Cryptocurrencies and Blockchain

Kambombo Mtonga

June 24, 2024

Outline

- Introduction to Blockchain Technology
- Cryptography in Blockchain
  - Cryptographic hash functions
  - Digital signatures
- Consensus Algorithms
- Wallets

Blockchain Technology

What is a Blockchain

A blockchain is an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way without the need for a central authority.

Remark

A ledger is basically a file that constantly grows and keeps the record of all transactions permanently.

Key characteristics of a Blockchain

- **Open:** Anyone can access blockchain.
- **Distributed or Decentralised:** Not under the control of any single authority.
- **Efficient:** Fast and Scalable.
- **Verifiable:** Everyone can check the validity of information since each node maintains a copy of the transactions.
- **Permanent:** Transactions done are persistent and unalterable.

Types of Blockchains

Three broad types exists:

- **Public Blockchain:** A permissionless blockchain that can be accessed by anyone by connecting to the network, e.g., Bitcoin.
- **Private Blockchain:** Closed networks open exclusively to authorized users. Can be used by companies for managing internal information and sensitive data.
- **Consortium Blockchain:** This is a network controlled by a group of entities or organizations. It is more decentralized than a private blockchain and can be used by organizations with common goals to ensure transparency between the participants.

Distributed systems

- A Blockchain is a decentralized-distributed system, i.e., it is a distributed ledger that can be centralized or decentralized.

Definition (Distributed system)

Distributed systems are a computing paradigm whereby two or more nodes work with each other in a coordinated fashion to achieve a common outcome. However, the system is modeled in such a way that end users see it as a single logical platform, e.g., Google Search Engine.

Definition (Nodes)

A node is a computer (or device) connected to the Blockchain Network. The connection is via a client that helps in validating and propagates transaction on to the Blockchain.

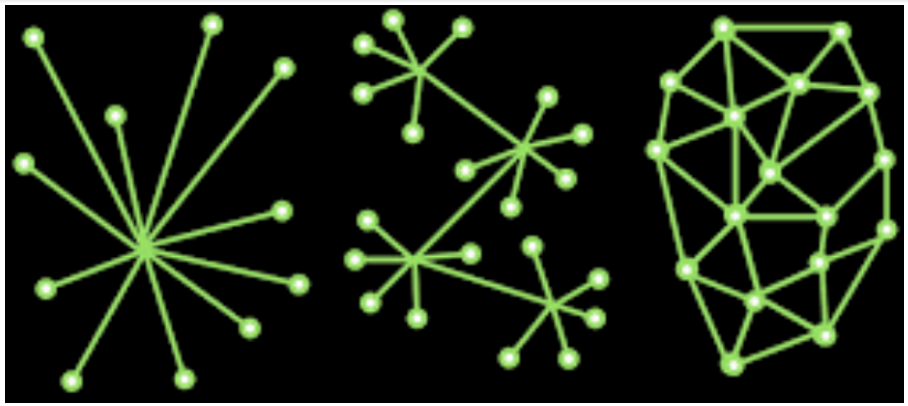
- When a node connects to the Blockchain, a copy of the Blockchain data gets downloaded into the system and the node comes in sync with the latest block of data on Blockchain.

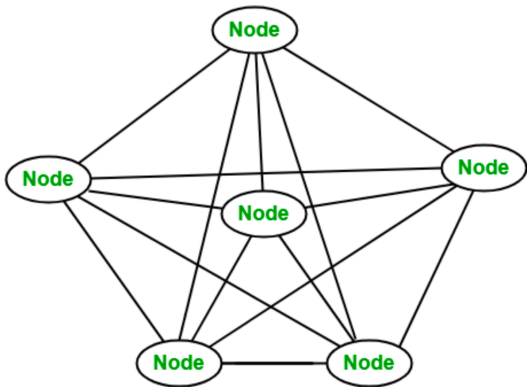
Distributed system (2)

- Nodes can be **honest**, **faulty**, or **malicious** and have their own memory and processor. A node that can exhibit arbitrary behavior is also known as a *Byzantine node*.

Definition (Miners)

The Node connected to the Blockchain which helps in the execution of a Transaction (or Tx) in return for an incentive.





Scenarios - To Motivate - Bank Frauds

- You find a check was used to pay someone but you never wrote the check
  - Someone forged your check and/or signature
- You did sign a check for x amount, but the amount field was modified
  - How do you prove to the bank that an extra 0 was not there in your signing time?
- The monthly statement says that you did a transaction but you did not recall or the amount of a transaction is different from what you had done
  - Someone got your password, and possibly redirected OTP to another SIM (SIM Fraud)
  - Bank employees themselves might have done something
- How do you argue to the bank? (Non-repudiation)
- How do you argue that the amount was modified? (Integrity)
- Finally, do you tally your transactions when you receive your monthly statement?
  - Most people do not

Scenarios - To Motivate - Supply chain provenance

- You buy ice cream for your restaurant from supplier B
- Supplier B actually transports ice cream made in Company C's factory
- Upon delivery, you have been finding that your ice cream is already melted
- Who is responsible?
  - Supplier B is keeping it too long on the delivery truck?
  - Supplier B's storage facility has a temperature problem?
  - Supplier C says it' s supplier B's fault as when picked up - ice cream was frozen
  - Supplier B says that when received, the temperature was too high, so C must have stored it or made it wrong
- How do you find the truth?
  - Put temperature sensors in B's truck and storage, C's factory and storage, and sensor data is digitally signed by the entity where the sensor is placed and put in a log
  - You check the log - but B and C both have hacked the log and deleted some entries? What to do?

Scenarios - To Motivate - Real estate

- You buy a piece of land
- Someone else claims to own the land
- But the one who sold you the land showed you paper work
- Land registry office earlier said that the owner was rightful
- Now they say that they made a mistake -it was owned by the other person
- You already paid for the land -to the first person
- First person goes missing
- How does any one prove who changed the land record?
- The government employees?

The need for Blockchain

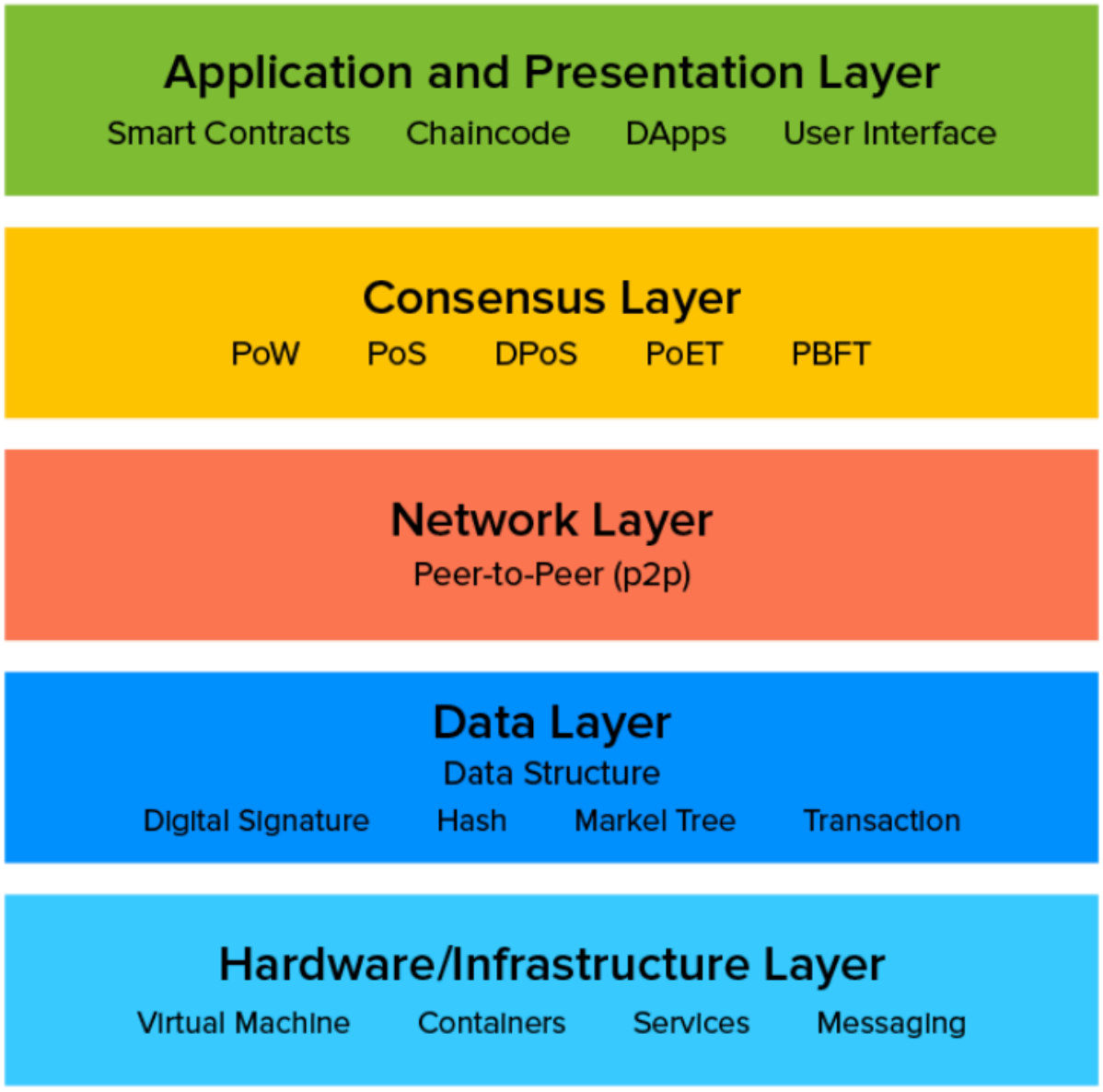
Disadvantages of traditional transaction system

1. Cash can only be used in low amount transaction locally.
2. Huge waiting time in the processing of transactions.
3. Need for third party for verification and execution of Transaction makes the process complex.
4. If the Central Server like Banks is compromised, whole System is affected including the participants.
5. Organization doing validation charge high process thus making the process expensive.

Building Trust with Blockchain

- When operating on a Blockchain network, trust is not necessary. Blockchain builds trust through the following five attributes:
  1. **Distributed:** The distributed ledger is shared and updated in real time with every incoming transaction among the nodes connected to the Blockchain.
  2. **Secure:** There is no unauthorized access to Blockchain - made possible through Permissions and Cryptography.
  3. **Transparent:** Every node in a Blockchain has a copy of the Blockchain data, it implies each node can verify the identities without the need for mediators.
  4. **Consensus-based:** All relevant network participants must agree that a transaction is valid. This is achieved through the use of Consensus algorithms.
  5. **Flexible:** Smart Contracts which are executed based on certain conditions can be written into the platform. Blockchain Network can evolve in pace with business processes.

The Blockchain architecture



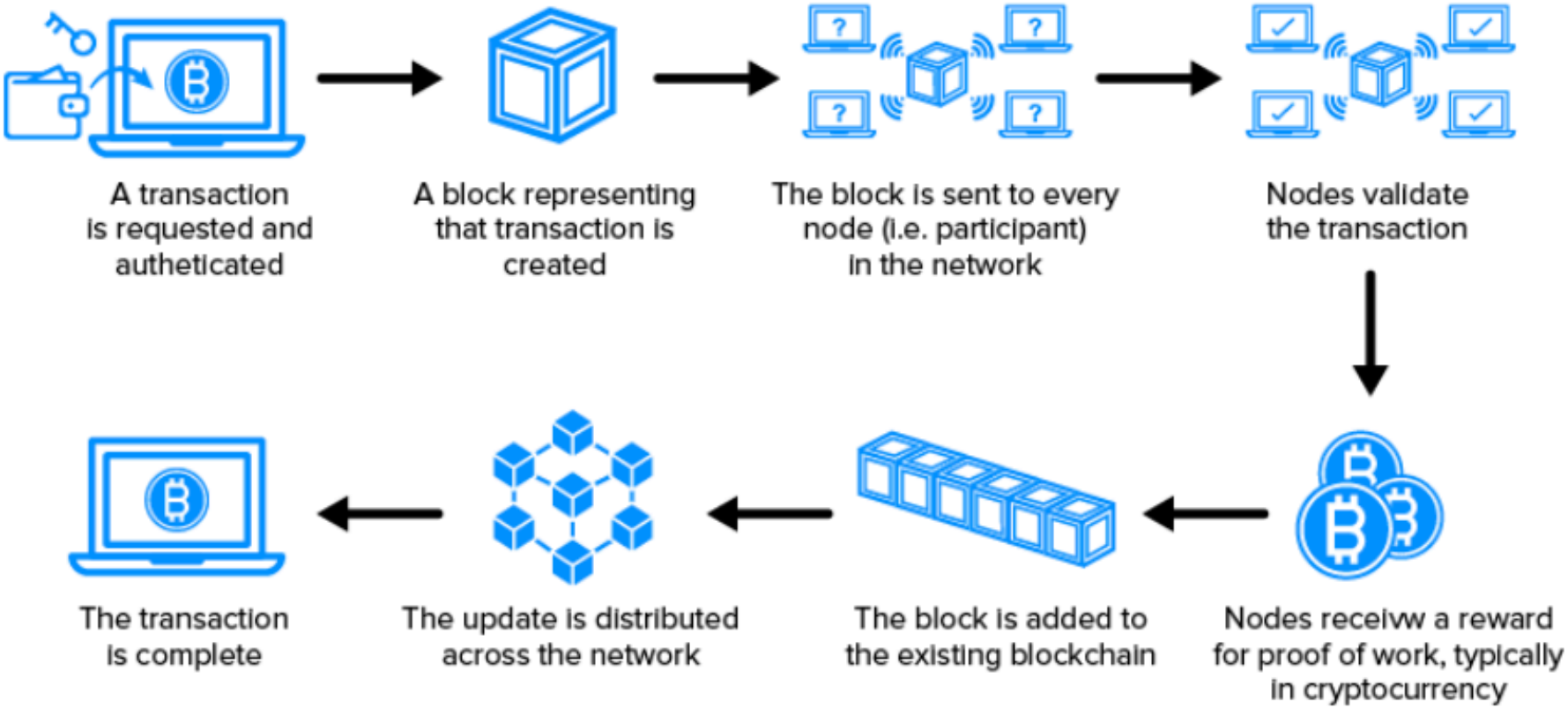
The Blockchain architecture (2)

- **Consensus layer:** Guarantees that all nodes in the network concur on the validity of each transaction. Consensus mechanisms, e.g., Proof of Work (PoW). Only validated transactions are added to the blockchain.
- **Application layer:** Apps are built on this layer. Examples of these implementations include, wallets, social media Apps, browsers, Defi Apps, and NFT platforms. While the UI/UX of the app is identical to that of any other standard application, the backend data storage of these applications is decentralized.

Some example Blockchain projects

1. `https://nevonprojects.com/blockchain-projects-development/`
2. `https://courses.cfte.education/blockchain-and-defi-projects/`
3. `https://www.hyperledger.org/projects`

Adding blocks to the network



CAP Theorem

Theorem

Any distributed system cannot have consistency, availability, and partition tolerance simultaneously.

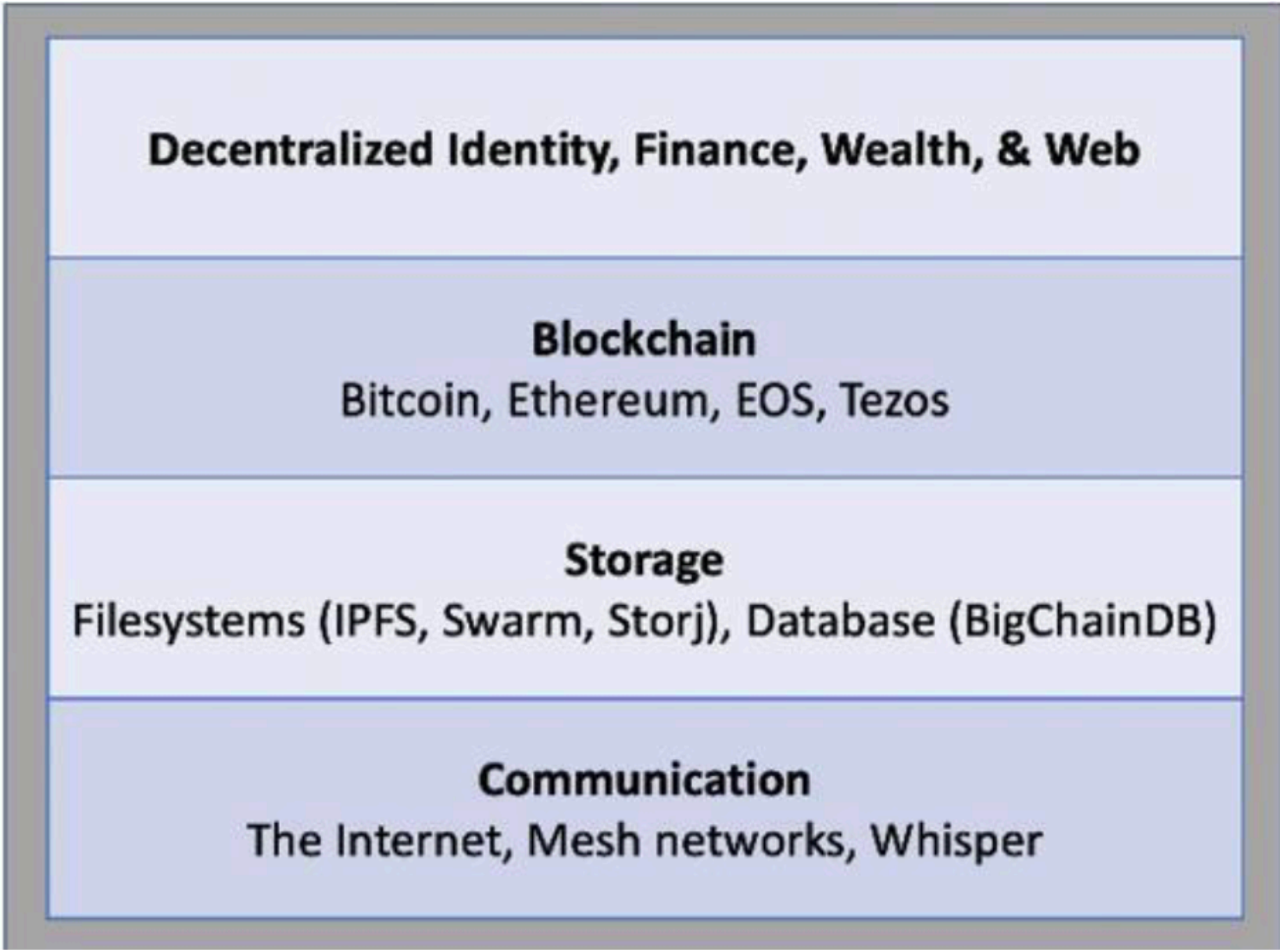
- **Consistency:** Implies all nodes in a distributed system have a single, current, and identical copy of the data.
- **Availability:** Means Nodes are up, accessible for use, and are accepting incoming requests and responding with data without any failures as and when required.
- **Partition tolerance:** The system continues to operate despite network failures (i.e., dropped partitions, slow network connections, or unavailable network connections between nodes.)

Questions to consider when decentralizing

The following four questions can guide on how a system can be decentralized:

1. What is being decentralized? e.g., identity system, trading system.
2. What level of decentralization is required? Either, full disintermediation or partial disintermediation.
3. What blockchain is used? e.g., Bitcoin blockchain, Ethereum blockchain, or any other blockchain that is deemed fit for the specific application.
4. What security mechanism is used? e.g., atomicity-based, reputation-based.

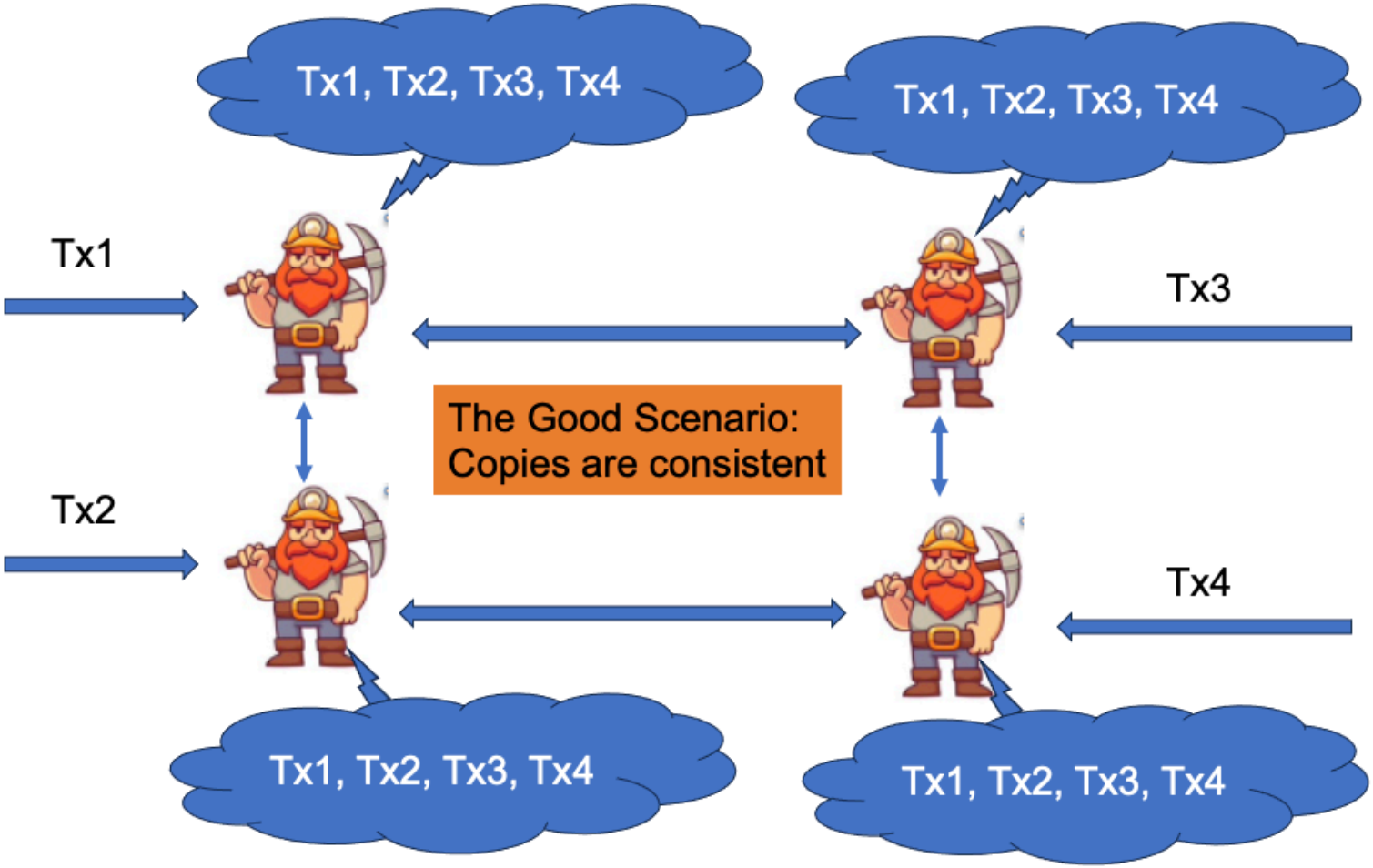
Computer power and decetralization



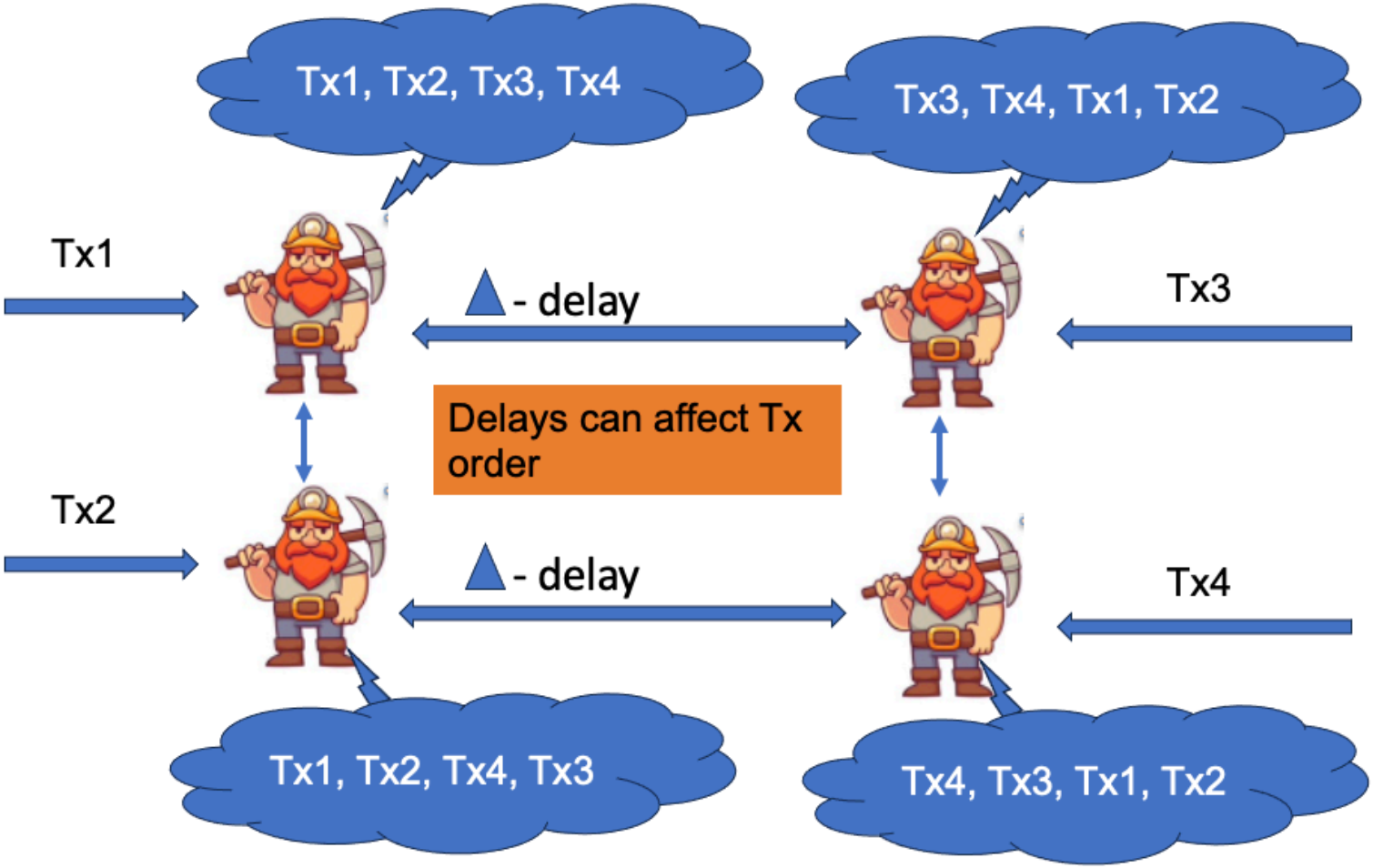
Computer power and decentralization (2)

- From previous slide we see an overview of a decentralized ecosystem.
- In bottom layer, the Internet or mesh networks provide a decentralized communication layer.
- The next layer is the storage layer that uses technologies such as IPFS and BigChainDB to enable decentralization.
  - Blockchain in a limited way provides a storage layer. But storing data on the blockchain can severely hamper speed and system capacity
  - IPFS and BigChainDB are more suitable for storing large amounts of data in a deccentralized way
- The Blockchain serves as a decentralized processing (computational) layer
- Top most layer is the identity and wealth layers that provides authentication (e.g., bitAuth and OpenID) and identification services with varying degrees of decentralization and security assumptions.

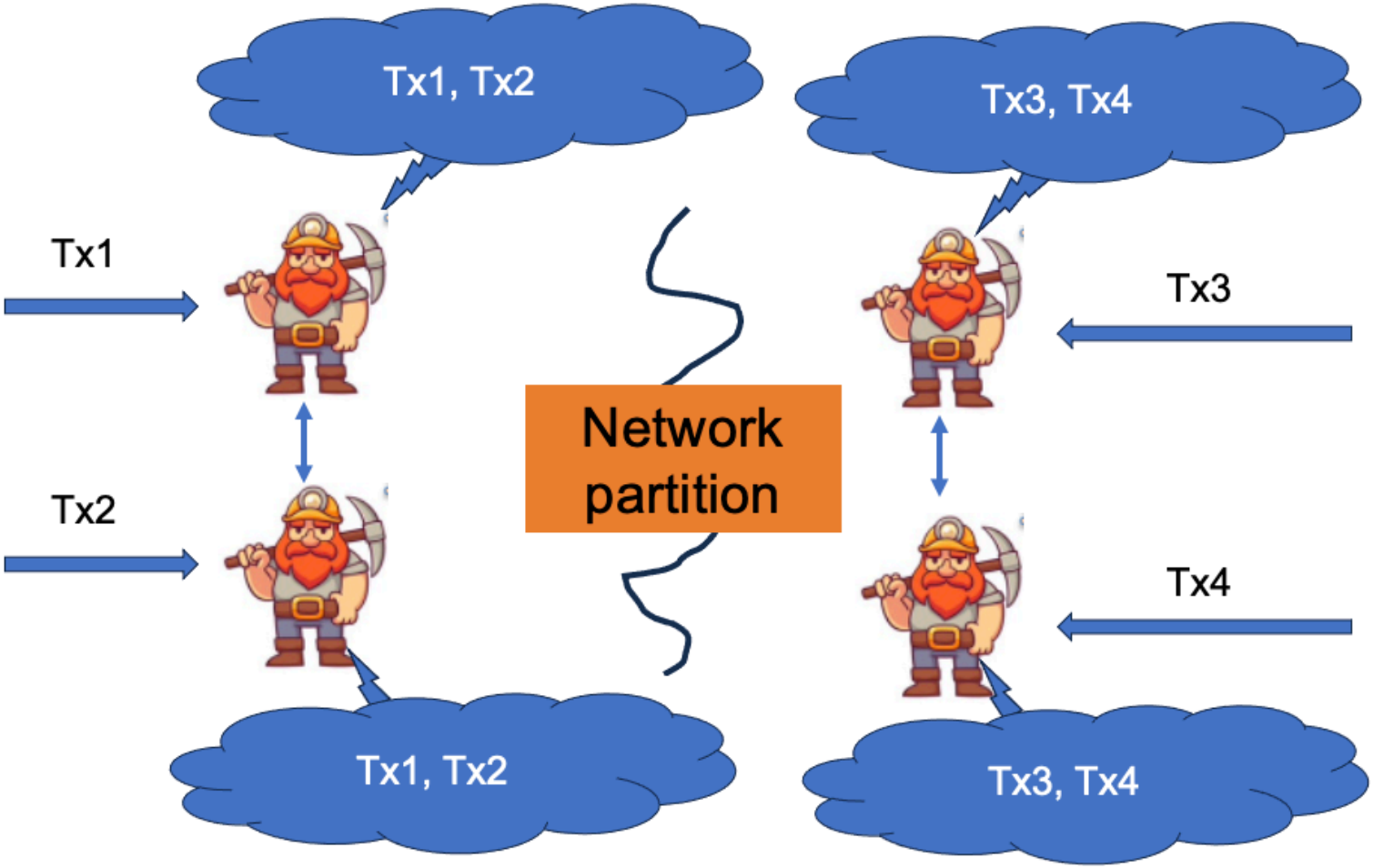
Why consensus is a hard problem



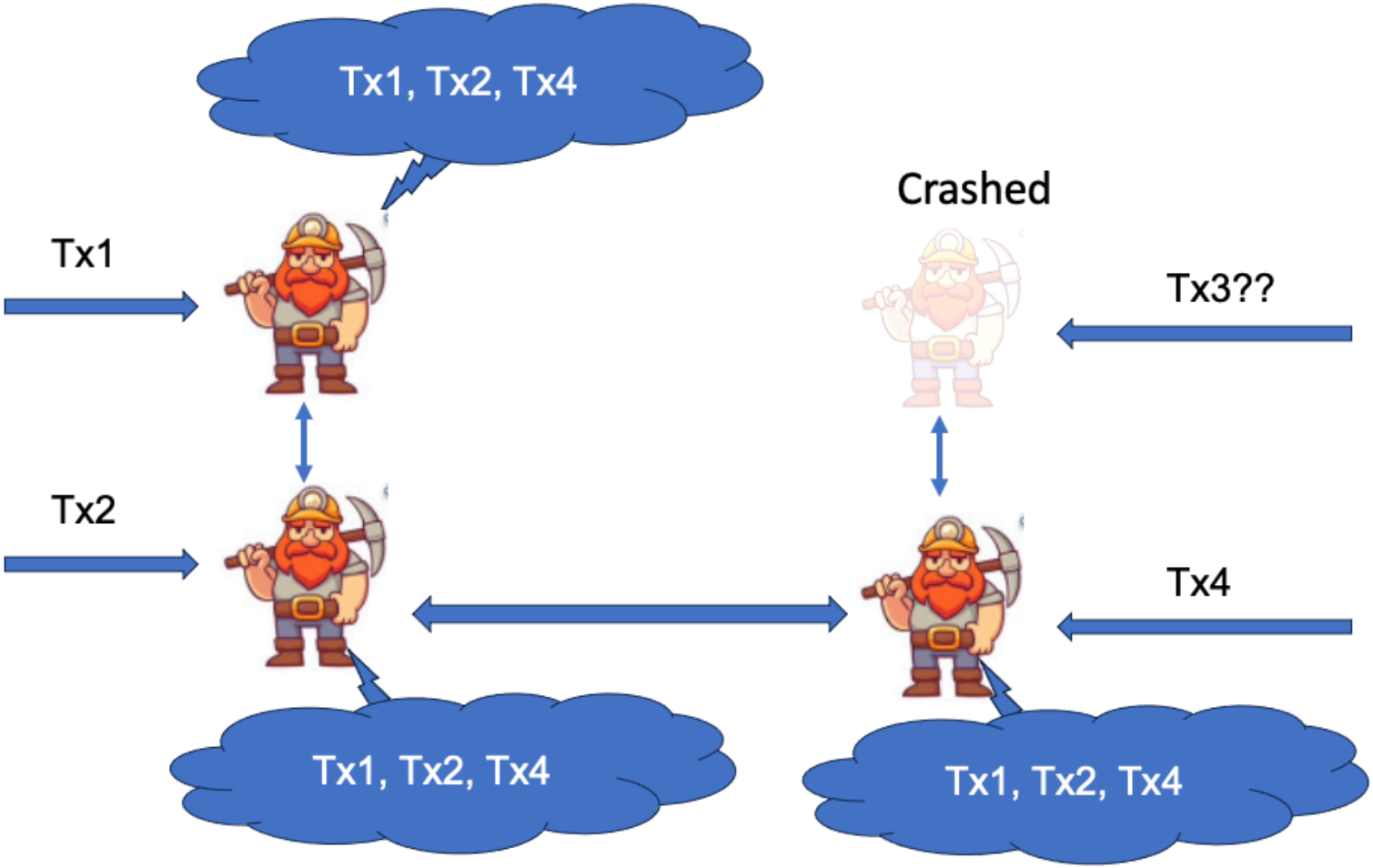
Why consensus is a hard problem (2)



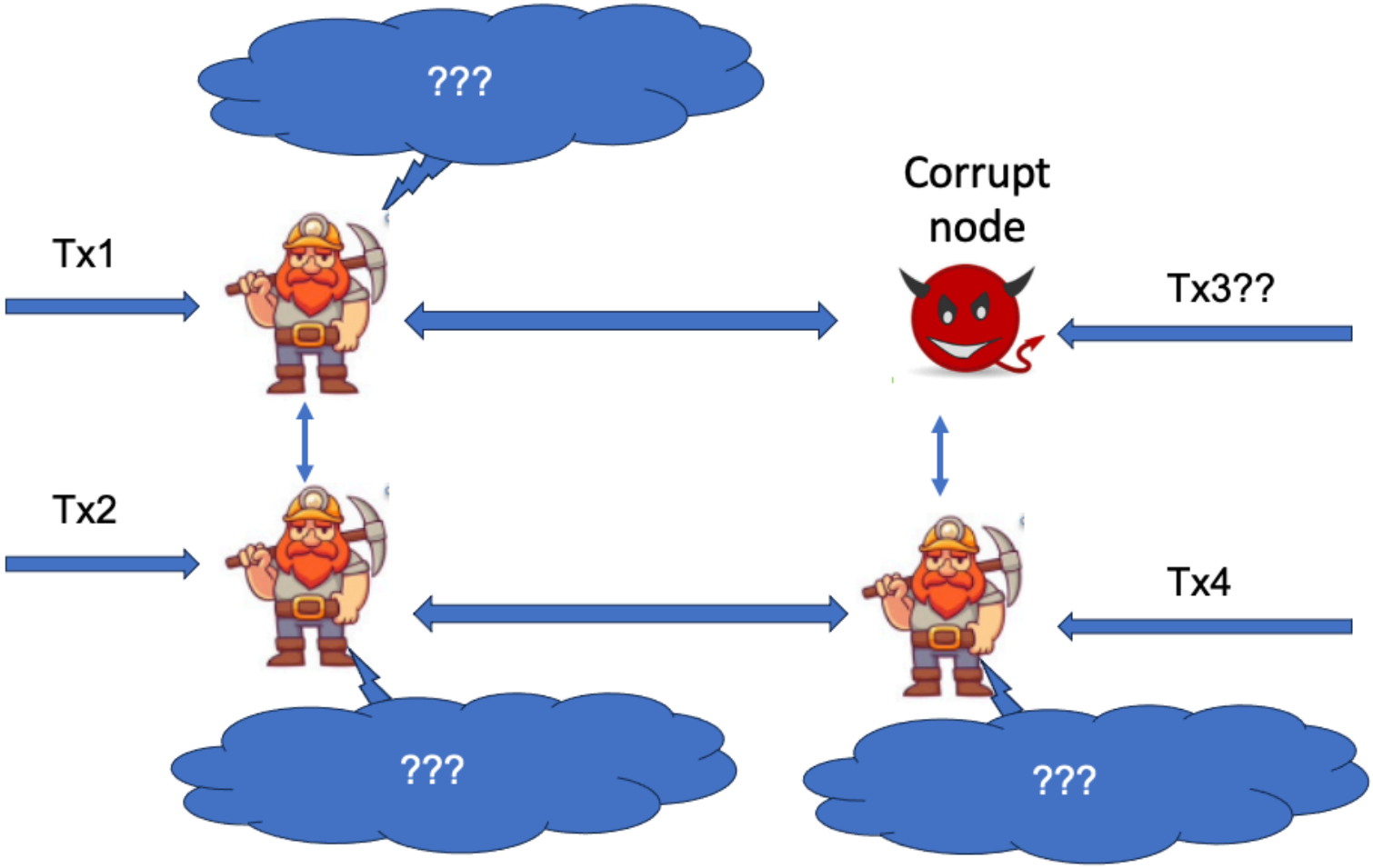
Why consensus is a hard problem (3)



Why consensus is a hard problem (4)



Why consensus is a hard problem (5)



Cryptographic Background

Definition (Cryptographic hash functions)

An efficiently computable function, $\mathcal{H} : \mathcal{M} \Longrightarrow \mathcal{T}$, where $|\mathcal{M}| >> |\mathcal{T}|$.

Basic properties

- (1) Deterministic, i.e. the same block of data always returns the same hash; (2) Quick to compute; (3) One-way function; given the hash one cannot derive the original value unless they brute-force all possible values (which is close to impossible for large data sets); (4) Any change to the original data, changes the resulting hash completely; (5)Collision resistant; no two different blocks of data give the same hash value

Hash functions properties

Definition (Collision)

A collision for $\mathcal{H} : \mathcal{M} \Longrightarrow \mathcal{T}$, is a pair $x \neq y$ such that $\mathcal{H}(x) = \mathcal{H}(y)$.

- Since $|\mathcal{M}| >> |\mathcal{T}|$, it implies many collisions exists.

Definition (Collision resistant)

A function $\mathcal{H} : \mathcal{M} \Longrightarrow \mathcal{T}$,is collision resistant if it is **hard** to find $x \neq y$ such that $\mathcal{H}(x) = \mathcal{H}(y)$.

Cryptographic Background (2)

Hash functions

Example (Collision resistant hash function)

SHA256: $\{x : len(x) < 2^{64} \text{ bytes}\} \Longrightarrow \{0,1\}^{256}$ (output is 32 bytes)

Applications

Example (Committing to data on a Blockchain)

Alice has a large file m. She posts $h = \mathcal{H}(m)$ (32 bytes). Bob reads h. Later Bob learns m' s.t. $\mathcal{H}(m') = h$.

- Since $\mathcal{H}$ is a CRF $\Longrightarrow$ Bob is convinced that $m' = m$ (otherwise, m and m' are a collision for $\mathcal{H}$)
- We say that $h = \mathcal{H}(m)$ is a **binding commitment** to m. (note: not hiding, h may leak information about m)

Cryptographic Background (3)

Applications

Example

Alice has $S = (m_1, m_2, \ldots, m_n)$ 32 bytes

- **Goal:**
  - Alice posts a short binding commitment to S, $h = commit(S)$
  - Bob reads h. Given $(m_i, proof\pi)$ can check that $S[i] = m_i$
  - Bob runs verify$(h, i, m_i, \pi_i) \Longrightarrow$ accept/reject

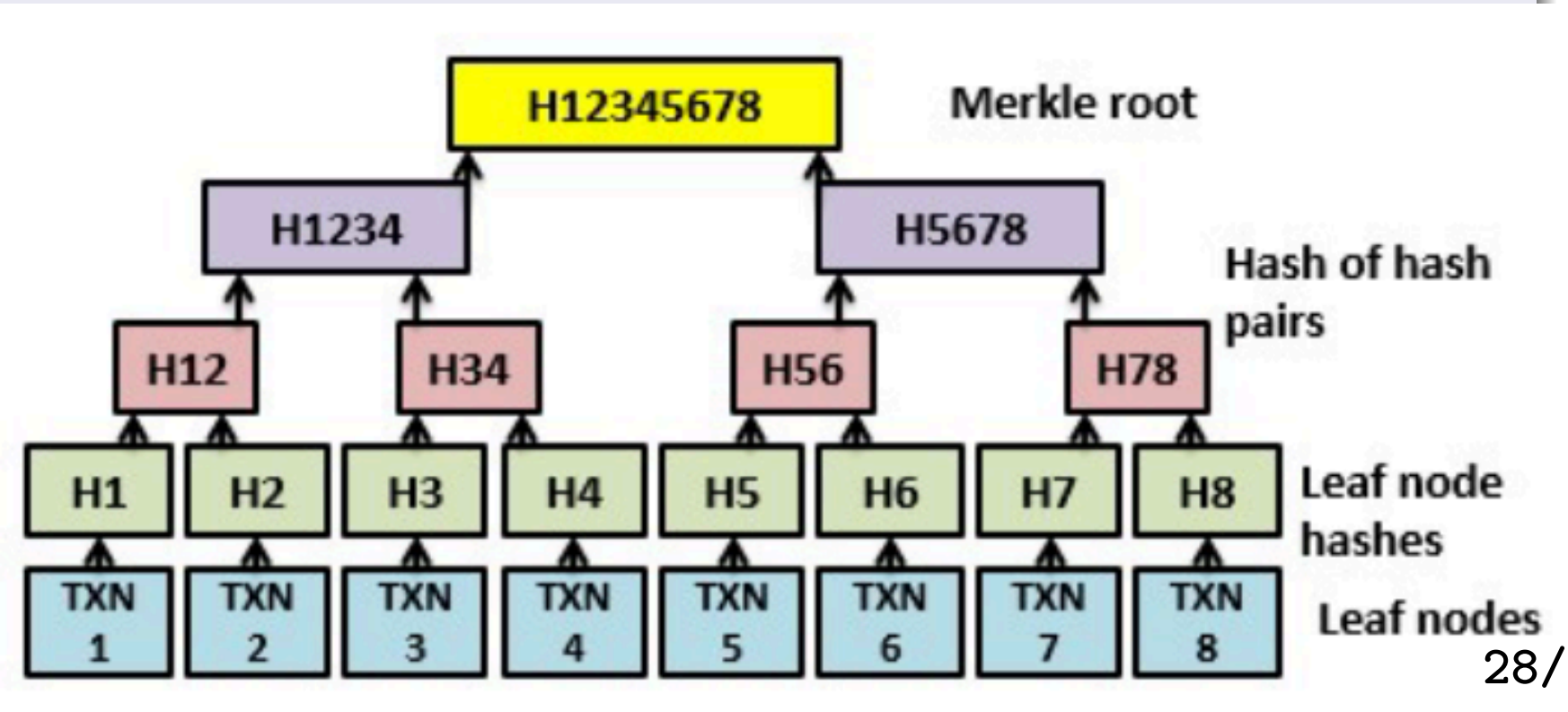
- **Security:** adv. cannot find (S, i, m, π) s.t. $m \neq S[i]$ and verify(h, i, m, π) =accept where $h = commit(S)$

Cryptographic Backgound (4)

Merkle tree

Definition

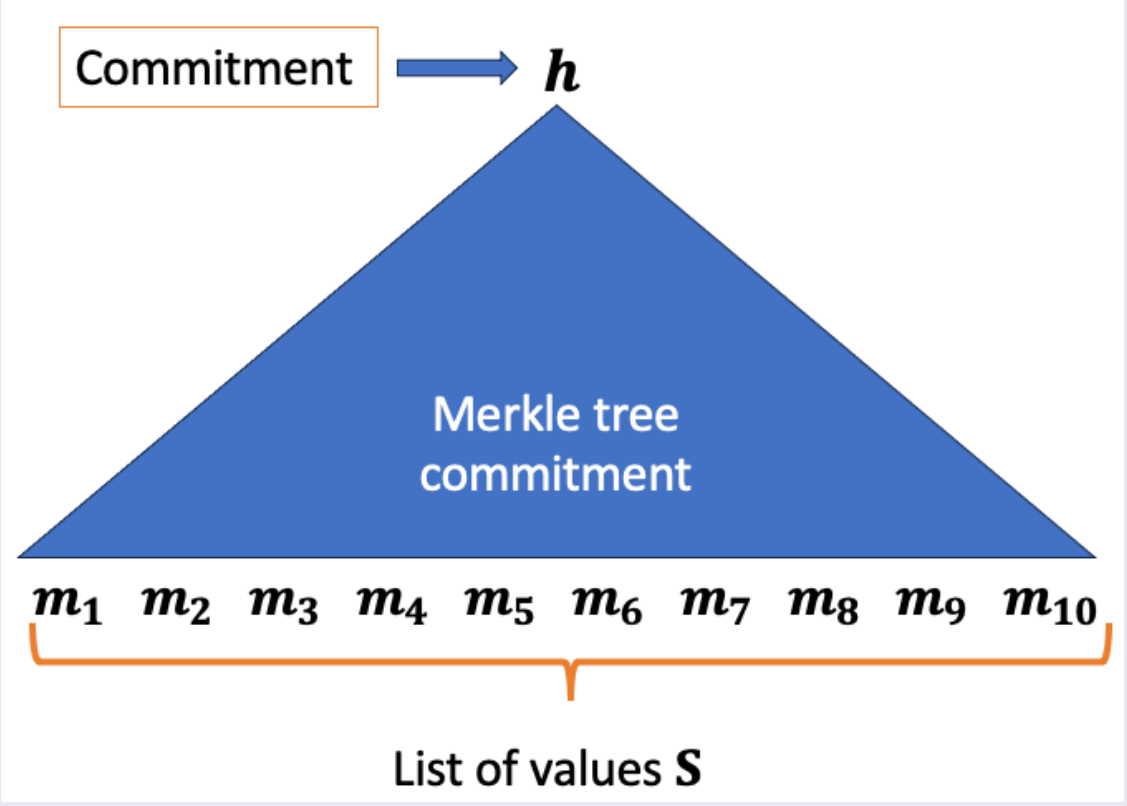
A Merkle tree is a data structure in which each leaf node is a hash of a block of data, and each non-leaf node is a hash of its children.



Cryptographic Background (5)

Merkle proof

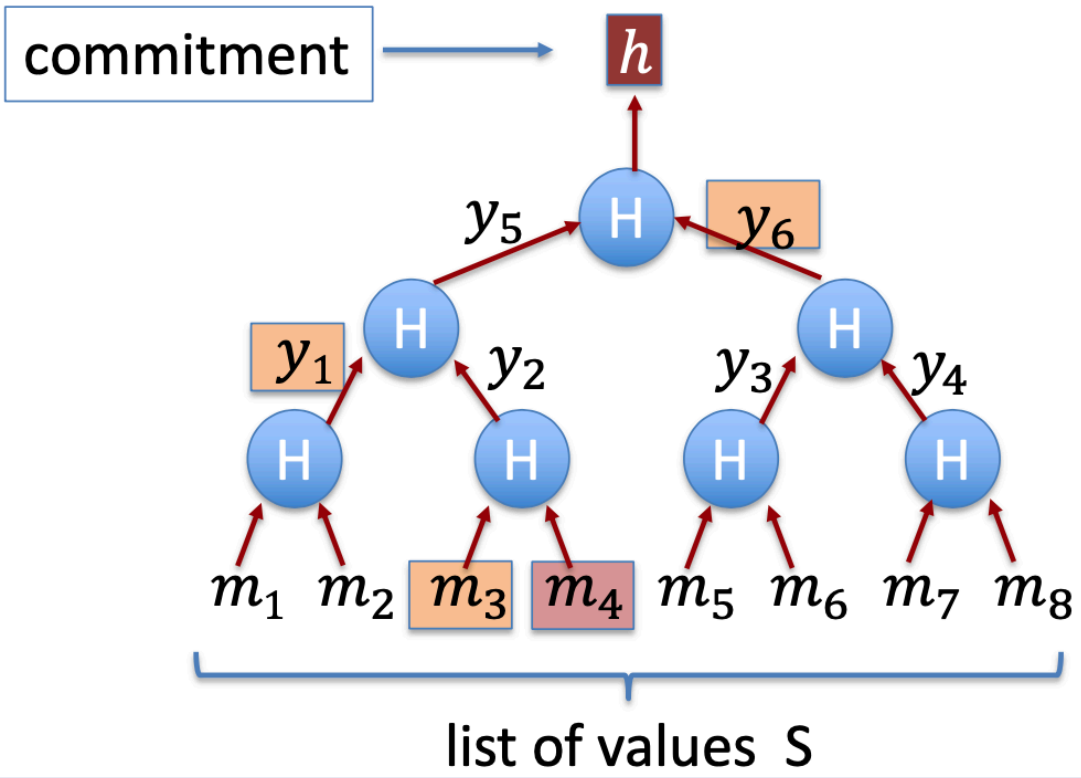
- An example of an **inclusion proof** - allows senders and receivers to verify specific data contained in a large dataset.



- Say, need to commit to list S of size n such that later anyone can prove that $S[i] = m_i$.

Cryptographic Background (6)

Merkle proof



Goal:

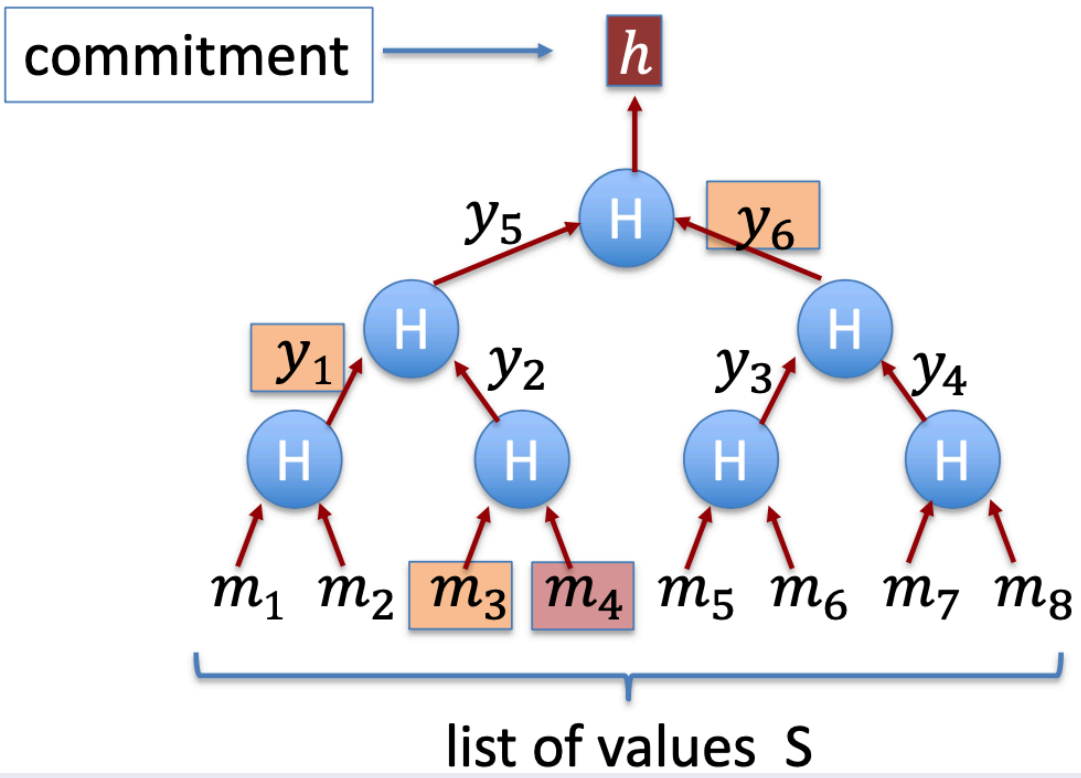
- commit to list S of size n
- Later prove $S[i] = m_i$

To prove $S[4] = m_4$,

proof $\pi = (m_3, y_1, y_6)$

length of proof: $\log_2 n$

Merkle tree



To prove $S[4] = m_4$,

proof $\pi = (m_3, y_1, y_6)$

Bob does:

$y_2 \leftarrow H(m_3, m_4)$

$y_5 \leftarrow H(y_1, y_2)$

$h' \leftarrow H(y_5, y_6)$

accept if $h = h'$

Cryptographic Background (8)

Merkle tree

Theorem

For a given n: if $\mathcal{H}$ is a CRF then, adversary cannot find (S, i, m, π) such that $|S| = n, m \neq S[i]$, $h = commit(S)$, and verify$(h, i, m, \pi) =$ accept

How is this useful?

To post a block of transactions S on the chain, one needs to only write commit(S) to the chain. This Keeps the chain small. Later can prove contents of every Tx.

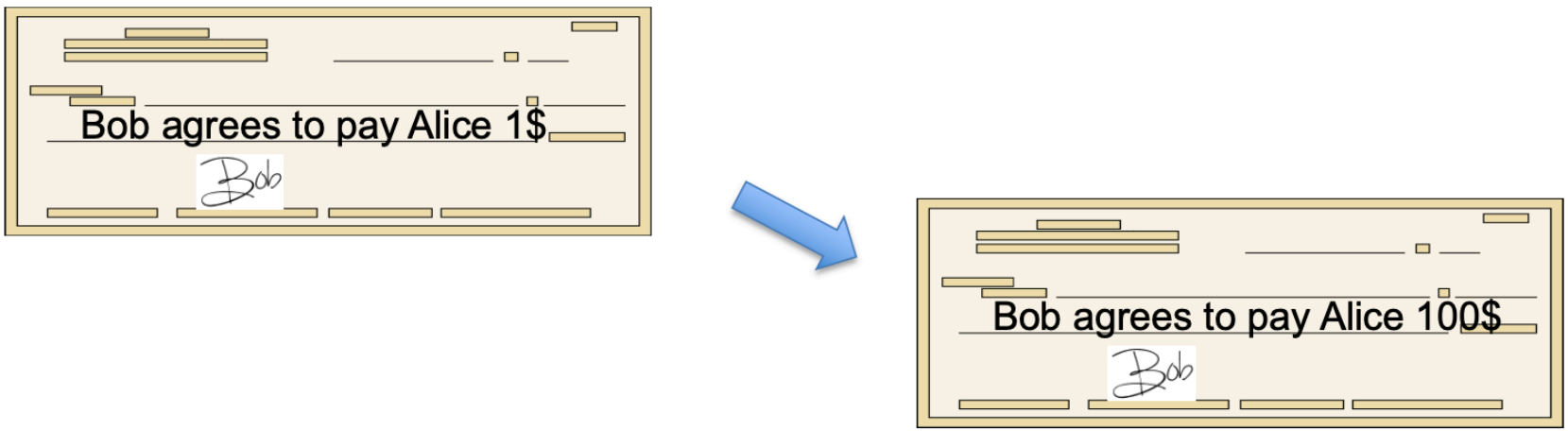
Applications of Merkle proofs

- Can be used to prove that a Tx is on the Blockchain.
- Useful during proof of work
- `https://ethereum.org/en/developers/tutorials/merkle-proofs-for-offline-data-integrity/`

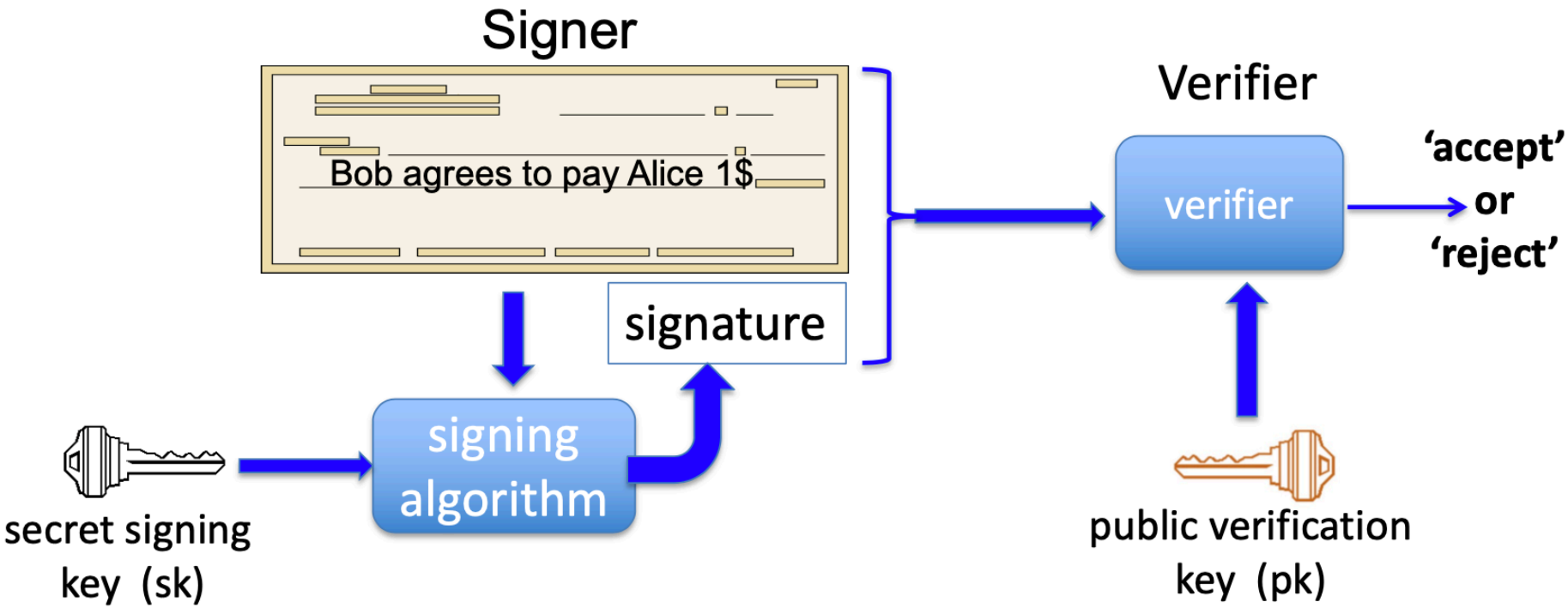
Cryptographic Background (9)

Digital signatures

- Physical signatures bind transaction to author.
- Unfortunately, in the digital world, anyone can copy Bob's signature from one document to another.



Solution: make signature depend on document



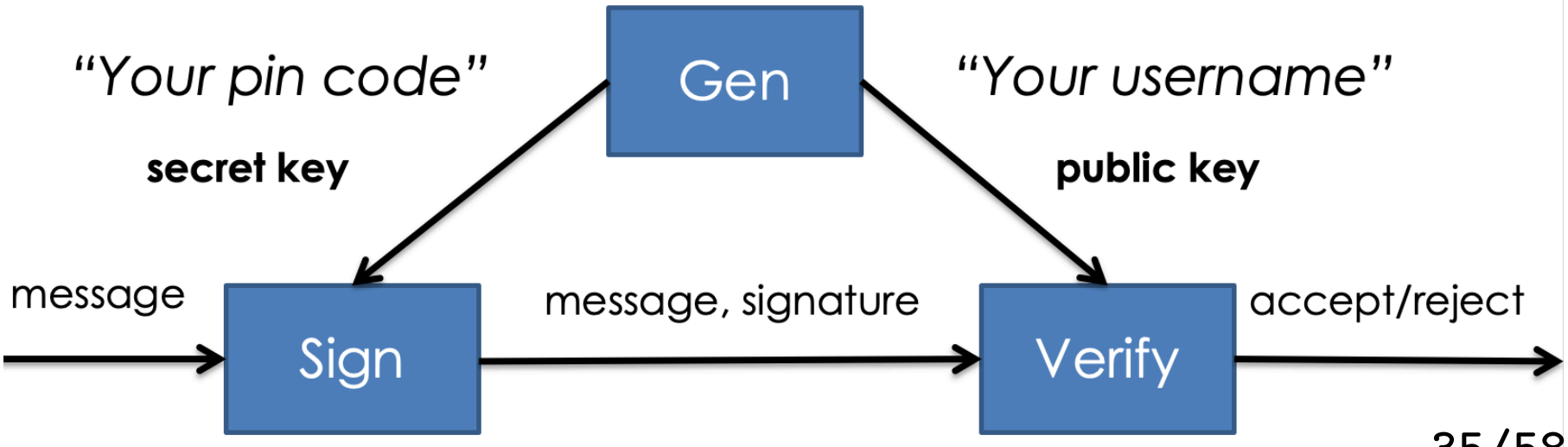
Cryptographic Background (11)

Digital signature

Definition

A signature scheme is triple of algorithms:

- **Gen():** outputs a key pair (pk, sk)
- **Sign**(sk, msg): outputs signature δ
- **Verify**(pk, msg, δ) outputs 'accept' or 'reject'



Cryptographic Background (12)

Families of signatures

1. RSA signatures (old ... not used in blockchains): long signatures and public keys ($\geq$256 bytes), fast to verify
2. Discrete-log signatures: Schnorr and ECDSA (Bitcoin, Ethereum). Short signatures(48 or 64bytes)and publickey(32bytes)
3. BLS signatures: 48 bytes, aggregatable, easy threshold (Ethereum 2.0, Chia, Dfinity)
4. Post-quantum signatures: long ($\geq$600 bytes)

Consensus Algorithms

Definition

A consensus algorithm is a procedure through which all the peers of the Blockchain network reach a common agreement about the present state of the distributed ledger.

Proof of Work (PoW) algorithm

- This consensus algorithm is used to select a miner for the next block generation.
- PoW algorithm involves solving a computationally challenging puzzle in order to create new blocks in the Bitcoin blockchain.
- This mathematical puzzle requires a lot of computational power and thus, the node who solves the puzzle as soon as possible gets to mine the next block.
- More information on PoW algorithm check: `https://www.geeksforgeeks.org/blockchain-proof-of-work-pow/`

Cryptocurencies using PoW

- Litecoin
- Ethereum
- Monero coin
- Dogecoin

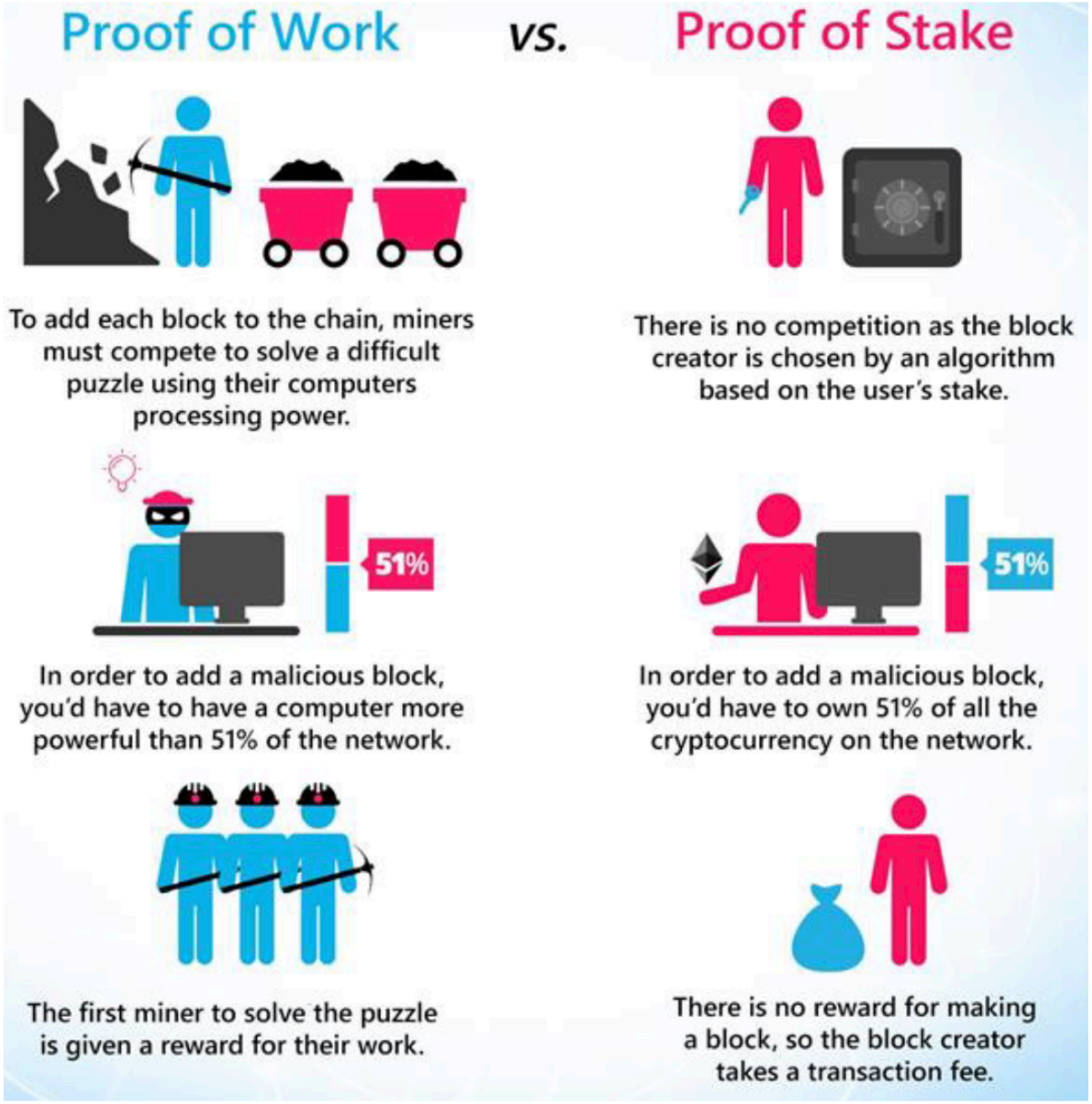
Proof of Stake (PoS) algorithm

- Used in Ethereum
- Under PoS, instead of investing in expensive hardware to solve a complex puzzle, validators invest in the coins of the system by locking up some of their coins as stake.
- Validators validate blocks by placing a bet on it if they discover a block which they think can be added to the chain.
- Based on the actual blocks added in the Blockchain, all the validators get a reward proportionate to their bets and their stake increase accordingly.
- A validator is chosen to generate a new block based on their economic stake in the network. Thus, PoS encourages validators through an incentive mechanism to reach to an agreement.

Cryptocurrencies using PoS

- Ethereum(Casper update)
- Peercoin
- Nxt

PoW Vs PoS


Proof of Work
vs.
Proof of Stake
To add each block to the chain, miners must compete to solve a difficult puzzle using their computers processing power.
There is no competition as the block creator is chosen by an algorithm based on the user's stake.
51%
In order to add a malicious block, you'd have to have a computer more powerful than 51% of the network.
51%
In order to add a malicious block, you'd have to own 51% of all the cryptocurrency on the network.
The first miner to solve the puzzle is given a reward for their work.
There is no reward for making a block, so the block creator takes a transaction fee.

Consensus Algorithms

Proof of burn (PoB) algorithm

- With PoB, instead of investing into expensive hardware equipment, validators 'burn' coins by sending them to an address from where they are irretrievable.
- By committing the coins to an unreachable address, validators earn a privilege to mine on the system based on a random selection process.
- Burning coins here means that validators have a long-term commitment in exchange for their short-term loss.

Proof of Capacity algorithm

- In the Proof of Capacity consensus, validators invest their hard drive space instead of investing in expensive hardware or burning coins.
- The more hard drive space validators have, the better are their chances of getting selected for mining the next block and earning the block reward.

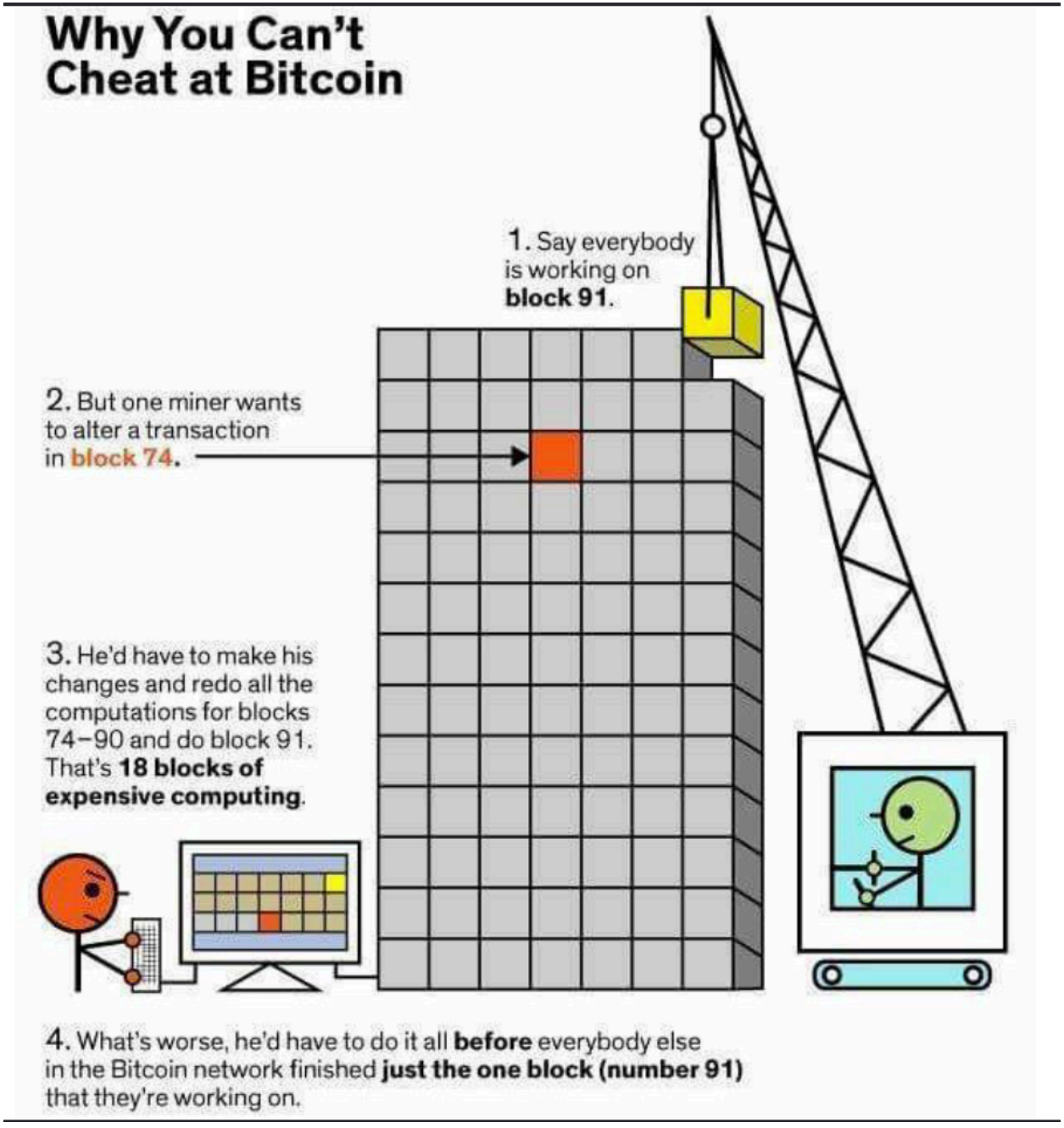
Consensus Algorithms

Proof of Elapsed Time (PoET) algorithm

- Considered the fairest consensus algorithms used in permissionned Blockchain networks.
- Every validator on the network gets a fair chance to create their own block. All the nodes do so by waiting for random amount of time, adding a proof of their wait in the block.
- The created blocks are broadcasted to the network for others consideration.
- The winner is the validator which has least timer value in the proof part.
- The block from the winning validator node gets appended to the Blockchain.
- There are additional checks to stop nodes from always winning the election, stop nodes from generating a lowest timer value.

- Other Algorithms include, Proof of Activity, Proof of Weight, Proof of Importance, Leased, Proof of Stake, etc.

Possible to cheat??



Wallets

- The wallet software is used to store private or public keys and bitcoin address.
- Can performs various functions, such as receiving and sending bitcoins.

Non-Deterministic Wallets

- Contain randomly generated private keys and are also called Just a Bunch of Key wallets
- Bitcoin core client generates some keys when first started and generates keys as and when required.

Deterministic wallets

- Keys are derived out of a seed value via hash functions.
- The seed number is generated randomly and is commonly represented by humanreadable mnemonic code words

Wallets (2)

Hardware wallets

- Hardware wallets are hardware devices that individually handle public addresses and keys.
- It looks like a USB with OLED screen and side buttons.
- when you open a wallet (in the hardware wallet or software wallet) you are provided with 2 pair of keys (sometimes more).
- Most popular hardware wallets are Ledger Nano S and Trezor.

Wallets (3)

Paper wallets

- It is a physically printed QR coded form wallet.
- Some wallets allow downloading the code to generate new addresses offline.

Wallets (4)

Desktop wallets

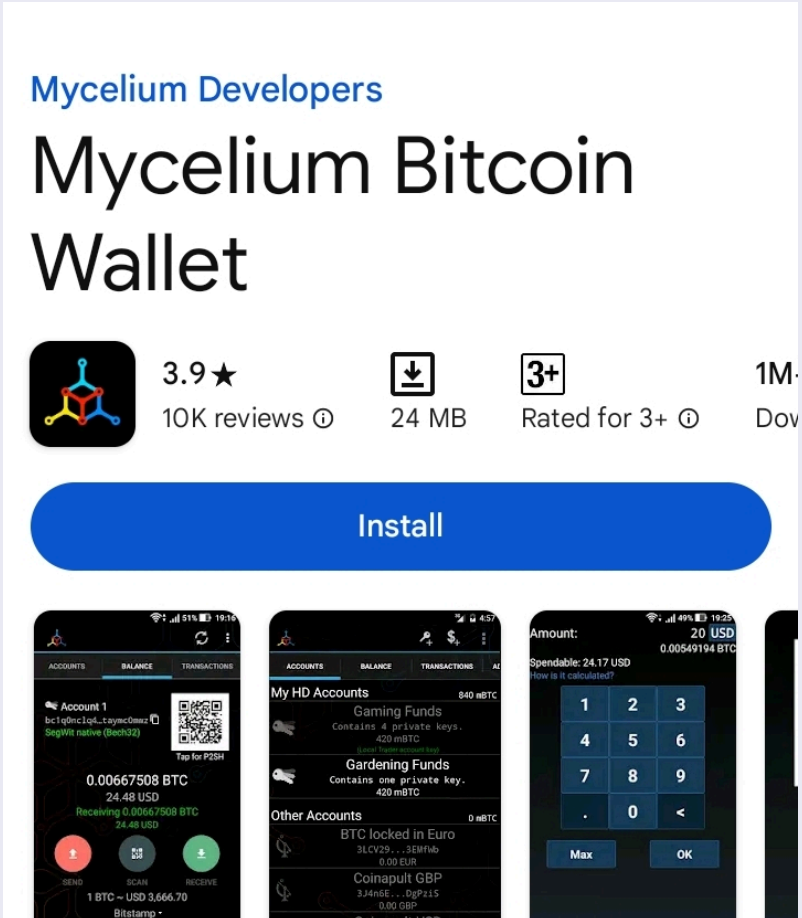
- Desktop wallets are programs that store and manage the private key for your Bitcoins on your computer's hard drive.

Electrum	Exodus	Bitcoin Core	Atomic Wallet
Type: SPV	Type: SPV	Type: Full node	Type: SPV
Beginner friendly: No	Beginner friendly: Yes	Beginner friendly: No	Beginner friendly: Yes
Platforms: Desktop only	Platforms: Desktop, mobile	Platforms: Desktop only	Platforms: Desktop, mobile
Visit website	Visit website	Visit website	Visit website

Wallets (5)

Mobile wallets

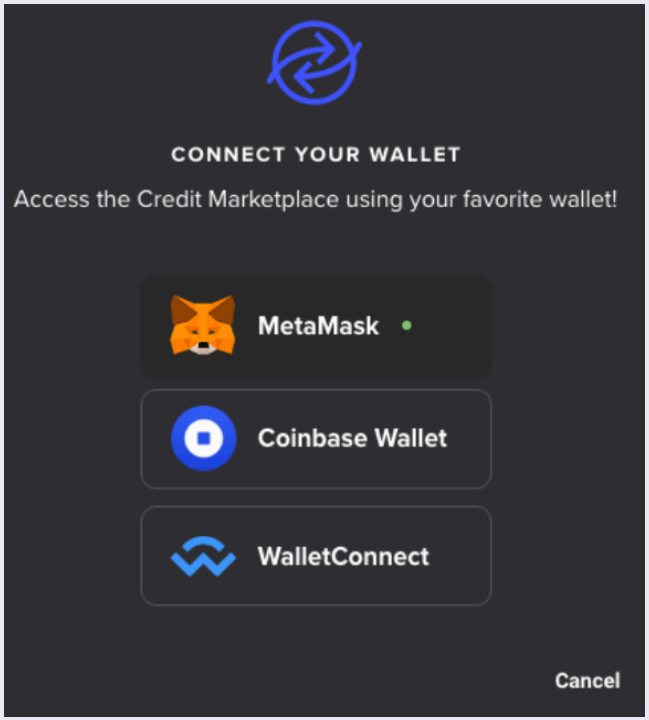
- A mobile wallet is a virtual wallet that stores payment card information on a mobile device.
- They are quite convenient as it uses QR codes for transactions
- Some mobile wallets are Coinomi and Mycelium



Wallets (6)

Web wallets

- These wallets are accessed by internet browsers.
- They are the least secure wallets.
- They are not the same as hot wallets.
- They are ideal for small investments and allow quick transactions.
- Some of these are MetaMask and Coinbase.



Wallets (7)

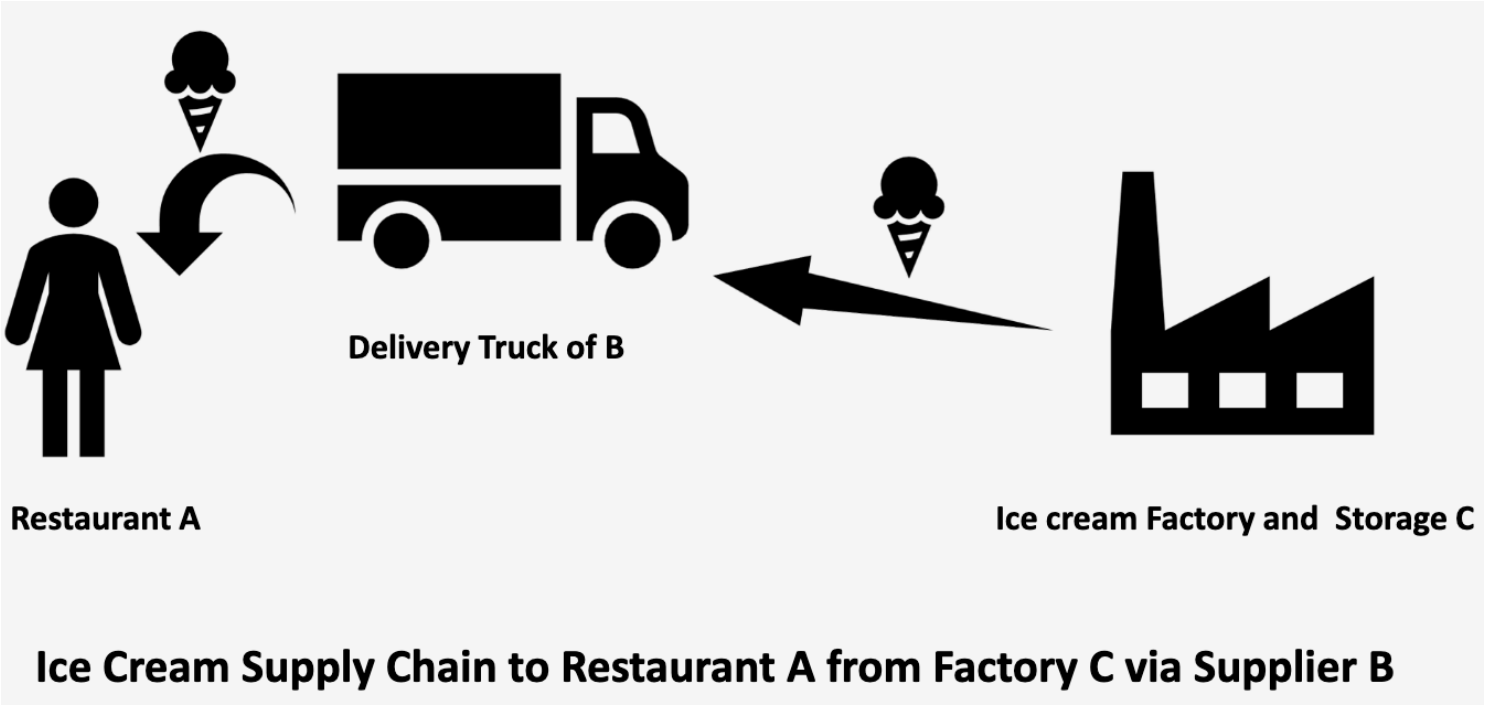
Online wallets

- Are stored entirely online and are provided as a service usually via cloud.
- A web interface is provided to the users to manage their wallets and perform various functions such as making and receiving payments.

Mobile wallets

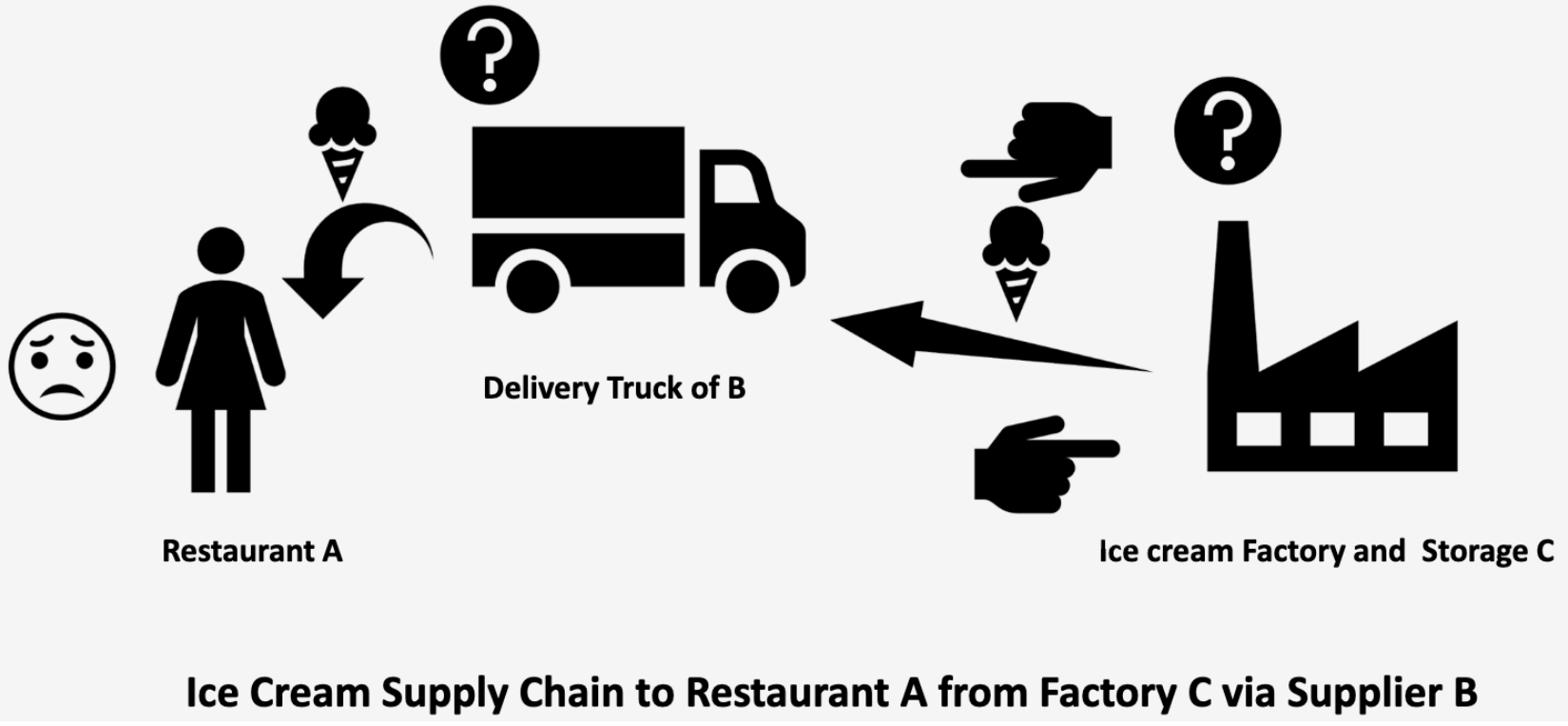
- Are installed on mobile devices.
- Provide various methods to make payments, most notably the ability to use smart phone cameras to scan QR codes quickly and make payments.
- Mobile wallets are available for the Android platform and iOS, for example, breadwallet, copay, and Jaxx.

A re-look at the supply chain problem



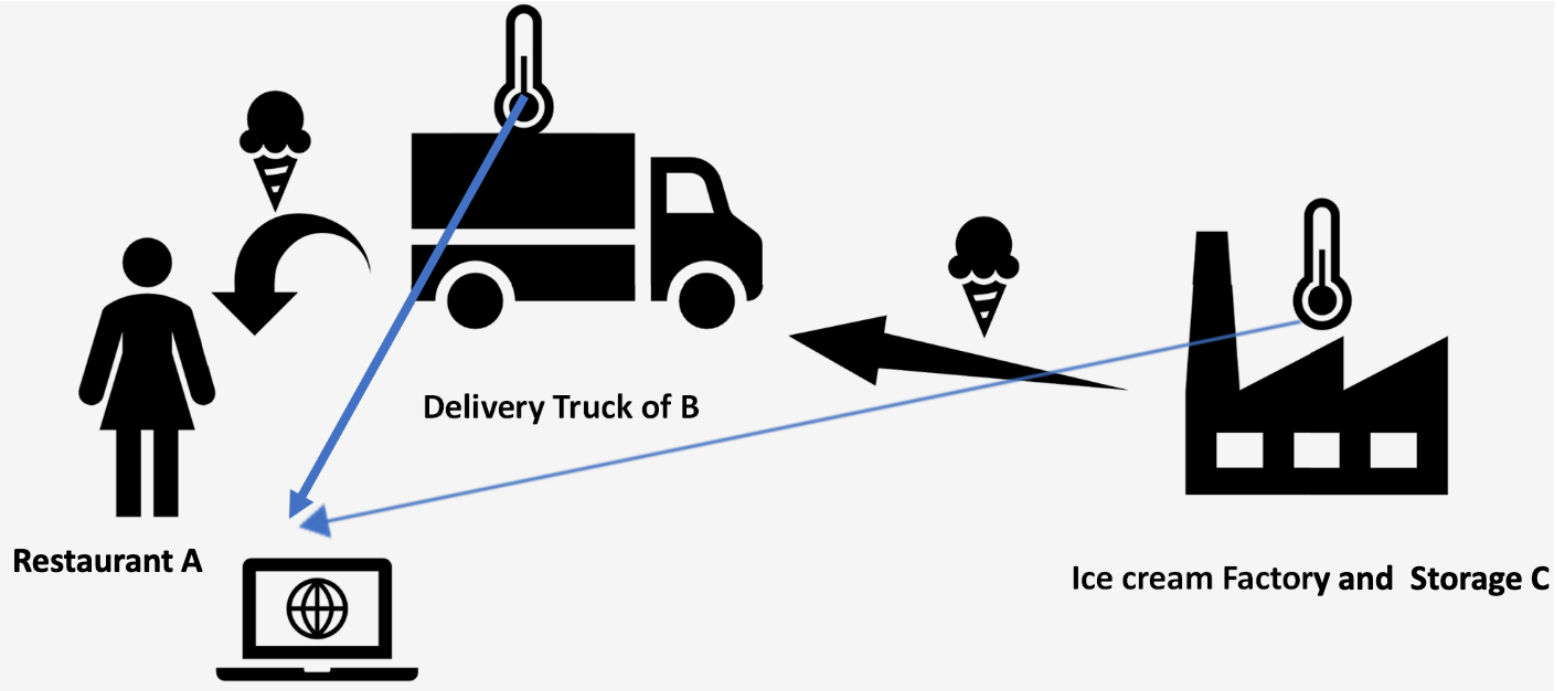
Ice Cream Supply Chain to Restaurant A from Factory C via Supplier B

supply chain problem - Ice cream melted



Ice Cream Supply Chain to Restaurant A from Factory C via Supplier B

Use IoT - to create non-repudiation



IoT sensors sends real-time data to the server at Restaurant A to periodically show the factory and the truck temperatures to Restaurant A

What can go wrong?

- IoT sensor data may be intercepted by a middle man and changed before it reaches the server **(data integrity)**
- IoT sensors may be stopped and old readings may be replayed (replay attack)
- What the server gets purportedly from factory C, may be manufactured by supplier B **(Authenticity)**
- If restaurant A claims that C' s temperature reading shows that ice cream was melting in the storage, C can say that message you received is not from me -there was an MITM attack **(repudiation)**
- So restaurant A will not be able to pinpoint any one in the supply chain with full confidence!!

What can be done?

- Use a message integrity proof **(Hashing)**
- Use digital signature of the individual IoT devices **(Authenticity and non-repudiation)**
  - assuming the digital signatures cannot be forged
  - private keys are kept safe
- Use authentic time stamping with the IoT data before hashing for integrity **(avoid replay attacks)**
- So now factory A can pinpoint with some basic security assumptions about this infrastructure

Concurrency issues

- A has other suppliers for other goods required for its business **(multiple concurrent supply chains)**
- B and C has multiple other consumers of their services
- Assuming N suppliers who are also consumers of some of these entities, we have an N^2 messaging problem. A offers that every one can look up their data from my server, so you can get linear number of messaging.

Concurrency issues (2)

- But do you trust A as purveyors of your data?

Possible solution

- Have a trusted authority or a cloud provider to become a publish-subscribe service provider
- Every supplier sends their IoT data with message integrity, authentication code etc., to the cloud server
- Every consumer subscribes to the events they are interested in on the cloud
- Every supplier becomes authenticated data generator on the cloud

- What if the cloud provider cannot be trusted?

Create a framework on which data is crowd sourced, validated by the crowd for the crowd?

- You get a block chain
- But now the question is as concurrent messages come in to this framework, how do you order them?
- DISTRIBUTED CONCENSUS IS REQUIRED TO DECIDE

1. of all messages coming in concurrently how are they ordered
2. But if some of the crowd are malicious, and tries to allow data that are wrong, or ordered wrong?
3. You need Byzantine fault-tolerant consensus