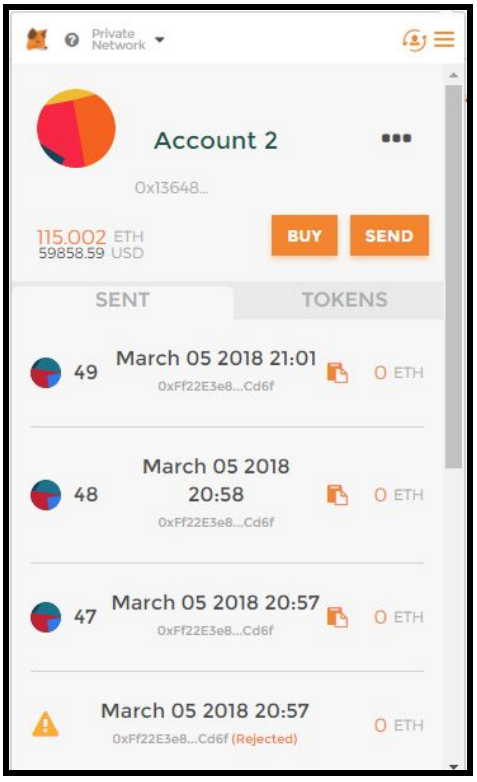
Enter in the URL of your application. Our URL is 134.88.13.89:8545.



The screen below will appear within Metamask. Enter the password that you used to configure your account to log in.



This is our account with 115.002 ether which is equivalent to 59858.59 USD.

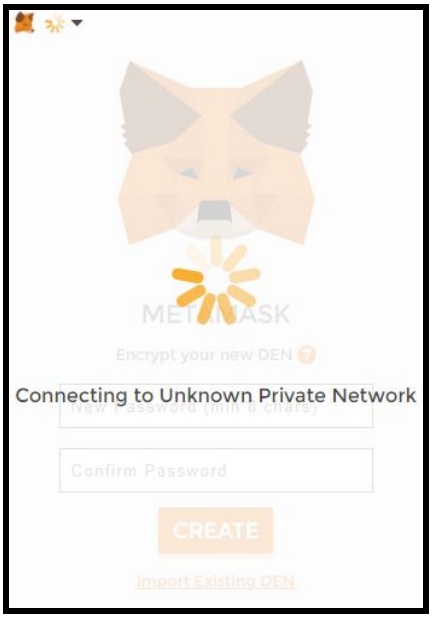


Congratulations. You have officially configured Metamask with your blockchain.

Chapter 11: Troubleshooting Issues with Development

Configuring MetaMask in Opera

When loading MetaMask in Opera, there were various times when the extension would get stuck in the loading screen.



To troubleshoot this issue, we did the following:

1. Exit the Opera browser and restart.
2. Disable and re-enable the MetaMask extension.
3. Try another browser such as Google Chrome or Mozilla Firefox.

Note: MetaMask is still in its early stages of development and it is not uncommon for it to stop working at times, especially in Opera as it is the most recent browser to acquire MetaMask support.

Appendices

Whitepapers

Blockchain

I. Democracy and the Blockchain

Hashgraph

I. Swirlds Hashgraph Consensus Algorithm
II. Hashgraph Detailed Examples
III. Hashgraph Overview
IV. Swirlds and Sybil Attacks
V. Hedera Hashgraph

Tangle

I. The Tangle

Hyperledger

I. Hyperledger Architecture
II. Hyperledger Sawtooth
III. Hyperledger Burrow

Corda

I. Corda: A Distributed Ledger

Resources for API/Language Documentation

Bootstrap Documentation
http://bootstrapdocs.com/v3.0.3/docs/css/

CSS Documentation
https://developer.mozilla.org/en-US/docs/Web/CSS

GETH Documentation
https://github.com/ethereum/go-ethereum/wiki/geth

HTML Documentation
https://developer.mozilla.org/en-US/docs/Web/HTML

Javascript Documentation
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference

JQuery Documentation
https://api.jquery.com

JSON Documentation
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON

Nodejs API Documentation
https://nodejs.org/api/index.html

Solidity Documentation
https://solidity.readthedocs.io/en/develop/

Web3js API Documentation
https://github.com/ethereum/wiki/wiki/JavaScript-API

Glossary

0-9

51% Attack

A 51% attack occurs when more than half of the computing power of a network is controlled by a single malicious entity/group. This entity or group may issue conflicting requests to harm the network.

A

Accounts

An account consists of a user ID and a password. In terms of the blockchain, our user ID is our public address and our password is our private key. Every transaction is validated with your private key.

Address

For IOTA, a public address is derived from the seed, security level, and key index. Addresses consist of 81 trits.

Address Checksum

In IOTA's Tangle, the address checksum is 81 trytes long and is used to construct a 90 tryte address to avoid address errors.

Agency

In Hyperledger Indy, this is a service provider that hosts Cloud Agents and may provision Edge Agents on behalf of a Ledger's Entities.

Agent

In Hyperledger Indy, this is a software program/process that is used by or acting on behalf of a

Ledger's Entity to interact with other agents. Edge Agents run at the edge of the network on a local device. Cloud Agents run remotely on a server or cloud hosting service.

AMQP

This is Corda's serialization mechanism used for everything except flow checkpoints and RPC.

Anchor Peer

In Hyperledger, an anchor peer is a peer node on a channel that all other peers on the network can communicate with and discover. Each member on the channel has an anchor peer. In some instances, members can have multiple anchor peers to help prevent against a single point of failure.

Ametsuchi

In Hyperledger Iroha, this is the storage component that stores blocks, indexes, and the latest ledger state.

Artemis

This is Corda's message queuing middleware.

Attachment

In Corda, an attachment is a piece of data that can be referred to within a transaction but is never marked as used.

B

Batch

In Hyperledger Sawtooth, this is a group of related transactions that is the atomic unit of state change for the blockchain. A batch can contain one or more transactions. If a transaction in a batch of more or one transactions fails, then all transactions in that batch will fail.

Blockchains

Blockchains are shared, trusted, and public ledgers of transactions. Everyone is able to see the transactions but no single user controls them. Blockchains are cryptographed, secure, and tamper resistant. They are distributed databases that are perfect for storing information such as values, identities, agreements, property rights, and credentials. Any information placed into a blockchain will remain there forever. Blockchains are decentralized, disintermediated, affordable, and censorship resistant. Various applications include Bitcoin, Namecoin, Sia, and Ethereum.

Blockchain Network

In Hyperledger, a blockchain network consists of a minimum of 1 peer that is responsible for endorsing and committing transactions. This peer uses an ordering service and a membership services component that distributes and revokes certificates representative of user identities and permissions.

Block

A block is a file that is used to permanently record data. It is a record of some or all of the most recent transactions that haven't been recorded in any previous blocks. New blocks are added at the end of the blockchain and can't be changed or removed once they are written. Each block memorializes what happened in the minutes before it was created and keeps a record of some or all recent transactions as well as a reference to its previous block.

Block Explorer

A block explorer is an online tool used for exploring the blockchain. It allows for watching and following all transactions in real time on the blockchain. They can serve as blockchain analysis and provide information such as total network hashrate and transaction growth.

Block Height

The number of blocks connected in the blockchain.

Bootstrap

In Hyperledger, this is the initial setup of a network.

Branch

In IOTA, a branch is also known as a trunk and refers to when two transactions that are referenced and validated by the transaction.

Bundles

In IOTA, a bundle consists of transactions that are grouped together during the creation of a transfer. Bundles are atomic transfers, therefore either all transactions in the bundle are accepted or none at all are accepted.

C

Chain

In Hyperledger, the chain is a transaction log structured as hash-linked blocks of transactions. Peers receive blocks of transactions from the ordering service, mark transactions as valid or invalid, and append the block to the hash chain on the peer's file system.

Chaincode

Chaincode is software that runs on a ledger and consists of transaction instructions for modifying assets.

Chain Linking

Chain linking is the process of connecting two blockchains. This allows for transactions between the two chains and also allows blockchains to communicate with other sidechains.

Channel

In Hyperledger, a channel is a private blockchain overlay that allows for data isolation and confidentiality. A channel-specific ledger is shared across all peers that are present on the channel. Transacting parties must be properly authenticated to a channel for interaction.

Client

A software program executed by a user on a desktop, laptop, or mobile device to launch an application.

Command

In Corda, commands are used for directing a transaction.

Commitment

In Hyperledger, each peer on the channel validates ordered blocks of transactions and commits them to its replica of the channel ledger. Peers mark each transaction in each block as valid or invalid.

Committer

In Hyperledger, this is a specific peer role where the peer appends the validated transactions to the channel-specific ledger.

Composite Key

In Corda, this is a tree data structure containing regular cryptographic public keys.

Concurrency Control Version Check

In Hyperledger, this is a method of keeping state in sync across peers on a channel. Peers execute their transactions in parallel. Before committing to the ledger, peers check that the data read at execution time hasn't changed. If the data has changed between execution time and commitment time, then a violation of the Concurrency Control Version Check has occurred, and the transaction is marked invalid on the ledger and values are not updated in the state database.

Configuration Block

In Hyperledger, a configuration block contains the configuration data that defines members and policies for a system chain or channel. Any configuration changes to a channel or network (e.g. a member leaving or joining) will result in a new configuration block being appended to the appropriate chain. This block contains the contents of the genesis block plus the delta.

Configuration System Chaincode

In Hyperledger Fabric, this is management system chaincode that handles configuration requests to change an aspect of the channel. This chaincode interrogates the channel's policies to determine if a new configuration block can be created.

Confirmed

In IOTA, a transaction is confirmed if it has been referenced by a milestone. Essentially, if the inclusion state of the transaction and the latest and the latest milestone as the tip is true, then the transaction is confirmed.

Consensus

A consensus requires that an agreement is made among a number of processes for a single data value.

Consortium Blockchains

A consortium blockchain is a blockchain where the consensus process is controlled by a preselected set of nodes. Each user operates a node and they must sign every block for the block to be valid. The right to read the blockchain may be public or restricted to its participants. Consortium blockchains are considered to be "partially decentralized."

Contract

In Corda, this is code that specifies how states can be created and used.

Corda

Corda is a distributed ledger for recording and maintaining financial agreements.

CorDapp

A CordApp is a distributed application that utilizes Corda. This consists of the following components: state objects, contract code, flows, any necessary APIs, wallet plugins, and UI components.

Cordformation

This is a gradle plugin used by Corda to deploy a set of local Corda nodes.

Cryptographic Hash Function

A cryptographic hash function is a math algorithm that takes an input that can be any kind of digital data (ex: password file) and produces a single fixed length output. The main properties of cryptographic hash functions are as follows: easy to compute a hash value for any message; infeasible to generate a message from its hash function except through brute force; infeasible to change a message's contents without changing the hash; infeasible to find two messages with the same hash; deterministic so that the same message always has the same hash output. Cryptographic hash functions may have security applications, indexing data in hash tables, fingerprinting, detecting duplicate data or uniquely identify files, and as checksums to detect accidental data corruption.

Curl

The main hash function that is used in IOTA's Tangle. It is based on the sponge construction invented by the creators of SHA-3 and is specifically adapted for IoT. It also happens to be the world's first trinary hash function.

Current State

In Hyperledger, the current state of the ledger represents the most recent values for all keys ever included in the chain transaction log. This is sometimes referred to as a World State.

D

dApp (Decentralized Application)

An application is decentralized if it meets the following criteria: completely open-source, operate autonomously, and with no entity controlling the majority of its tokens and all changes being decided on by a consensus of its users; data and operation records must be cryptographically stored in a public and decentralized blockchain to avoid central points of failure; must use a

cryptographic token necessary for access to the application; must generate tokens according to a standard cryptographic algorithm acting as a proof of valid contributing nodes.

Deploy

In Hyperledger Fabric, this refers to chaincode applications being deployed on the chain. It is first sent from the client SDK/CLI to a Lifecycle System Chaincode in the form of a proposal.

Depth

In IOTA, the depth is the starting point used for the random walk algorithm. The higher the depth is, the farther back in the tangle the random walk will begin and the longer the runtime will be. The typical depth value for IOTA is 3 - which starts the algorithm 3 milestones back.

Directed Acyclic Graph (DAG)

A DAG is a specific data structure that is based on a graph without any directed circles. On a DAG, there can be multiple branches instead of having a single branch with nodes having only one edge.

Distributed Consensus System

A Distributed Consensus System (DCS) is usually composed of replicated state machines that utilize a consensus protocol (Proof of work, gossip about gossip, proof of stake, etc). This consensus protocol is used to guarantee the validity of your digital data and also make it very difficult for threat agents to corrupt, steal, or change any of that data. A DCS can utilize a number of mechanisms to implement these consensus protocols, such as a blockchain, DAG, or hashgraph.

Distributed Network

A network where processing power and data are spread over the network nodes instead of having a centralized data center.

Digital Signature

A digital code generated by a public key encryption that is attached to an electronically

transmitted document to verify its contents and the sender's identity.

DSL

This is Corda's Domain Specific Language that is specifically designed for a domain.

Dynamic Membership

In Hyperledger Fabric, a dynamic membership consists of the addition and removal of peers, members, and ordering service nodes without compromising the functionality of the overall network.

E

End User

An end user is someone who would interact with a blockchain through a set of APIs.

Endorsement

In Hyperledger, endorsement is the process where specific peer nodes execute a chaincode transaction and return a response to the client application. This proposal response consists of chaincode response message, results, and events, as well as the signature to serve as proof of the peer's execution.

Endorsement Policy

In Hyperledger, the endorsement policy defines the peer nodes on a channel that need to execute transactions attached to a specific chaincode application, as well as the required combination of endorsements.

Endorsement System Chaincode

In Hyperledger Fabric, this is system chaincode that handles the endorsement policy for specific pieces of chaincode deployed on the network. This chaincode also defines the required parameters for a transaction proposal to receive a successful proposal response.

Endorser

In Hyperledger, the endorser is a specific peer role that is responsible for simulating transactions and prevents unstable/non-deterministic transactions from passing through the network. All endorsing peers are also committing peers (can write to the ledger).

Ethereum

Ethereum is an open software platform that uses blockchain technology. It enables developers to write smart contracts and build and deploy decentralized applications.

Ethereum Wallet

An Ethereum Wallet is a piece of software used to connect to an Ethereum blockchain.

F

Fairness

In Hashgraph, it should be difficult for a small group of attackers to unfairly influence the order of transactions that is chosen as the consensus.

Famous Witness

In Hashgraph, the community could put a list of n transactions into order by running separate Byzantine agreement protocols on O(n log n) different yes/no questions of the form “did event x come before event y?” A much faster approach is to pick just a few events (vertices in the hashgraph), to be called witnesses, and define a witness to be famous if the hashgraph shows that most members received it fairly soon after it was created. Then it’s sufficient to run the Byzantine agreement protocol only for witnesses, deciding for each witness the single question “is this witness famous?” Once Byzantine agreement is reached on the exact set of famous witnesses, it is easy to derive from the hashgraph a fair total order for all events.

Fork

Forks are created when two blocks are created at the same time. This essentially creates two

parallel blockchains with one of the two being the winning blockchain. The winning blockchain gets determined by its users by the majority choosing which blockchain the clients should listen to.

Flow

This is a set of instructions in Corda that determines how nodes communicate with the goal of consensus.

G

Genesis Block

The very first block in a blockchain.

Gossip

In Hashgraph, information spreads by each member repeatedly choosing another member at random, and telling them all they know.

Gossip about Gossip

In Hashgraph, the hashgraph is spread through the gossip protocol. The information being gossiped is the history of the gossip itself, so it is “gossip about gossip”. This uses very little bandwidth overhead beyond simply gossiping the transactions alone.

Gossip Protocol

In Hyperledger, a gossip protocol performs 3 functions: managing peer discovery and channel membership, disseminating ledger data across all channel peers, and synchronizing ledger states across all peers on the channel.

Gradle

An industry standard build and deployment tool used extensively within Corda.

H

Hash

Performing a hash function on the output data. This is used for confirming transactions.

Hardfork

A hardfork is a change to the protocol of the blockchain that makes previously invalid blocks or transactions valid. It requires all users to upgrade their clients.

Hashcash

Hashcash is a proof-of-work system that is used to limit email spam and DoS attacks.

Hashgraph

A data structure that records who gossiped to whom, and in what order.

Hijiri

In Hyperledger Iroha, this is the algorithm used for reputation recalculation and leader election.

Hyperledger Fabric CA

A CA is the default Certificate Authority component that issues PKI-based certificates to network member organizations and their users. The CA issues one root certificate per member and one enrollment certificate to each authorized user.

I

Inclusion State

In IOTA, the inclusion state is used to determine if a transaction was accepted or confirmed by the Tangle network. Given a transaction and a list of tips, the inclusion state is true if tip references the transaction.

Initialize

In Hyperledger, initializing is a method to initialize a chaincode application.

Inputs

In IOTA, an input is an address that is used to fund value transfers. The objects consist of the address, security level, and key index.

Install

In Hyperledger, an install is the process of placing a chaincode application on a peer's file system.

Instantiate

In Hyperledger, this is the process of starting and initializing a chaincode application on a specific channel. After instantiation, peers that have this chaincode installed can accept chaincode invocations.

Invoke

In Hyperledger, an invoke is used to call chaincode functions. This is done by sending a transaction proposal to a peer. The peer executes the chaincode and returns an endorsed proposal response to the client application which then gathers enough responses to satisfy the endorsement policy. The client application then submits the transaction results for ordering, validation, and commitment.

IOTA Reference Implementation (IRI)

IOTA's core client node that is written in Java. It communicates with IOTA's network to relay transactions and provide a limited API to users. This is done for security reasons and makes it possible to connect to a remote node.

J

JVM

Java Virtual Machine

K

Kerl

SHA-3 encryption adapted to trinary which uses additional conversion for input and output from/to 243 trits into 48 bytes using two's complement.

Key Index

In IOTA's Tangle, the key index is an integer that is used to specify which key to derive from the seed.

Kotlin

This is Corda's main language that is fully compatible with any JVM language.

L

Leading Peer

In Hyperledger, the leading peer is one of the peers belonging to the member that communicates with the network ordering service on behalf of the member.

Ledger

In Hyperledger, this is an append-only transaction log managed by peers. The ledger keeps the log of ordered transaction batches. The two denotations for a ledger are peer and validated. The peer ledger contains all batched transactions coming out of the ordering service. The validated ledger contains fully endorsed and validated transaction blocks.

Lifecycle System Chaincode

In Hyperledger Fabric, this is system chaincode that handles deployment, upgrade, and termination transactions for user chaincodes.

Light Node

A light node is a computer on the blockchain network that only verifies a limited number of transactions.

Lightning Network

A lightning network is a decentralized network that uses smart contract functionality on the blockchain to allow for instant payments across a network of participants. It allows transactions to happen instantly without worrying about block confirmation times. Lightning networks also allow two participants on the network to create an entry, conduct transactions between each other, and record the state of the transactions on the blockchain.

M

Member

In Hyperledger, a member is a legally separate entity that owns a unique root certificate for the network. Peer nodes and application clients can be linked to a member.

Membership Service Provider

In Hyperledger, this refers to an abstract system component that provides credentials to clients and peers in order for them to participate on the Fabric network. Credentials can be used for transaction and endorsement authentication.

Membership Services

In Hyperledger, these services authenticate, authorize, and manage identities on a permissioned blockchain network. The membership services code that runs in peers/orderers authenticate and authorize operations on a blockchain. This is a PKI-implementation of the Membership Service Provider abstraction.

Merkle Tree

A Merkle tree's main concept is to have some piece of data linking to another. This can be accomplished by linking things together via a cryptographic hash. The content itself can be used to determine the hash and this allows us to address the content. The content becomes immutable because if you change anything in the data, the hash changes and the link is different. Every block points to the previous block and a block becomes invalid if it is modified.

Milestones

In IOTA, milestones are checkpoint transactions that are issued every minute on average.

Miner

A miner is an Ethereum node that is used to validate a transaction.

Minimum Weight Magnitude (MWM)

In IOTA, the MWM is the amount of work for Proof of Work that will be carried out. Each increment of MWM is 3x harder PoW on average.

Mist Wallet

Mist Wallet is the software used to connected to the Blockchain network using its UI page. This software can also be used to connect with multiple blockchain networks. Mist Wallet lets you create accounts and transactions. It runs indirectly on top of the Ethereum Client software.

Multi-Channel

In Hyperledger, Fabric allows for multiple channels with a designated ledger per channel. This allows for multilateral contracts where only restricted participants on the channel can submit, endorse, order, or commit transactions on that specific channel. A single peer can maintain multiple ledgers without compromising privacy and confidentiality.

MultiSig Address

IOTA's Tangle uses a MultiSig address which is an address derived by absorbing multiple signature keys. This type of address would need multiple parties to sign the transaction to move funds.

N

Network Map Service

A network service utilized by Corda that maintains a map of node names and their network locations. This is used by nodes so they can communicate directly with other parties.

Node

A node is any computer that is connected to the blockchain network. Nodes are considered to be full nodes if they fully enforce all rules of the blockchain. Most nodes are lightweight nodes but full nodes form the network backbone.

Notary Service

A network service used by Corda that guarantees that it will only add its signature to transactions if all input states have not been consumed.

O

Oracles

Smart contracts on the blockchain that cannot access the outside network on their own. Oracles sit between a smart contract and the external world. They provide data needed by the smart contract and send its commands to external systems.

Orderer

In Hyperledger, the orderer is one of the network entities that comprise the ordering service.

Ordering Service

In Hyperledger, this is a defined collective of nodes that can order transactions into a block. This exists independent of the peer processes and order transactions on a first-come first-serve basis for all network channels. The ordering service is a common binding for the whole network and

contains the cryptographic identity material tied to each member.

Output

In Corda, an output is a state generated from the transaction that are then used as inputs for subsequent transactions.

P

Peer

In Hyperledger, a network entity that maintains a ledger and runs chaincode containers in order to perform read/write operations. Peers are owned and maintained by members.

Peer-to-Peer Network

The decentralized interactions between two parties or more in a highly-interconnected network. Participants of a P2P network deal directly with each other through a single mediation point.

Pending

In IOTA, a transaction is pending if it has been seen on the network but not yet confirmed.

Permissioned Network

A permissioned network is a blockchain network where a node is required to maintain a member identity on the network. End users must be authorized and authenticated in order for network use.

Policy

In Hyperledger, a policy may be for endorsement, validation, management of chaincode or the network/channel.

Public Address

The cryptographic hash of a public key. These can act as email addresses that can be published

anywhere unlike private keys.

Private Keys

String of data that allows you to access the tokens. These act as passwords that are kept hidden from anyone but the owner of the address.

Private Blockchains

Private blockchains are blockchains where write permissions are centralized to a single organization. Read permissions can be public or restricted.

Proof of Authority (PoA)

A proof of authority is a method of consensus in a private blockchain. It gives one or more clients with one particular private key the right to make all blocks on a blockchain.

Proof of Work (PoW)

A proof of work system is a measure used to deter denial of service attacks and other service abuses such as spam. PoW systems require some work from the service requester. Typical work may include processing time by a computer.

Proposal

In Hyperledger, a proposal is a request for endorsement that is directed at specific peers on the channel. Each proposal is either an instantiate or an invoke request.

Public Blockchains

A public blockchain is a blockchain that allows for anyone to read it, send transactions, and participate in the consensus process. Public blockchains are secured by crypto economics which is the combination of economic incentives and cryptographic verification by using proof of work or proof of stake. Public blockchains are considered to be fully decentralized.

Q

Query

In Hyperledger, this is a chaincode invocation that reads the ledger's current state but doesn't write to the ledger.

R

Random Walk

In IOTA, Random Walk is an algorithm used to select the pair of previous transactions used in tip selection.

Reattach

In IOTA, a reattach takes a bundle's signature and transfer information and reattaches it to the Tangle structure, selecting new tips and performing proof of work.

Rebroadcast

In IOTA, a rebroadcast is the process of resending the raw trytes of a transaction if one transaction is missing from a bundle. Rebroadcasting can help with confirmation or if the bundle doesn't appear on an explorer.

Remainder Address

In IOTA, the remainder address is the address used to send the remainder of a transfer value if one exists.

Ring Signature

Ring signatures are cryptographic and provide a decent level of anonymisation on a blockchain. They ensure that individual transaction outputs cannot be traced. A message signed with a ring signature is endorsed by someone. It should be computationally infeasible to determine which group member key was used to produce the ring signature.

S

Security Level

In IOTA's Tangle, you have the choice of selecting between 3 security models: 1 (81-trit security, low), 2 (162-trit security, medium), 3 (243-trit security, high). The security level affects the length of the transaction bundles where each spending address signature is spread on 1, 2, or 3 transactions.

Seed

In IOTA's Tangle, the seed is your master private key. The seed is used to derive all private and public keys. It must be kept secure and not shared with anyone. Seeds consist of 81 trytes in IOTA which is equivalent to 384-bit security. A seed is analogous to a private key or a password and can be used to access your account.

SHA (Secure Hash Algorithm)

A SHA is a group of cryptographic hash functions published by the National Institute of Standards and Technology. It takes an input of any size and form, mixes it up, and creates a fixed size output known as a hash. A hash can be viewed as a fingerprint of the data. They are one-way functions and cannot be decrypted back to their original form unless brute force is used.

Smart Contracts

Smart Contracts are computer protocols that facilitate, verify, or enforce the negotiation or performance of a contract. They usually have a user interface and emulate the logic of contractual clauses. They aim to provide security superior to traditional contract law.

Softfork

A soft-fork is a change to the protocol where only previously valid blocks are made invalid. It is backward-compatible since old nodes will recognize new blocks as valid. This type of fork requires only a majority of miners to upgrade and enforce the new rules.

Software Development Kit (SDK)

A structured environment of libraries for developers to write and test applications.

Solidity

A programming language used for developing Smart Contracts.

SVP (Simplified Payment Verification) Client

SVP Clients are lightweight clients that don't download and locally store the entire blockchain. They provide a way to verify transactions without having to download the whole blockchain. They only download the block headers by connecting to a full node.

State Channel

A state channel is an interaction that is made off of the blockchain without significantly increasing participant risk. This is done by moving interactions off of the chain which can lead to improvements in cost and speed. They function by locking part of the blockchain state so a set of participants can completely agree with each other to update it.

State Database

In Hyperledger, current state data is stored in a state database for efficient reading and queries from the chaincode.

Stateful Validation

In Hyperledger Iroha, this validation performs checks of schema validity and signatures of the transaction.

Stateless Validation

In Hyperledger Iroha, this validation checks the permissions and possibility of the action as well as any business rules.

Strongly Seeing

Given any two vertices x and y in the hashgraph, it can be immediately calculated whether x can strongly see y, which is defined to be true if they are connected by multiple directed paths passing through enough members. This concept allows the key lemma to be proved: that if Alice and Bob are both able to calculate Carol's virtual vote on a given question, then Alice and Bob get the same answer. That lemma forms the foundation for the rest of the mathematical proof of Byzantine agreement with probability one.

Sumeragi
In Hyperledger Iroha, this is the reputation-based consensus algorithm.

Swarm
A Swarm is a distributed storage platform and content distribution service. Its primary objective is to provide a decentralized and redundant store of Ethereum's public record. Essentially, it stores and distributes dApp code and data as well as blockchain data.

System Chain
In Hyperledger, the system chain contains a configuration block that defines the network at a system level. The system chain exists within the ordering service and has an initial configuration containing important information. Any change to the network will result in a new configuration block being added to this system chain.

System Chaincode
In Hyperledger, this is chaincode built with the peer and runs in the same process as the peer. It is responsible for broader configurations of Hyperledger Fabric behavior, such as timing and naming services.

T

Tag
In IOTA, the tag is a short message that can be attached to a transfer. It is up to 27 trytes in length

and is searchable.

Tangle

A DAG as a distributed ledger that stores all transactional data of the IOTA network. Essentially, a Tangle is a blockchain without block structures and the chain. Tangl is the first distributed ledger to have achieved scalability, no transaction fees, data integrity and transmission as well as quantum-computing protection.

Tail Transaction

In IOTA, a tail transaction is the last transaction added to a bundle which identifies the instance of the bundle. A tail can be constructed for a bundle and validated by traversing the trunk of each transaction.

Token

A token is a digital identity for something that can be owned and is created as a sophisticated smart contract system with permission systems and interaction paths attached to them.

Tori

In Hyperledger Iroha, this is the data transmission service that is the entrypoint for transactions and pipelining of queries.

Testnet

A testnet is a second blockchain used by developers to test new versions of client software. This is done to prevent putting data at risk.

Tips

Transactions that have no other transactions referencing them.

Tip Selection

In order to broadcast a new transaction in the Tangle, you need to validate the two previous transactions. This confirmation occurs by validating the trytes of the transaction, signatures, and

cross-checking for any conflicts in transactions.

Transaction (IOTA's Tangle)

A transaction in Tangle is the smallest unit of data and consists of 2673 trytes. It can be used to either transfer value or data on the tangle.

Transaction (Hyperledger)

In Hyperledger, this consists of invoke/instantiate results that are submitted for ordering, validation, and commit.

Transaction (Hashgraph)

In Hashgraph, any member can create a signed transaction at any time. All members get a copy of it, and the community reaches Byzantine agreement on the order of those transactions.

Transaction Block

Collection of transactions gathered into a block that can then be hashed and added into the blockchain.

Transfers

In IOTA, transfers are sending value or data to an address and are a higher level abstraction of bundles and transactions.

Transfer Objects

In IOTA, a transfer object is the recipient of the transfer that contains the destination address and the value.

Trinary

Trinary is the alternative to Binary and uses a base-3 numeral system. A trit is analogous to a bit, and a ternary digit is a trinary digit. The digits can have values of 0, 1, or -1. A tryte is analogous to a byte, and consists of 3 trits which represent 27 values.

U

User Keys (Public and Private)

Public Key: A User's address on the blockchain. (A long string of randomly generated numbers)

Private Key: Gives access to a User's digital assets.

Having a public and private key is what allows users to share digital assets through the blockchain safe and securely. You must safeguard and protect your private key however; and if you lose your private key you will be unable to retrieve your assets.

V

Validation System Chaincode

In Hyperledger Fabric, this is chaincode that handles the validation policy for specific chaincode sections deployed on the network. This chaincode is defined at the time of the deployment transaction proposal and validates the specific level of endorsement in order to prevent malicious or faulty behavior from the client.

Validator Node

In Hyperledger Indy, this is a node that validates new transactions of Identity REcords and writes valdi transactions to an Indy-Powered Ledger using the Plenum Consensus Protocol.

Verifiable Claim

In Hyperledger Indy, this is a claim that includes proof from the issuer. This proof is typically in the form of a digital signature and can be verified by a public key associated with the issuer's DID.

Virtual Voting

In Hashgraph, very member has a copy of the hashgraph, so Alice can calculate what vote Bob would have sent her, if they had been running a traditional Byzantine agreement protocol that involved sending votes. So Bob doesn't need to actually her the vote. Every member can reach

Byzantine agreement on any number of decisions, without a single vote ever being sent. The hashgraph alone is sufficient. So zero bandwidth is used, beyond simply gossiping the hashgraph.

W

Wallet (Blockchain)
In Blockchain, this is a file that houses private keys and usually contains a software client that allows access to view and create transactions on a specific blockchain.

Wallet (Hyperledger Indy)
In Hyperledger Indy, this is a software module for securely storing and accessing Private Keys, Master Secrets, and other sensitive material. A wallet can either be an Edge Wallet or a Cloud Wallet.

Web3.js
Web3 is a channel that is used to run ethereum commands from the front end or back end to interact with the blockchain.

Whisper
A whisper is a part of the Ethereum protocol that allows for messaging between users on the same network that runs the blockchain. Its main task is the provision of a communication protocol between decentralized applications.

Winternitz One-Time Signature (W-OTS)
A Post Quantum Signature Scheme used to authorize spending from an address in IOTA. Since it has a one-time nature, the security of funds belonging to an address decreases rapidly if you use the same key to sign multiple transactions. Therefore, that is why in IOTA you don't reuse an address that has been spent from.

Z

Zero Knowledge Proof

In Hyperledger Indy, this is a proof that uses special cryptography and a Master Secret to allow selective disclosure of information in a set of Claims. This proves that some or all of the data in a set of Claims is true without revealing any other information.

Sources: [71], [72], [73], [74], [75], [76], [77], [78].

References

[1]"What is Blockchain Technology? A Step-by-Step Guide For Beginners", Blockgeeks, 2017.

[2] E. Piscini, D. Dalton and L. Kehoe, "Blockchains and Cyber Security", Deloitte, 2017.

[3] "Know more about Blockchain: Overview, Technology, Application Areas and Use Cases - Lets Talk Payments", Lets Talk Payments, 2017.

[4]"4 Key Features of Blockchain", Techracers, 2017.

[5]M. Gupta, Blockchain for Dummies. for Dummies: A Wiley Brand, 2017.

[6] "Ethereum Project", Ethereum.org, 2017

[7] A. Gonsalves, "Cisco says blockchain ledger technology has networking role", SearchSDN, 2017.

[8] K. Leary, "Illinois is experimenting with blockchains to replace physical birth certificates", Futurism, 2017.

[9] N. Shimizu, "Blockchain's new client: Connected cars- Nikkei Asian Review", Nikkei Asian Review, 2017.

[10]M. Orcutt, "Why the CDC thinks blockchain can save lives", MIT Technology Review, 2017.

[11]"Using Blockchain Technology to Boost Cyber Security", Hacker Noon, 2017.

[12]A. Meola, "The growing list of applications and use cases of blockchain technology in business & life", Business Insider, 2017.

[13]"Blockchain Glossary." Blockchainhub, 16 October 2017.

[14]“Comprehensive Blockchain Glossary: From A-Z - Blockgeeks." Blockgeeks. 16 October 2017.

[15]

[16]"What are Blockchain's Issues and Limitations? - CoinDesk", CoinDesk, 2017.

[17] Antonopoulos, Andreas M. (2017-06-12). Mastering Bitcoin: Programming the Open Blockchain. O'Reilly Media. Kindle Edition.

[18] Church, Zach. “Newsroom.” Blockchain, Explained, 25 May 2017.

[19] Laurence, Paul. Blockchain: Step-By- Step Guide to Understanding and Implementing Blockchain Technology. Kindle Edition.

[20] P. Sandner, "Comparison of Ethereum, Hyperledger Fabric and Corda", Medium, 2017.

[21] S. Gr, "Ethereum Blockchain — concepts and terminologies – Seetharaman Gr – Medium",

Medium, 2017.

[22]I. Lin and T. Liao, "International Journal of Modern Trends in Engineering & Research", 2017.

[23] H. Kancharana, "Blockchain 101 - A Beginner's Guide", Zinnov, 2017.

[24] "Blockchains & Distributed Ledger Technologies", BlockchainHub, 2017.

[25] "Homepage | Hedera Hashgraph", Hedera Hashgraph, 2018.

[26] L. Baird, "Hashgraph Consensus: Detailed Examples", Swirlds.com, 2016.

[27] L. Baird, "The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance", Swirlds.com, 2016.

[28] J. Danneman, "Blockchain Just Became Obsolete. The Future is Hashgraph.", Squawker.org, 2017.

[29] L. Baird, "Overview of Swirlds Hashgraph", Swirlds.com, 2016.

[30] S. Popov, "The Tangle", Iota.org, 2017.

[31] D. Schiener, "Current role of the coordinator · IOTA Guide", Domschiener.gitbooks.io, 2018.

[32] G. Konstantopolous, "Understanding Blockchain Fundamentals, Part 1: Byzantine Fault Tolerance", Medium, 2017.

[33] "New to /r/Hashgraph? Please read this post first!", reddit, 2018.

[34] "DAG vs the Blockchain", BTCMANAGER, 2018.

[35] C. Tozzi, "Byzantine Fault Tolerance: The Key for Blockchains", NASDAQ.com, 2017.

[36] "Go Ethereum", GitHub, 2015.

[37] "The network — R3 Corda V3.0 documentation", Docs.corda.net, 2018.

[38] "Corda: The Introduction", Docs.corda.net, 2018.

[39] "Web 3.0: The Third Generation Web is Coming", Lifeboat.com, 2018.

[40] "Web 3: A platform for decentralized apps — Ethereum Homestead 0.1 documentation", Ethdocs.org, 2018.

[41] J. DeMuro, "Here are the 10 sectors that blockchain will disrupt forever", TechRadar, 2018.

[42] P. Loop, "5 Blockchain Developments Coming in 2018 - CoinDesk", CoinDesk, 2018.

[43] B. Marr, "35 Amazing Real World Examples Of How Blockchain Is Changing Our World", Forbes.com, 2018

[44] S. Pongnumkul, C. Siripanpornchana and S. Thajchayapong, "Performance Analysis of Private Blockchain Platforms in Varying Workloads - IEEE Conference Publication", Ieeexplore.ieee.org, 2017.

[45] R. Nagpal, "17 blockchain platforms — a brief introduction – Blockchain Blog – Medium", Medium, 2017.

[46] M. Macdoland, L. Liu-Thorrold and R. Julien, "The Blockchain: A Comparison of Platforms and Their Uses Beyond Bitcoin", The University of Queensland, 2017.

[47] "imbaniac/awesome-blockchain", GitHub, 2018.

[48] B. Nielson, "Review of the 6 Major Blockchain Protocols - RICHTOPIA", RICHTOPIA, 2018.

[49] J. Bharwani, "Top 8 Blockchain Platforms You Should Check Out Now - B2B News Network", B2B News Network, 2017.

[50] "What are the Ethereum disk space needs?", Ethereum.stackexchange.com, 2016.

[51] "Sharding FAQ", GitHub, 2018.

[52]"How can IOTA's proof of work provide sufficient network security?", Iota.stackexchange.com, 2018.

[53] "What is "Snapshotting" in IOTA?", Steemit.com, 2017.

[54] M. Wind, "Everything you didn't know about IOTA – Misty Wind – Medium", Medium, 2017.

[55] "What is IOTA - IOTA Docs", Docs.iota.org, 2018.

[56]"Hyperledger Architecture: Introduction to Hyperledger Business Blockchain Design Philosophy and Consensus", Hyperledger.org, 2017.

[57] "Introduction — Sawtooth v1.0.1 documentation", Sawtooth.hyperledger.org, 2018.

[58] "Settings Transaction Family Specification — Sawtooth v1.0.1 documentation", Sawtooth.hyperledger.org, 2018.

[59] "Information and FAQ", reddit, 2018.

[60] "Welcome to Corda ! — R3 Corda V3.0 documentation", Docs.corda.net, 2018.

[61] "Corda: Frictionless Commerce", corda.net, 2018.

[62] "R3", R3.com, 2018.

[63] T. Kersten, "R3 Corda isn't a Blockchain. What Does This Mean for You?", Medium, 2016.

[64] "Hyperledger Welcomes Indy", Hyperledger, 2017.

[65] "Hyperledger Welcomes Iroha", Hyperledger, 2016.

[66] "Hyperledger Iroha Whitepaper", GitHub, 2017.

[67] "Hyperledger Burrow v0.16", GitHub, 2017.

[68] "Welcome to Hyperledger Fabric — hyperledger-fabricdocs master documentation",

Hyperledger-fabric.readthedocs.io, 2018.

[69] "Hyperledger Projects", Hyperledger, 2018.

[70] "corda.net", corda.net, 2018.

[71] "IOTA Tangle Glossary", Iota.readme.io, 2018.

[72] "Glossary for Hyperledger Fabric", Hyperledger-fabric.readthedocs.io, 2018.

[73] "Hyperledger Fabric Glossary", Fabrictestdocs.readthedocs.io, 2017.

[74] "Hyperledger Iroha Glossary", GitHub, 2017.

[75] "Glossary — Sawtooth latest documentation", Sawtooth.hyperledger.org, 2017.

[76] T. Kuhrt, "Hyperledger Indy Glossary", Wiki.hyperledger.org, 2018.

[77] "Glossary — R3 Corda V3.0 documentation", Docs.corda.net, 2018.

[78] P. Whipp, "Glossary — monsuite 0.0.1 documentation", Monsuite.readthedocs.io, 2018.

[79] A. Tar, "Decentralized and Distributed Databases, Explained", Cointelegraph.com, 2017.

[80] "Web3 - The Decentralized Web", BlockchainHub, 2018.

[81] M. VerHill, "The Significance of Decentralization", Ethfinex, 2017.

[82] "Blockchain for 2018 and Beyond: A (growing) list of blockchain use cases", Medium, 2018.

[83] M. Redka, "Market Alternatives for the Ethereum Blockchain", Mlsdev.com, 2017.

[84] J. Bharwani, "Top 8 Blockchain Platforms You Should Check Out Now - B2B News Network", B2B News Network, 2017.

[85] J. Maurice, "Top 10 Best Blockchain Platforms for ICOs in 2018 - DisruptorDaily", DisruptorDaily, 2018.

[86] J. Bushmaker, "The Top 50 Cryptocurrencies - Invest In Blockchain", Invest In Blockchain, 2018.

[87] B. Marr, "35 Amazing Real World Examples Of How Blockchain Is Changing Our World", Forbes.com, 2018.

[88] O. Krauth, "5 companies using blockchain to drive their supply chain", TechRepublic, 2018.

[89] "Monax", Monax, 2018.

[90] "Multichain Review - Open Source Private Blockchain Application Creation?", Bitcoinexchangeguide.com, 2018.

[91] S. Ray, "Merkle Trees – Hacker Noon", Hacker Noon, 2017.

[92] "ethereum/wiki", GitHub, 2018.

[93] V. Buterin, "Merkling in Ethereum - Ethereum Blog", Ethereum Blog, 2015.

[94] J. Hu, "IOTA Tangle: Introductory overview of white paper for Beginners", Medium, 2017.

[95] "What is Iota? – Hacker Noon", Hacker Noon, 2017.

[96] "ELI5 How does a Merkle-Patricia-trie tree work?", Ethereum.stackexchange.com, 2018.

[97] A. Castor, "A (Short) Guide to Blockchain Consensus Protocols - CoinDesk", CoinDesk, 2017.

[98] A. Rosic, "What Is Hashing? Under The Hood Of Blockchain - Blockgeeks", Blockgeeks, 2017.

[99] "If you understand Hash Functions, you'll understand Blockchains", Decentralize Today, 2016.

[100] I. Simpson, "To Understand Blockchains, You Should Understand Cryptographic Hashes First", Medium, 2017.

[101] B. Team, "What is a Merkle Tree and How Does It Help Organize Data On The Bitcoin Blockchain? - Bitcoin.com.au", Bitcoin.com.au, 2017.

[102] "Proof of Authority: consensus model with Identity at Stake.", Medium, 2017.

[103] "Blockchain Networks: Possible Attacks and Ways of Protection", InfoSec Resources, 2018.

[104] "What are Blockchain's Issues and Limitations? - CoinDesk", CoinDesk, 2018.

[105] "Blockchain Advantage and Disadvantages – nudjed – Medium", Medium, 2018.