Blockchain: Smart Contracts

Lecture 2

Please join course TEAMS: oob4snl

Basic cryptographic primitives

Basic cryptographic primitives behind the blockchain technology

- **Cryptographically Secure Hash Function**
- **Digital Signature**

Hash Function: Used to connect the “blocks” in a “chain” in a tamper-proof way

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

Cryptographic Hash Functions

Takes any arbitrarily sized string as input

- Input M: The message

Fixed size output (We use 256 bits in Blockchain)

- Output H(M): We call this as the message digest

Efficiently computable

Cryptographic Hash Function: Properties

Deterministic

Always yield identical hash value for identical input data

Collision-Free

If two messages are different, then their digests also differ

Hiding

Hide the original message

Puzzle-friendly

Given *X* and *Y*, find out *k* such that $Y = H(X||k)$ - used to solve the mining puzzle in Bitcoin Proof of Work

Collision Free

Hash functions are one-way; Given an x, it is easy to find $H(x)$. However, given an $H(x)$, **no deterministic algorithm** can find x

It is **difficult to find** x and y, where $x \neq y$, but $H(x) = H(y)$

Note the phrase **difficult to find**, collision is **not impossible**

Try with randomly chosen inputs to find out a collision – but it takes too long

Collision Free – How Do We Guarantee

It may be relatively easy to find collision for some hash functions

Birthday Paradox: Find the probability that in a set of n **randomly chosen persons,** some of them will have the same birthday

- By *Pigeonhole Principle*, the probability reaches 1 when number of people reaches 366 (not a leap year) or 367 (a leap year)
- 0.999 probability is reached with just ~70 people, and 0.5 probability is reached with only ~23 people

Collision Free – How Do We Guarantee

Birthday paradox places an upper bound on collision resistance

If a hash function produces N bits of output, an attacker need to compute only $2^{\frac{N}{2}}$ hash operations on a random input to find two matching outputs with probability > 0.98

For a 256 bit hash function, the attacker needs to compute 2^{128} hash operations – this is significantly time consuming

- If every hash computation takes only 1 microsecond, it will need $\sim 10^{25}$ years

Hash as A Message Digest

If we observe $H(x) = H(y)$, it is safe to assume $x = y$

We need to remember just the hash value rather than the entire message – we call this as the **message digest**

To check if two messages x and y are same, $\text{i.e.}, \text{whether } x = \mathrm{y}$, simply check if $H(x) = H(y)$

This is efficient because the size of the digest is significantly less than the size of the original messages

Hashing - Illustration

http://www.blockchain-basics.com/HashFunctions.html

Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Information Hiding through Hash

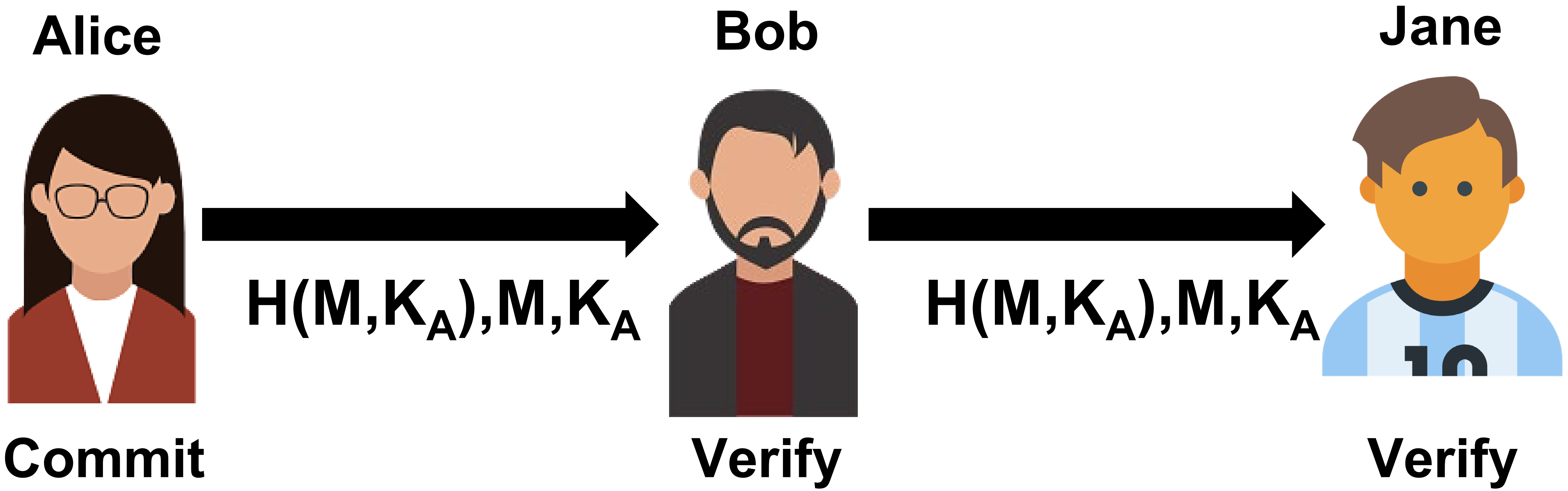
Given an $H(x)$, it is “computationally difficult” to find x

The difficulty depends on the size of the message digests

Hiding helps to commit a value and then check it later

- Compute the message digest and store it in a digest store – commit
- To check whether a message has been committed, match the message digest at the digest store

Message Commitment through Multiple Parties



Alice

Bob

Jane

$H(M,K_A),M,K_A$

$H(M,K_A),M,K_A$

Commit

Verify

Verify

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

Puzzle Friendly

Say M is chosen from a widely spread distribution; it is computationally difficult to compute k, such that $Z = H(M||k)$, where M and Z are known a priori.

A Search Puzzle (Used in Bitcoin Mining)

M and Z are given, k is the search solution

Note: It might be not exactly a particular value Z, but some properties that Z satisfies, i.e., Z could be a set of possible values

Puzzle friendly property implies that random searching is the best strategy to solve the above puzzle

Hash Function – SHA256

SHA256 is used in Bitcoin mining – to construct the Bitcoin blockchain

Secure Hash Algorithm (SHA) that generates 256 bit message digest

A part of SHA-2, a set of cryptographic hash functions designed by United States National Security Agency (NSA)

SHA256 Algorithm - Preprocessing

Pad the message such that the message size is a multiple of 512

Suppose that the length of the message M is l; and $l \ mod \ 512 \neq 0$

Append the bit “1” at the end of the message

Append k zero bits, where k is the smallest non-negative solution to the equation $l + 1 + k \equiv 448 \ mod \ 512$

Append the 64-bit block which is equal to the number l written in binary

The total length gets divisible by 512

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

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

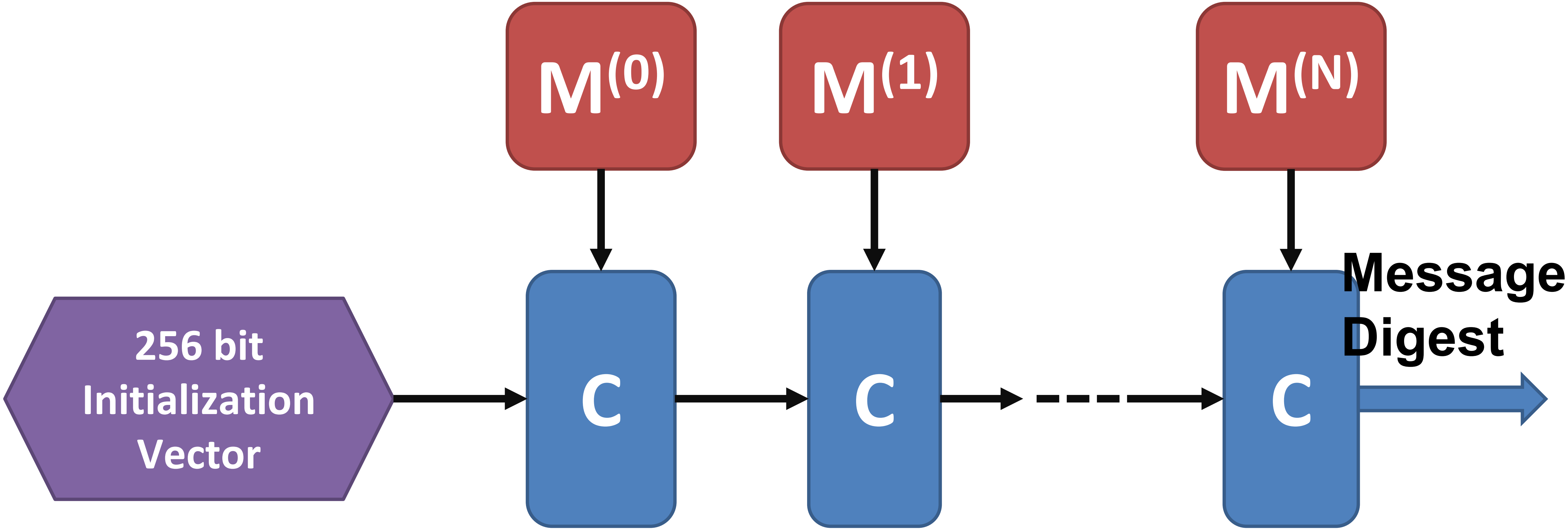
SHA-256 Algorithm

The message blocks are processed one at a time

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

Sequentially compute $H^{(i)} = H^{(i-1)} + C_{M^{(i)}}(H^{(i-1)})$; C is the SHA-256 *compression function* and + means mod 2^{32} addition. $H^{(N)}$ is the hash of M.

SHA-256 Algorithm



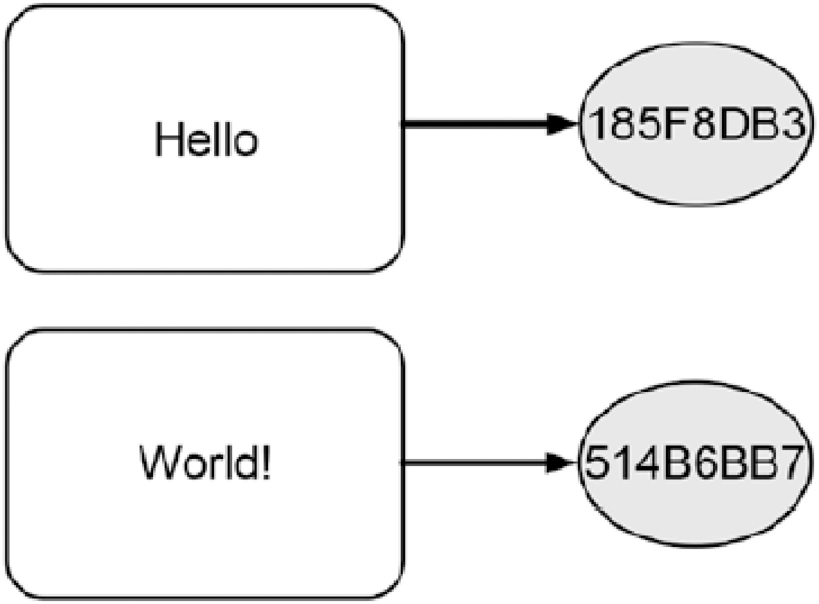
Patterns of Hashing Data

Independent hashing
Repeated hashing
Combined hashing
Sequential hashing
Hierarchical hashing

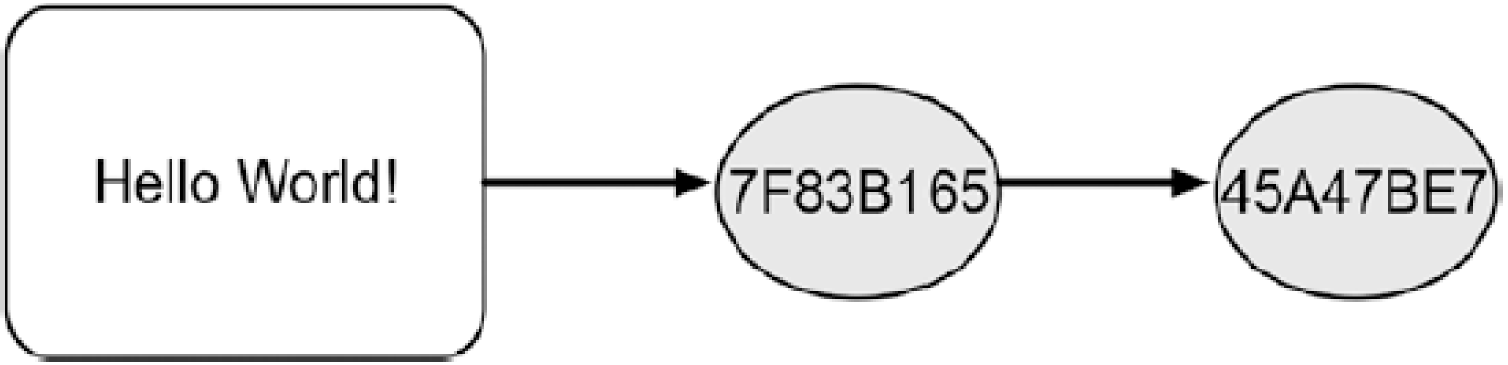
Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Types of Hashing

Independent hashing

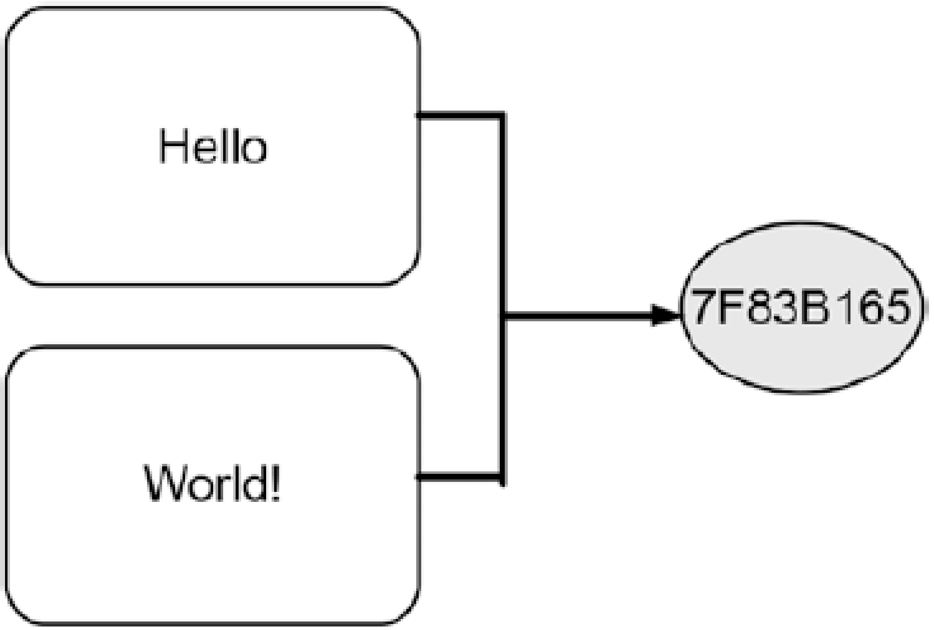


Repeated hashing

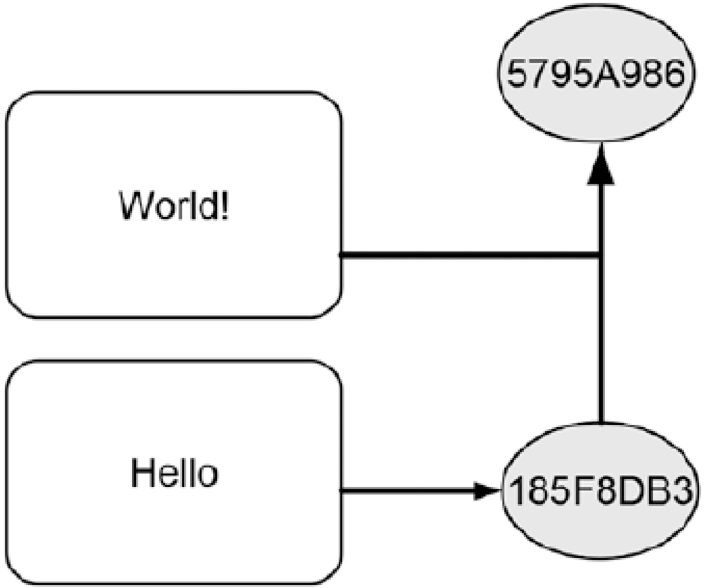


Types of Hashing

Combined hashing

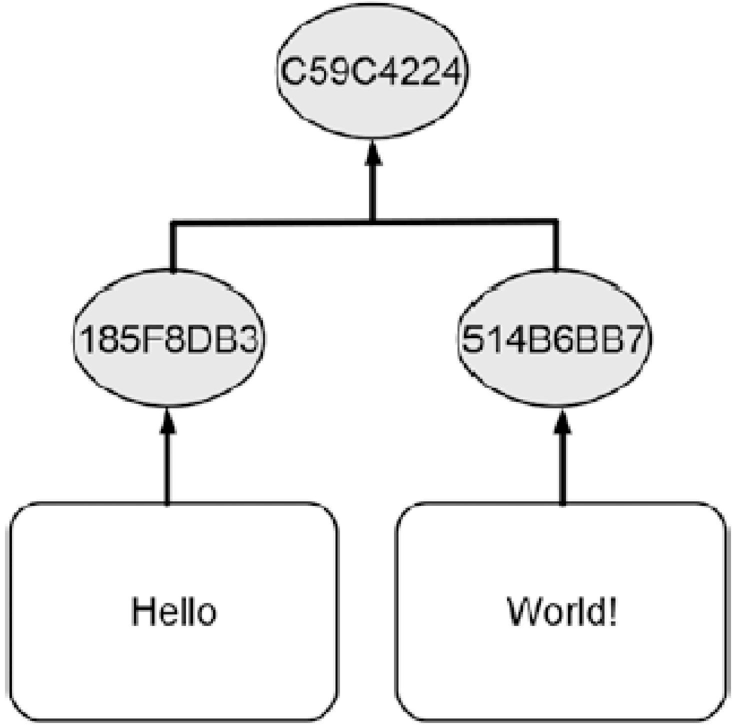


Sequential hashing



Types of Hashing

Hierarchical hashing



Hash Pointer

A **Cryptographic Hash Pointer** (Often called Hash Reference) is a pointer to a location where

Some information is stored

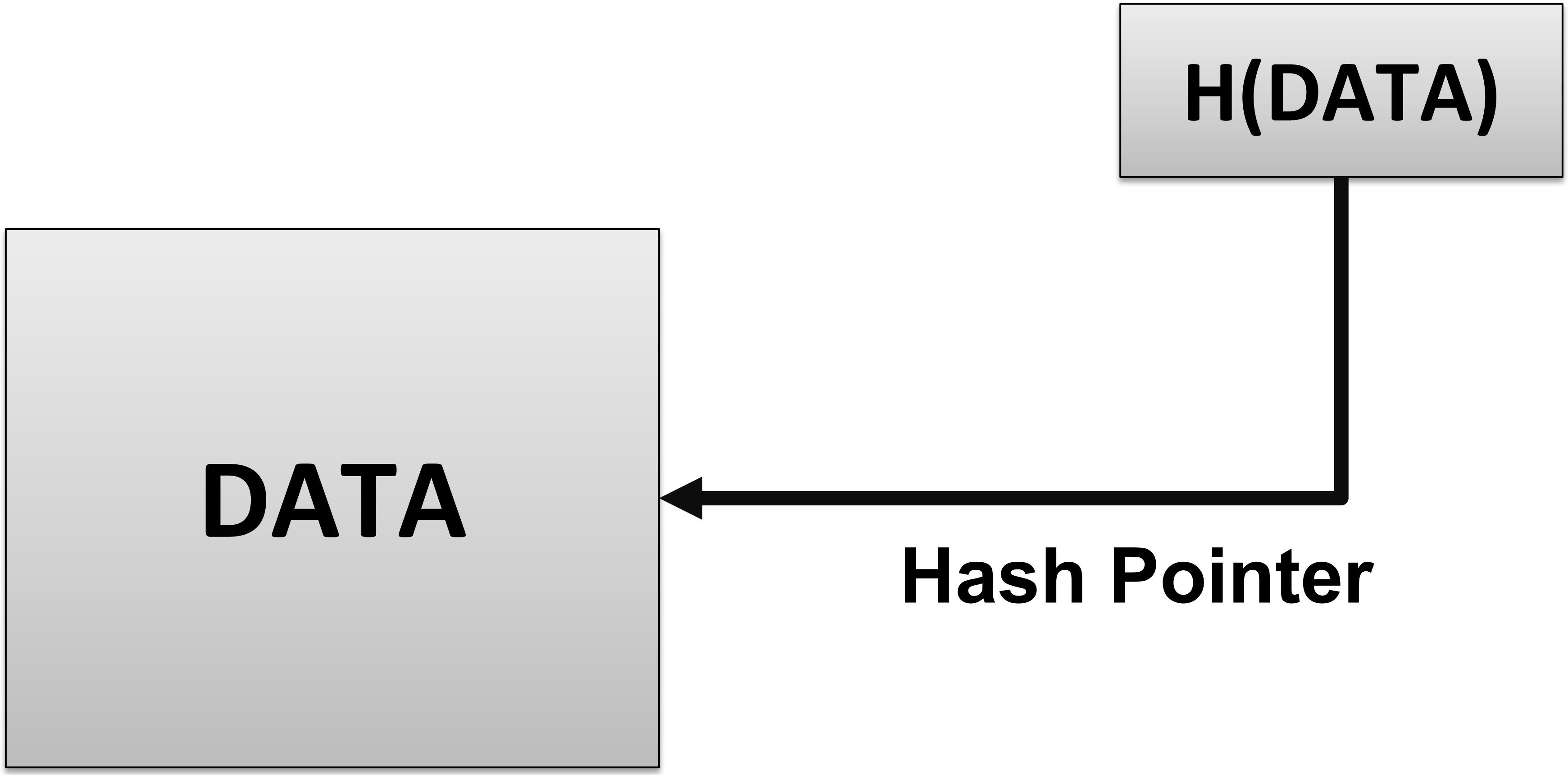
Hash of the information is stored

With the hash pointer, we can

Retrieve the information

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

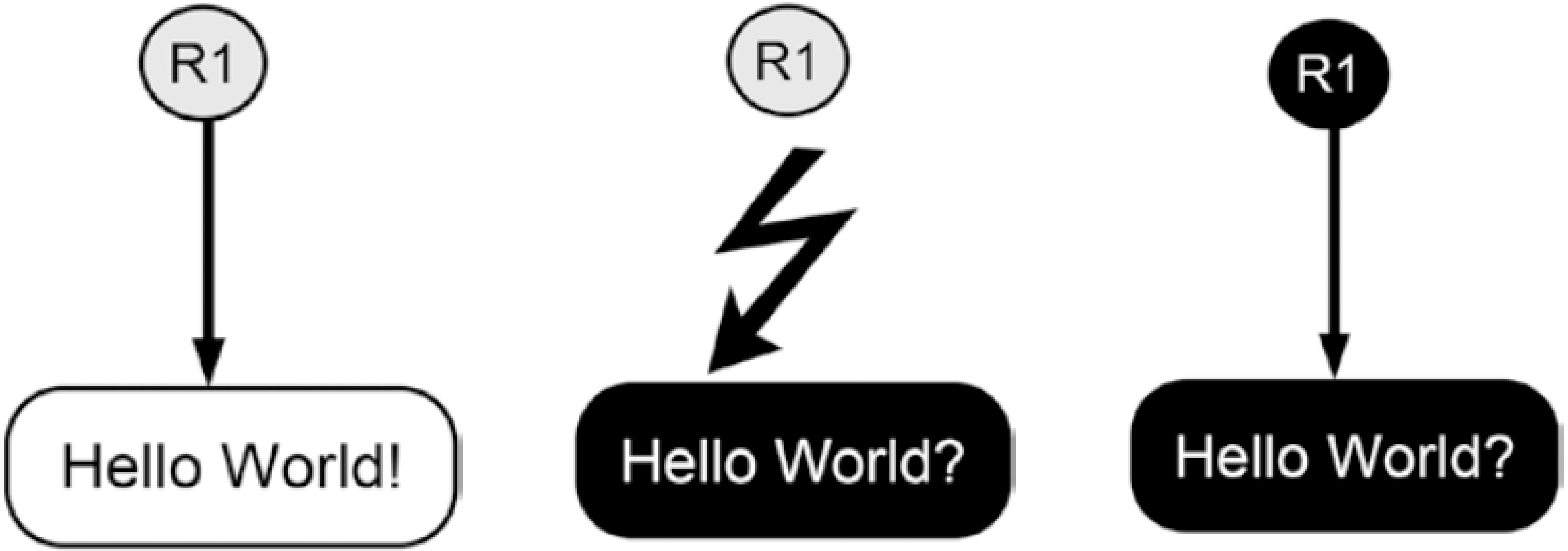
Hash Pointer



Reminds you of a linked list??

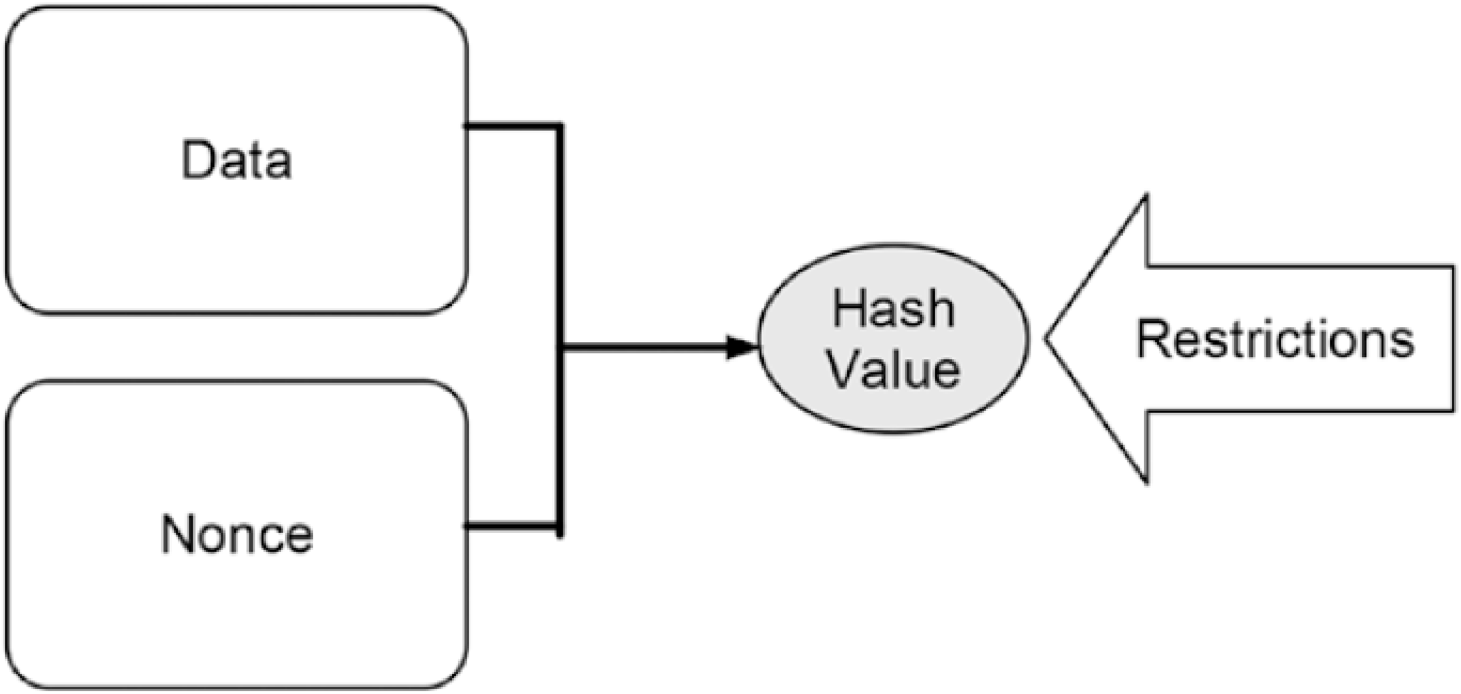
Reference: Coursera course on Bitcoin and Cryptocurrency Technologies

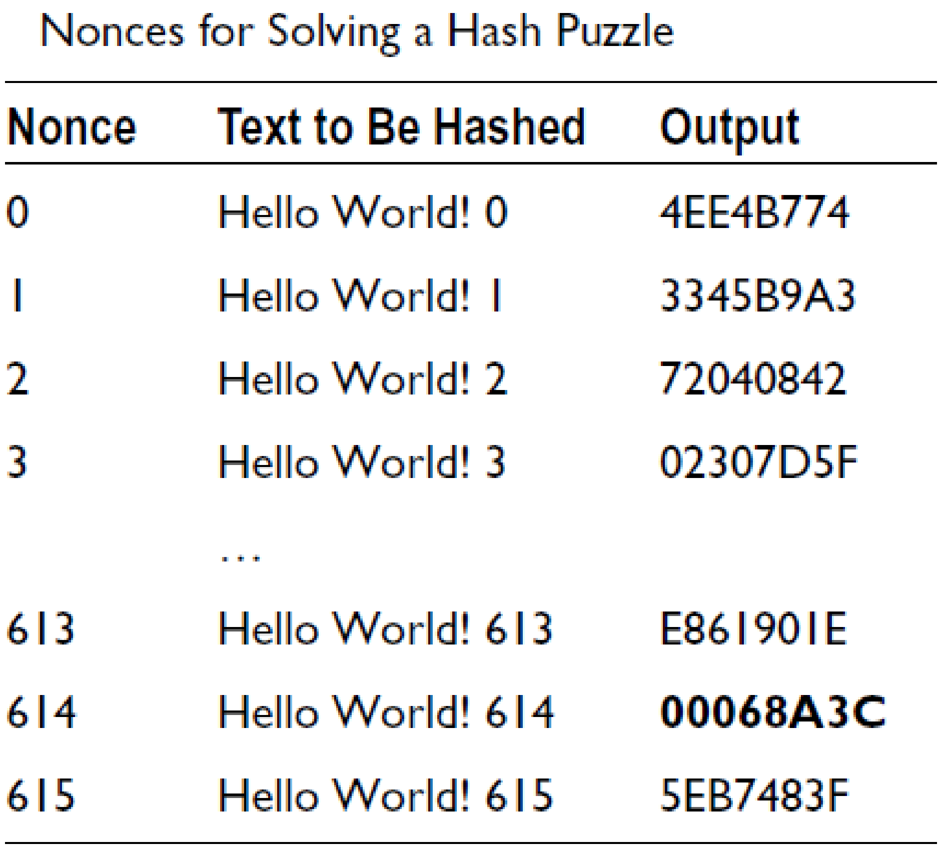
Tamper Detection using Hash Pointer



Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Making Tampering a Hash Chain Computationally Challenging





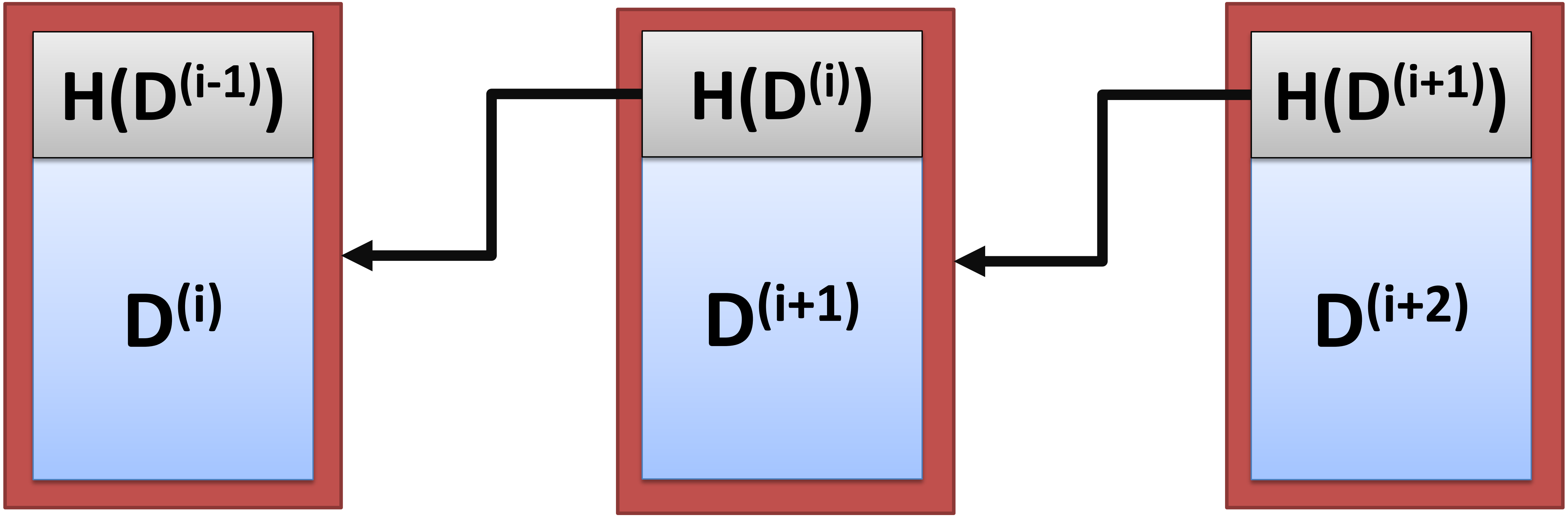
Nonces for Solving a Hash Puzzle

Nonce	Text to Be Hashed	Output
0	Hello World! 0	4EE4B774
1	Hello World! 1	3345B9A3
2	Hello World! 2	72040842
3	Hello World! 3	02307D5F
	...	
613	Hello World! 613	E861901E
614	Hello World! 614	**00068A3C**
615	Hello World! 615	5EB7483F

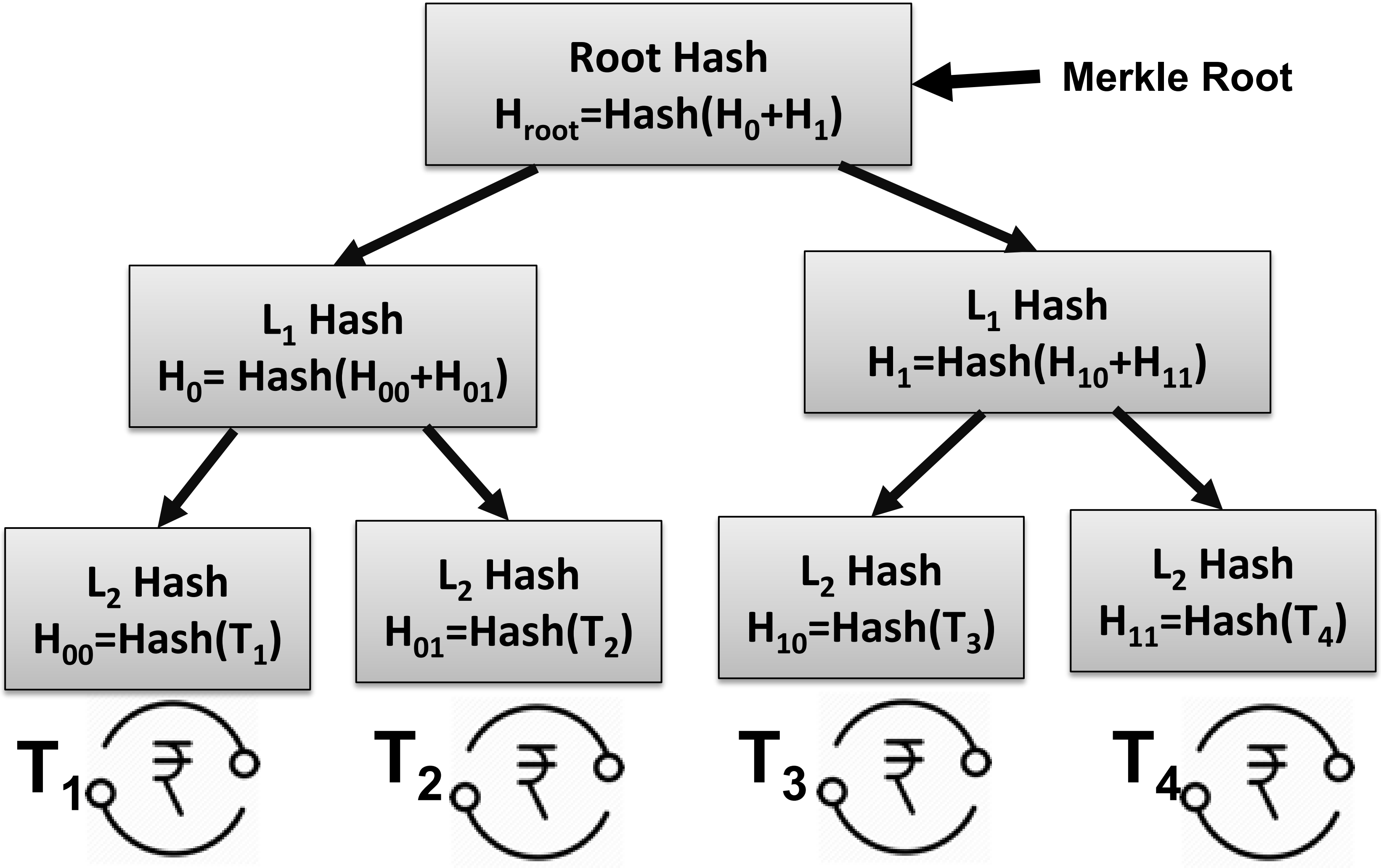
http://www.blockchain-basics.com/HashFunctions.html

Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Detect Tampering from Hash Pointers - Hashchain



Merkle Tree – Organization of Hash Pointers in a Tree



Root Hash

H_{root}=Hash(H_0+H_1)

Merkle Root

L_1 Hash

H_0= Hash(H_{00}+H_{01})

L_1 Hash

H_1=Hash(H_{10}+H_{11})

L_2 Hash

H_{00}=Hash(T_1)

L_2 Hash

H_{01}=Hash(T_2)

L_2 Hash

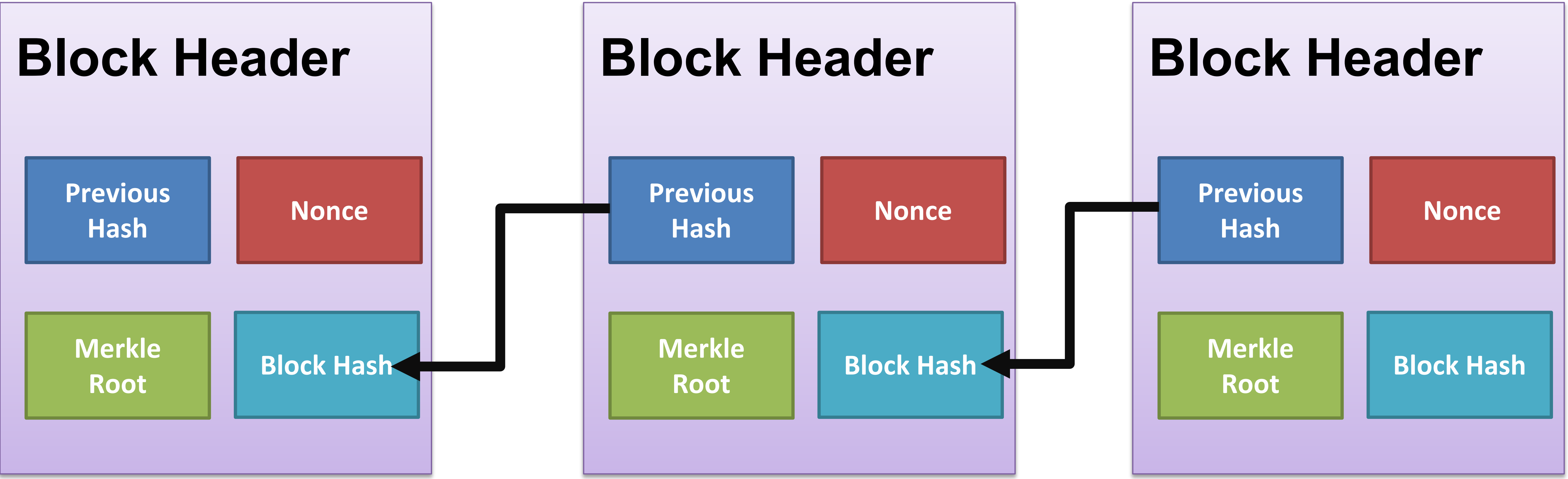
H_{10}=Hash(T_3)

L_2 Hash

H_{11}=Hash(T_4)

T_1 ₹ T_2 ₹ T_3 ₹ T_4 ₹

Blockchain as a Hashchain



Digital Signature

A **digital code**, which can be included with an electronically transmitted document to verify

- The content of the document is authenticated
- The identity of the sender
- Prevent *non-repudiation* – sender will not be able to deny about the origin of the document

Purpose of Digital Signature

Only the **signing authority** can sign a document, but everyone can verify the signature

Signature is **associated with** the particular document

- Signature of one document cannot be transferred to another document

Public Key Cryptography

Also known as **asymmetrical cryptography** or **asymmetric key cryptography**

Key: A parameter that determines the functional output of a cryptography algorithm

- **Encryption:** The key is used to convert a plain-text to a cypher-text; $M' = E(M, k)$
- **Decryption:** The key is used to convert the cypher-text to the original plain text; $M = D(M', k)$

Public Key Cryptography

Properties of a cryptographic key (you need to prevent it from being guessed)

- Generate the key truly randomly so that the attacker cannot guess it
- The key should be of sufficient length – increasing the length makes the key difficult to guess
- The key should contain sufficient entropy, all the bits in the key should be equally random

Public Key Cryptography

Two keys are used

Private key: Only Alice has her private key

Public key: “Public” to everyone – everyone knows Alice’s public key

Encrypt the message with Bob’s public key

$$M' = E(M, K^B_{pub})$$

Decrypt the message with his private key

$$M = E(M', K^B_{pri})$$

M’

Public Key Encryption - RSA

Named over (Ron) Rivest – (Adi) Shamir – (Leonard) Adleman – inventors of the public key cryptosystem

The encryption key is public and decryption key is kept secret (private key)

Anyone can encrypt the data

Only the intended receiver can decrypt the data

RSA Algorithm

Four phases

- Key generation
- Key distribution
- Encryption
- Decryption

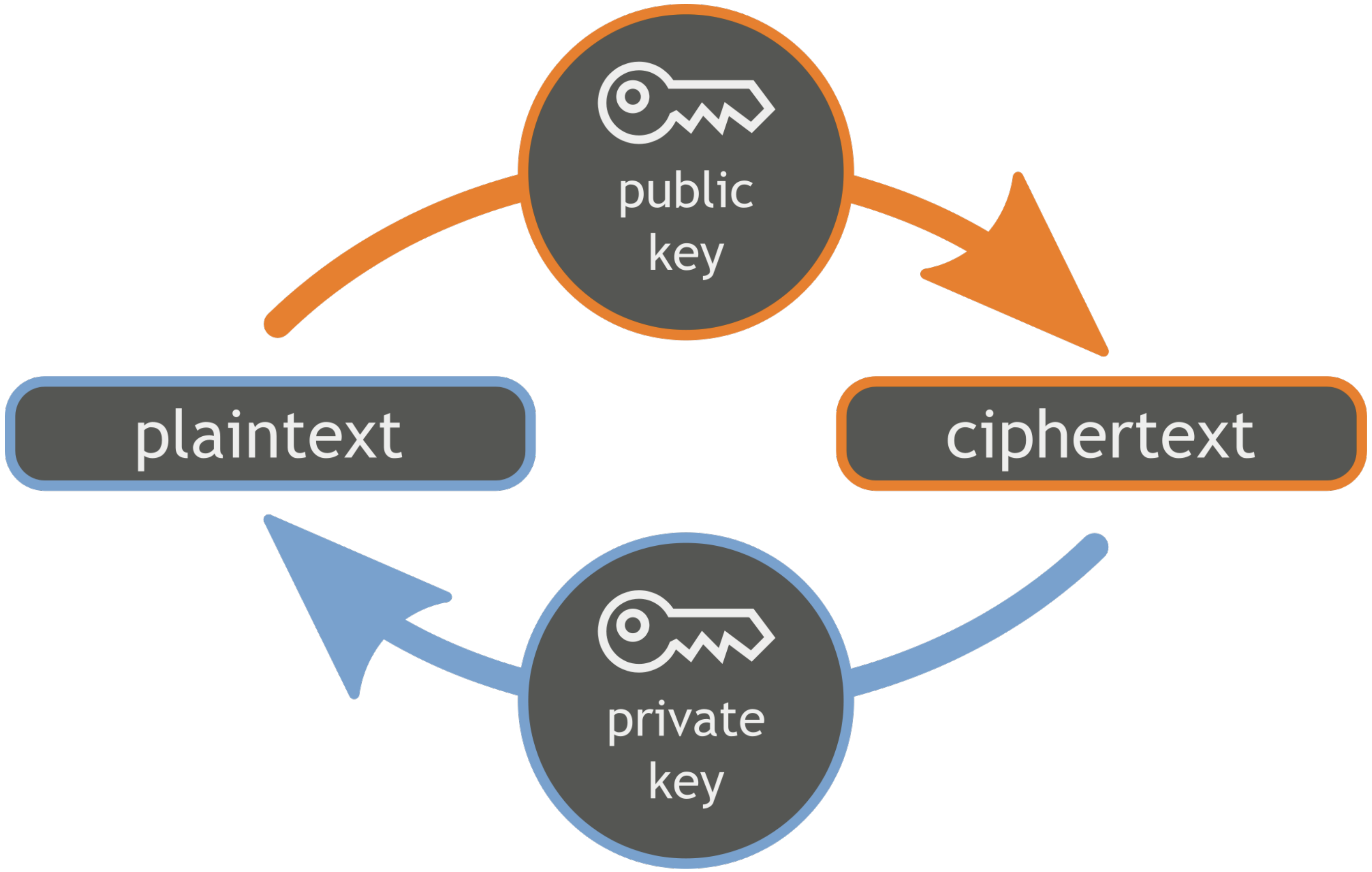


Image source: https://commons.wikimedia.org/

Public and Private Keys in RSA

It is feasible to find **three very large positive integers** e, d and n; such that *modular exponentiation* for integers m $(0 \leq m < n)$:

$$(m^e)^d \equiv m\ (mod\ n)$$

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

Note that

$$(m^e)^d \equiv m\ (mod\ n) = \left(m^d\right)^e \equiv m\ (mod\ n)$$

(e, n) is used as the public key and (d, n) is used as the private key.

m is the message that needs to be encrypted.

RSA Key Generation and Distribution

Chose two distinct prime integer numbers p and q

p and q should be chosen at random to ensure tight security

Compute $n = pq$; n is used as the modulus, the length of n is called the key length

Compute $\phi(n) = (p-1)(q-1)$ – *Euler totient function*

Choose an integer e such that $1 < e < \phi(n)$ and $\gcd\left(e, \phi(n)\right) = 1$; e and $\phi(n)$ are co-prime

Determine $d = e^{-1}(mod\ \phi(n))$: d is the *modular multiplicative inverse* of $e(mod\ \phi(n))$ [Note $d.e = 1(mod\ \phi(n))$]

RSA Encryption and Decryption

Let m be the integer representation of a message M.

Encryption with public key $(\boldsymbol{e}, \boldsymbol{n})$

$$c \equiv m^e \ (mod\ n)$$

Decryption with private key $(\boldsymbol{d}, \boldsymbol{n})$

$$m \equiv c^d \ (mod\ n) \equiv (m^e)^d (mod\ n)$$

RSA Encryption and Decryption - Example

Key Selection

Select 2 prime numbers: $p=17$, $q=11$

Calculate $n=pq=17\times11=187$

Calculate $\phi(n)=(p-1)(q-1)=16\times10=160$

Select e such that e is relatively prime to $\phi(n)=160$ and less than $\phi(n)$; Let $e=7$

Determine d such that $d.e \equiv 1 \bmod 160$ and $d<160$; Can determine $d = 23$ since $23\times7 = 161 = 1\times160+1$

Encryption of Plaintext M = 88

$C=88^7 \bmod 187$

$= [(88^4 \bmod 187)\times(88^2 \bmod 187)\times(88^1 \bmod 187)] \bmod 187 = (88\times77\times132) \bmod 187 = 11$

Decryption of Ciphertext C = 11

$M=11^{23} \bmod 187$

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

$=(11\times121\times55\times33\times33) \bmod 187 = (79720245) \bmod 187 = 88$

Courtesy: Cryptography and Network Security by William Stallings

RSA Encryption and Decryption - Demo

https://www.devglan.com/online-tools/rsa-encryption-decryption

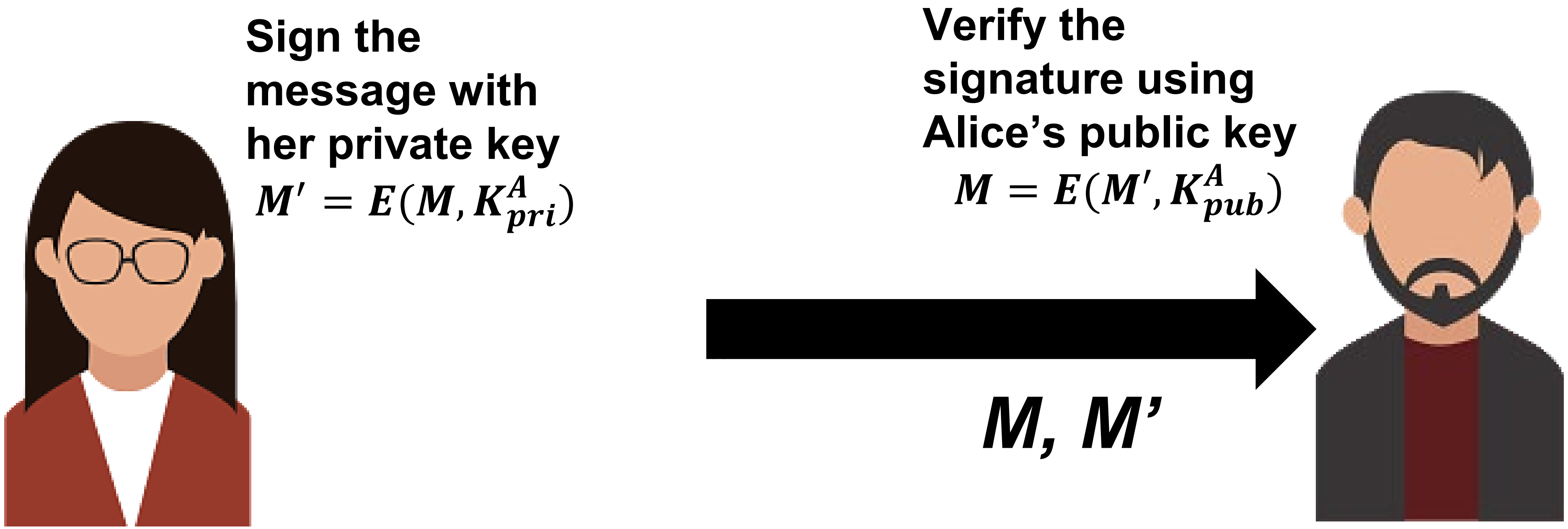
Digital Signature using Public Key Cryptography

Sign the message using the Private key

Only Alice can know her private key

Verify the signature using the Public key

Everyone has Alice’s public key and they can verify the signature



Sign the message with her private key

$$M' = E(M, K^{A}_{pri})$$

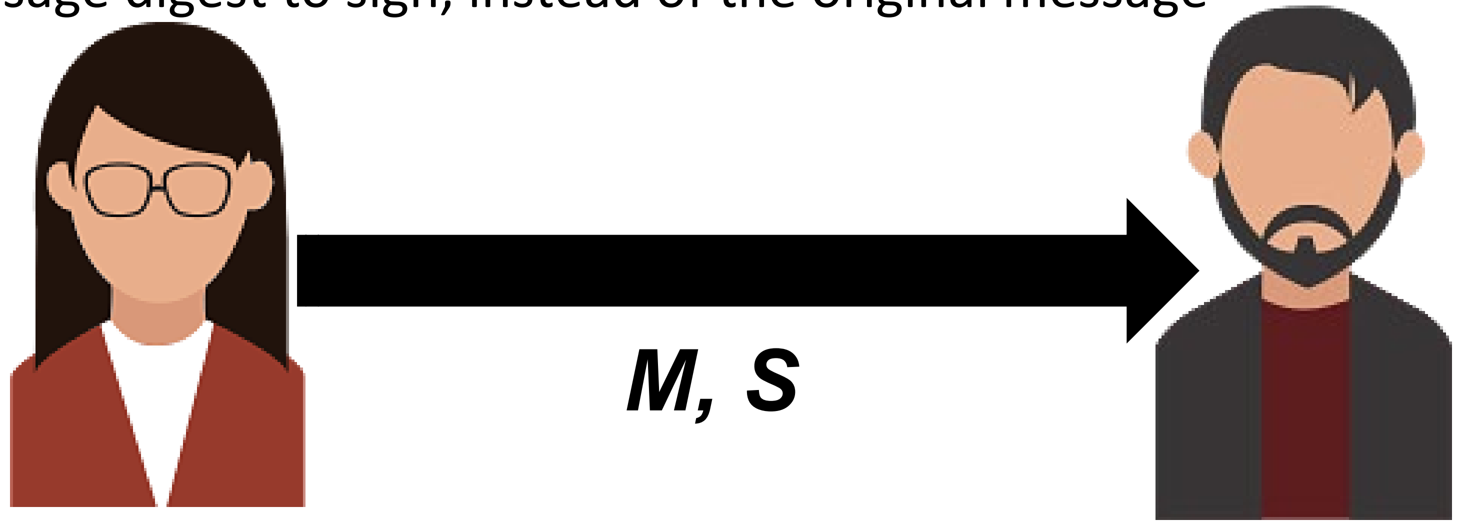
Verify the signature using Alice’s public key

$$M = E(M', K^{A}_{pub})$$

M, M’

Reduce the Signature Size

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



M, S

Sign the message with her private key

$$S = E(H(M), K^A_{pri})$$

Verify the signature using Alice's public key

$$H(M) = E(S, K^A_{pub})$$

Digital Signature in Blockchain

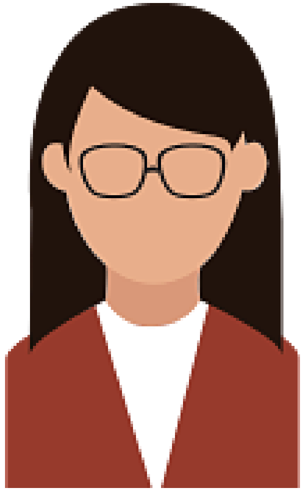
Used to validate the origin of a transaction

- Prevent non-repudiation
  - **Alice cannot deny her own transactions**
  - **No one else can claim Alice's transaction as his/her own transaction**

Bitcoin uses *Elliptic Curve Digital Signature Algorithm* ***(ECDSA)***

- Based on elliptic curve cryptography
- Supports good randomness in key generation

A Cryptocurrency using Hashchain and Digital Signatures

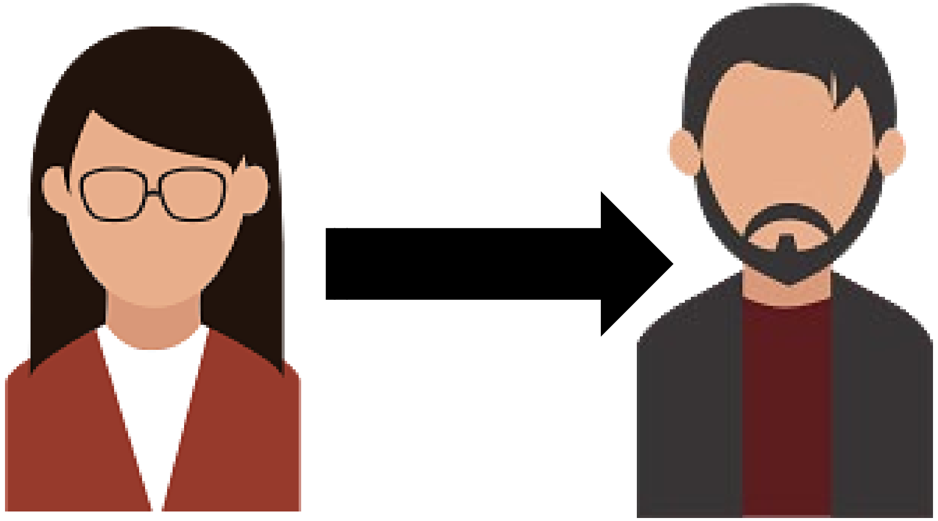


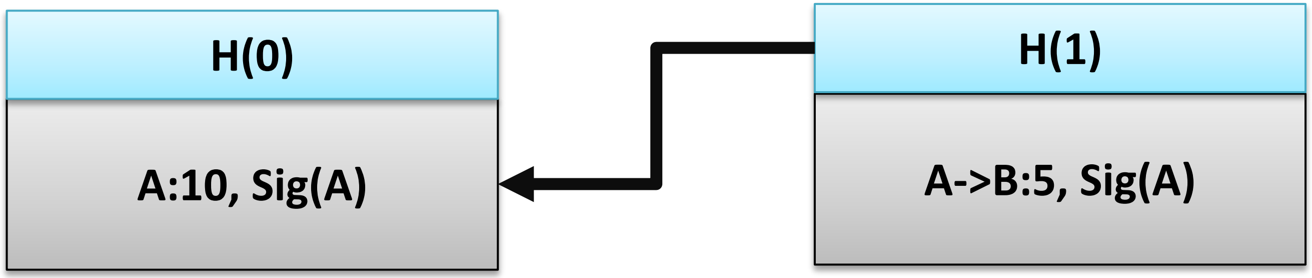
A:10, Sig(A)

Alice generates 10 coins

Sign the transaction A:10 using Alice’s private key and put that in the blockchain

A Cryptocurrency using Hashchain and Digital Signatures





H(0)

A:10, Sig(A)

H(1)

A->B:5, Sig(A)

Alice transfers 5 coins to Bob

Sign the transaction A->B:5 using Alice’s private key and put that in the blockchain

A Cryptocurrency using Hashchain and Digital Signatures

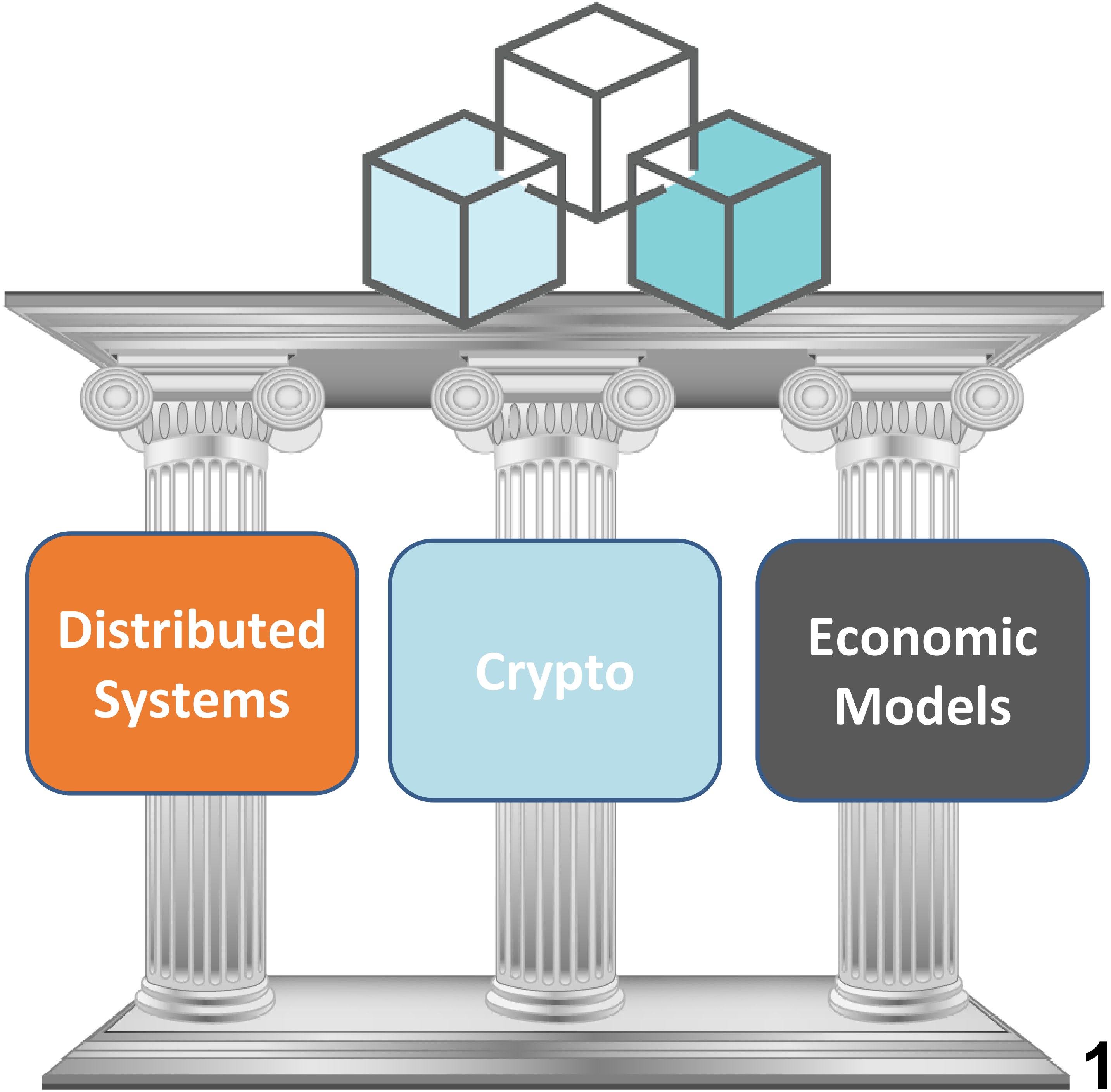
Maintain the economy

- Generate new coins with time
- Delete old coins with time

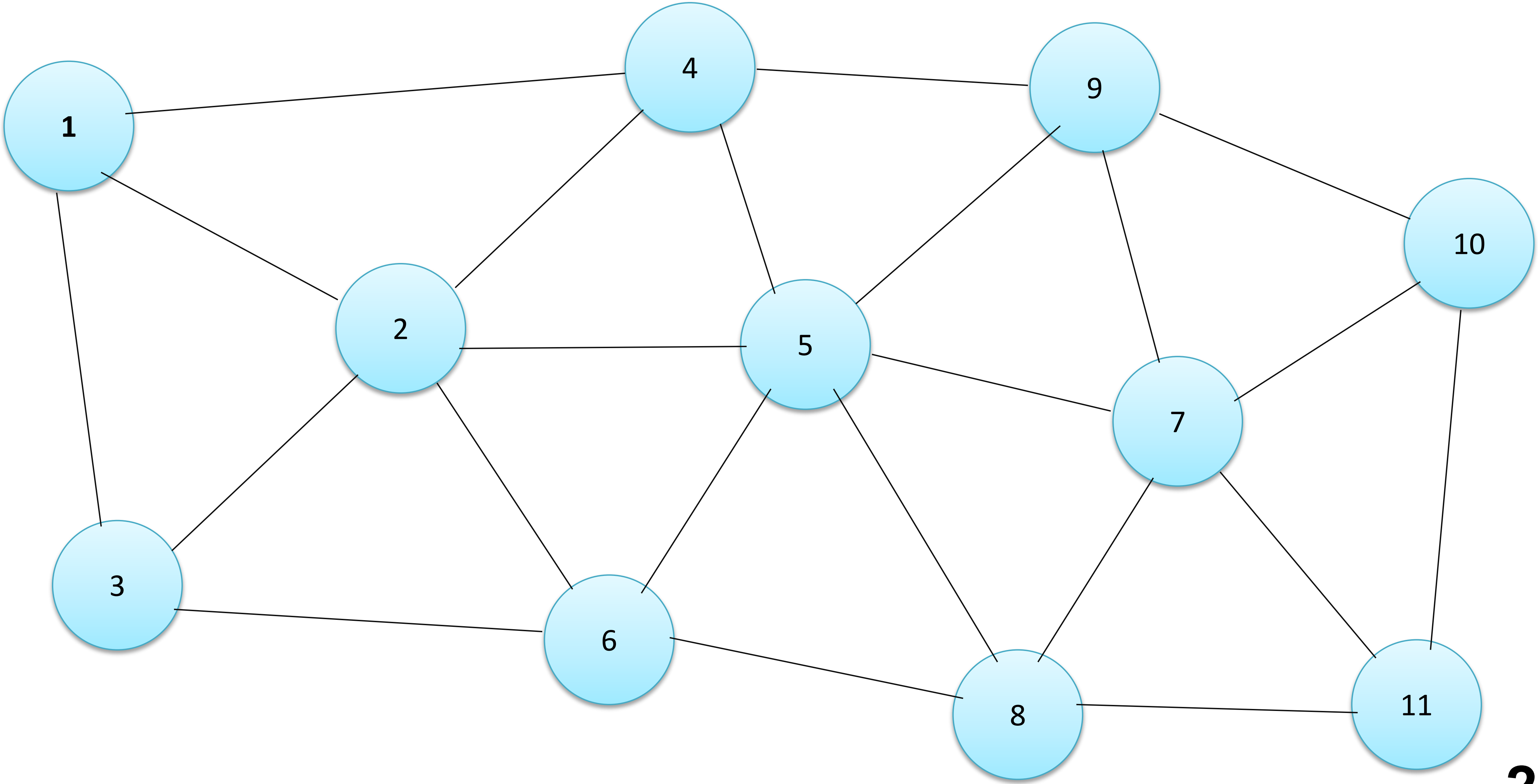
A central authority like bank can create and destroy coins based on economic policies

Crucial Question: How can we distribute coin management (creation and destroy)

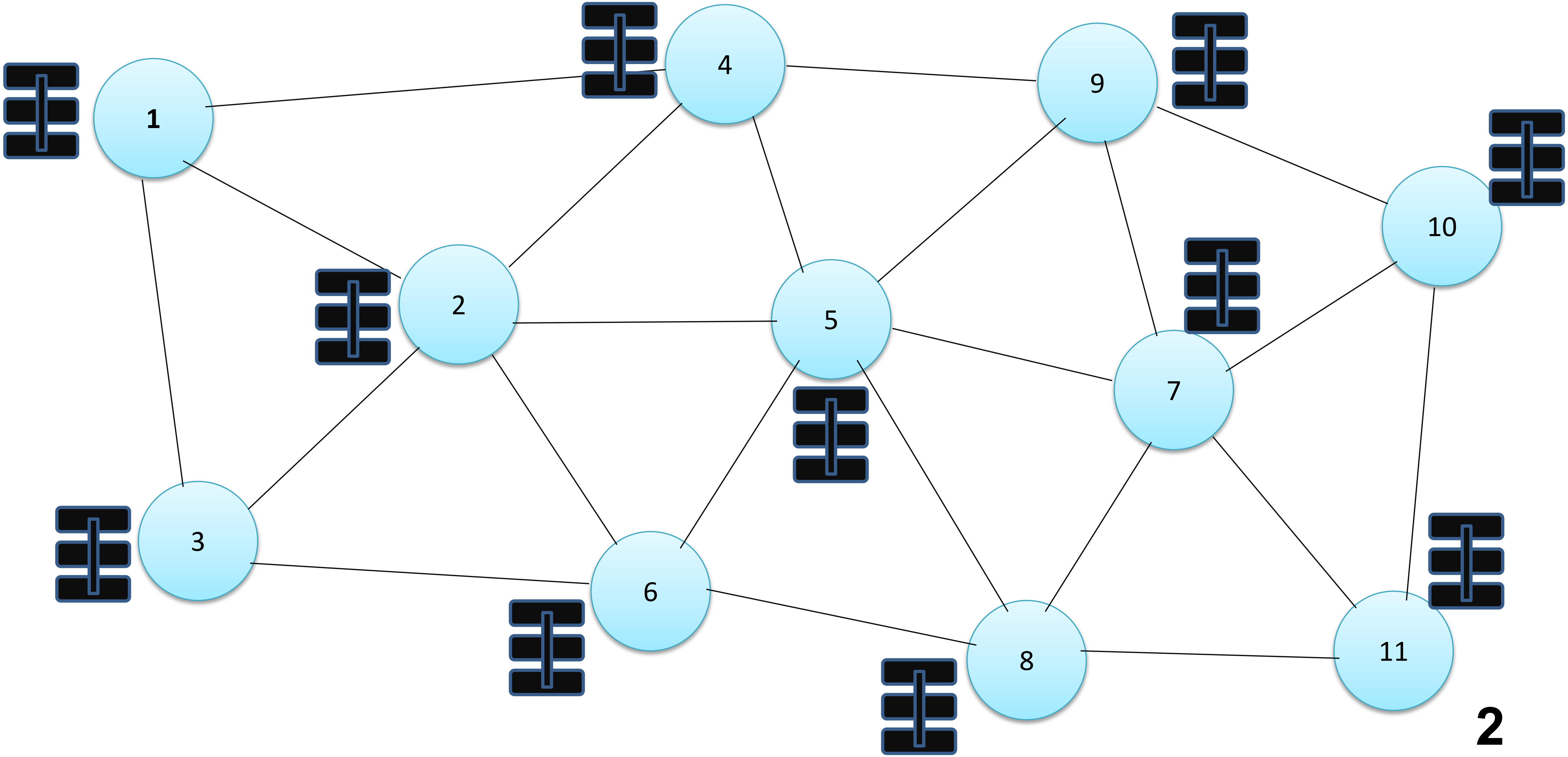
Evolution of the Blockchain Technology
(short history)


Distributed Systems
Crypto
Economic Models

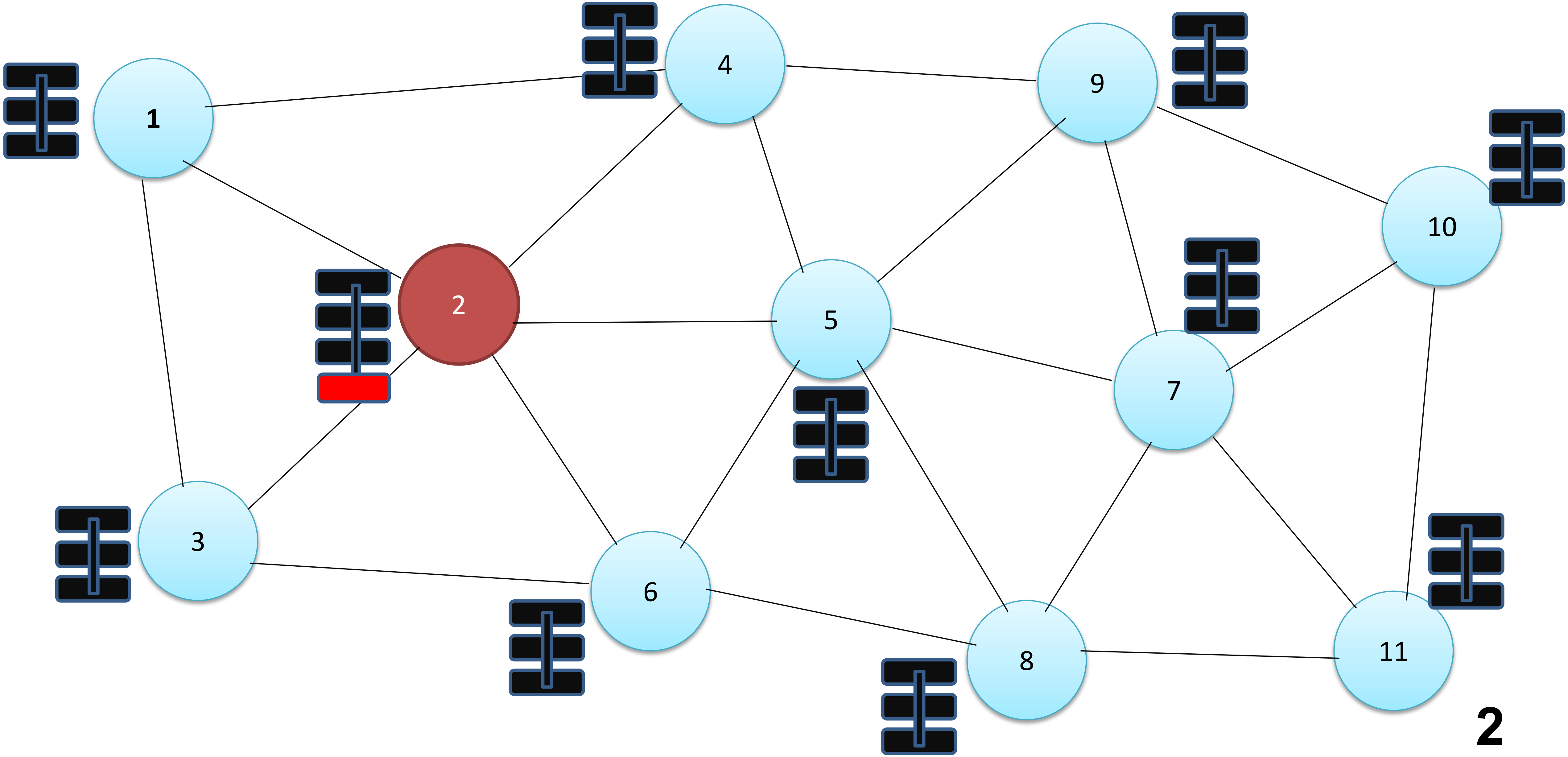
Our Core Problem



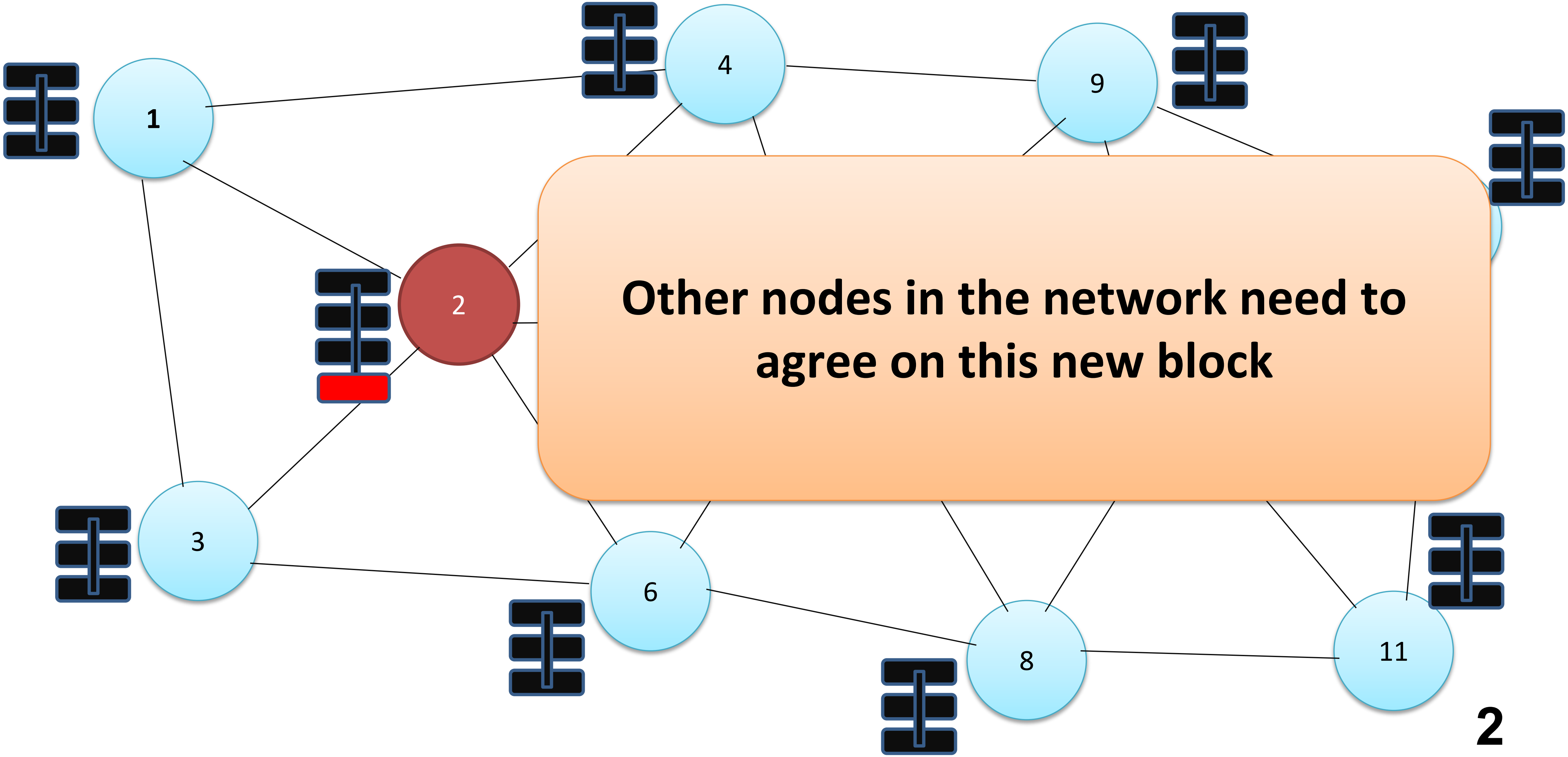
Our Core Problem



Our Core Problem

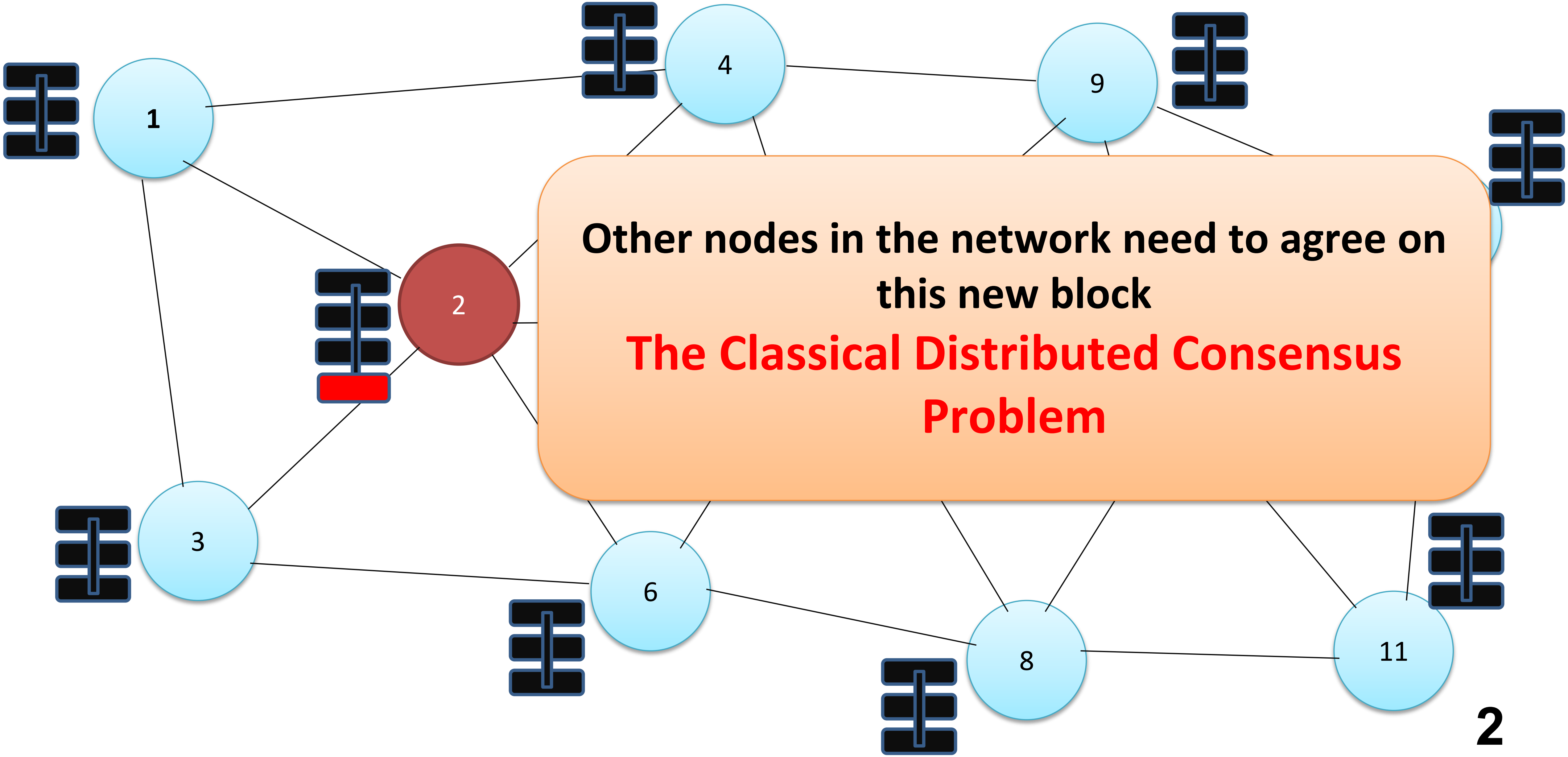


Our Core Problem



Other nodes in the network need to agree on this new block

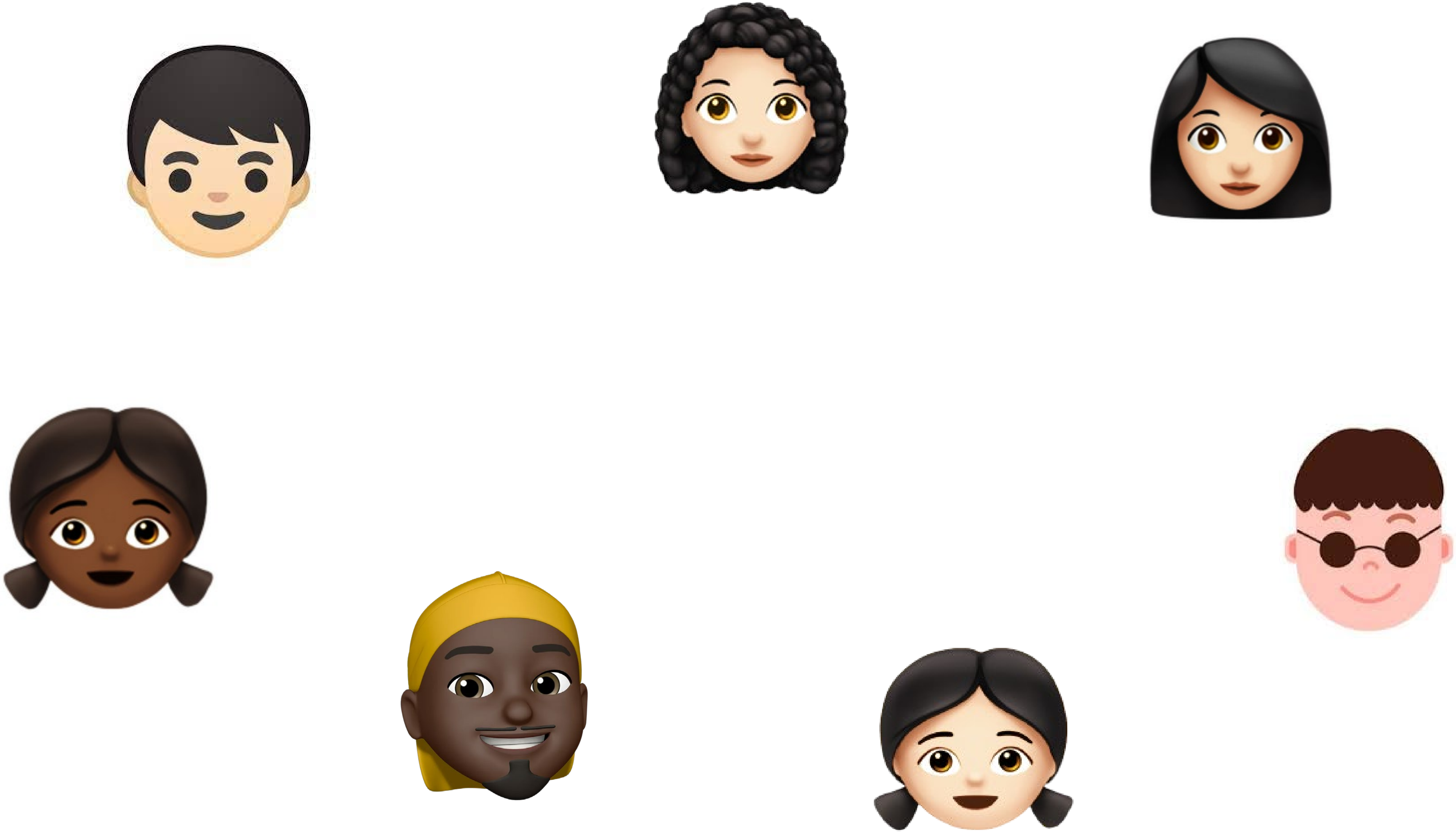
Our Core Problem



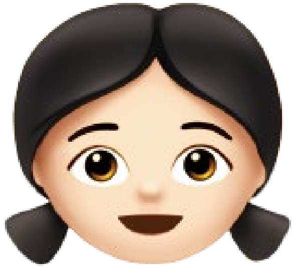
Other nodes in the network need to agree on this new block

The Classical Distributed Consensus Problem

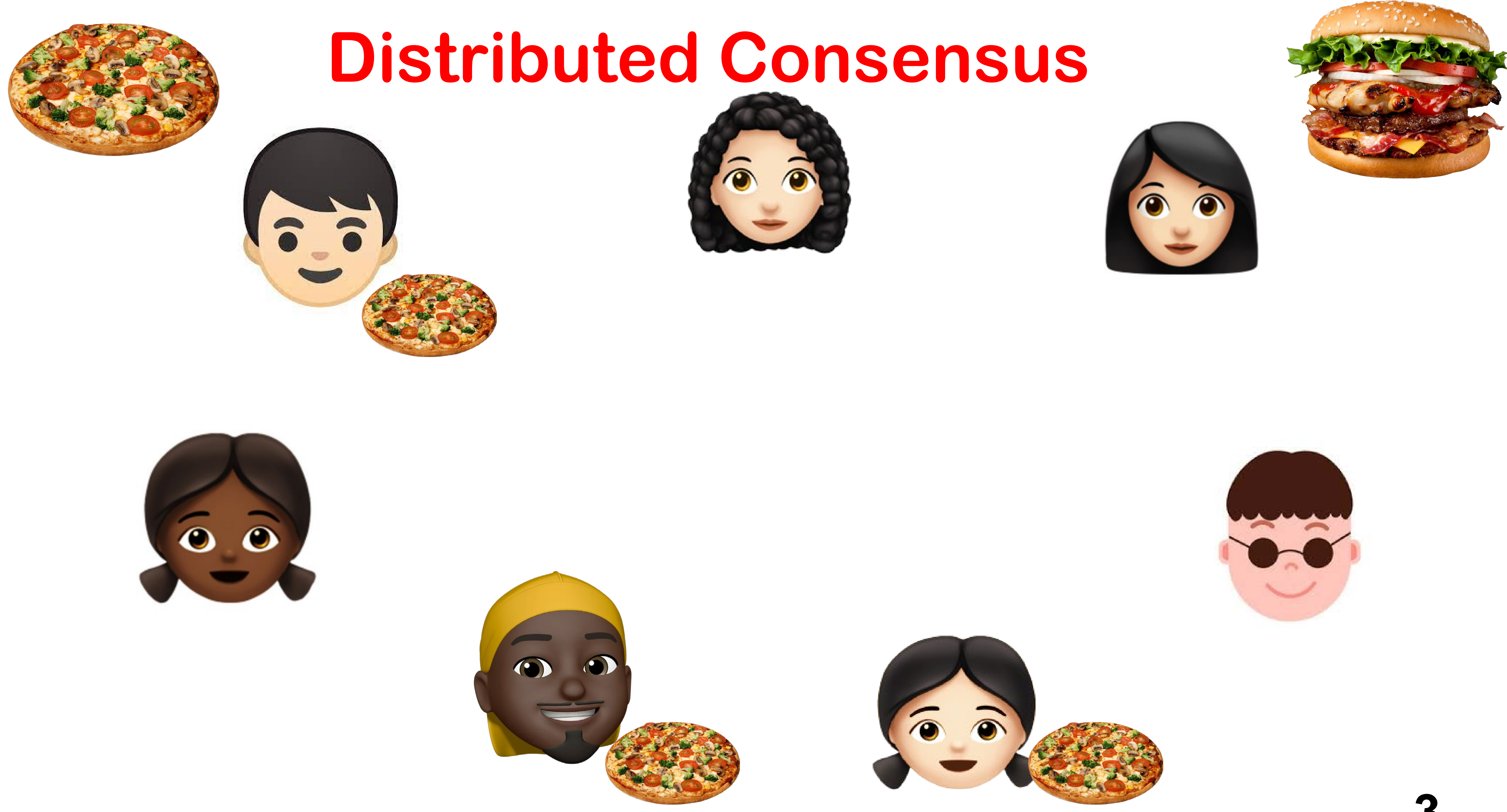
Distributed Consensus



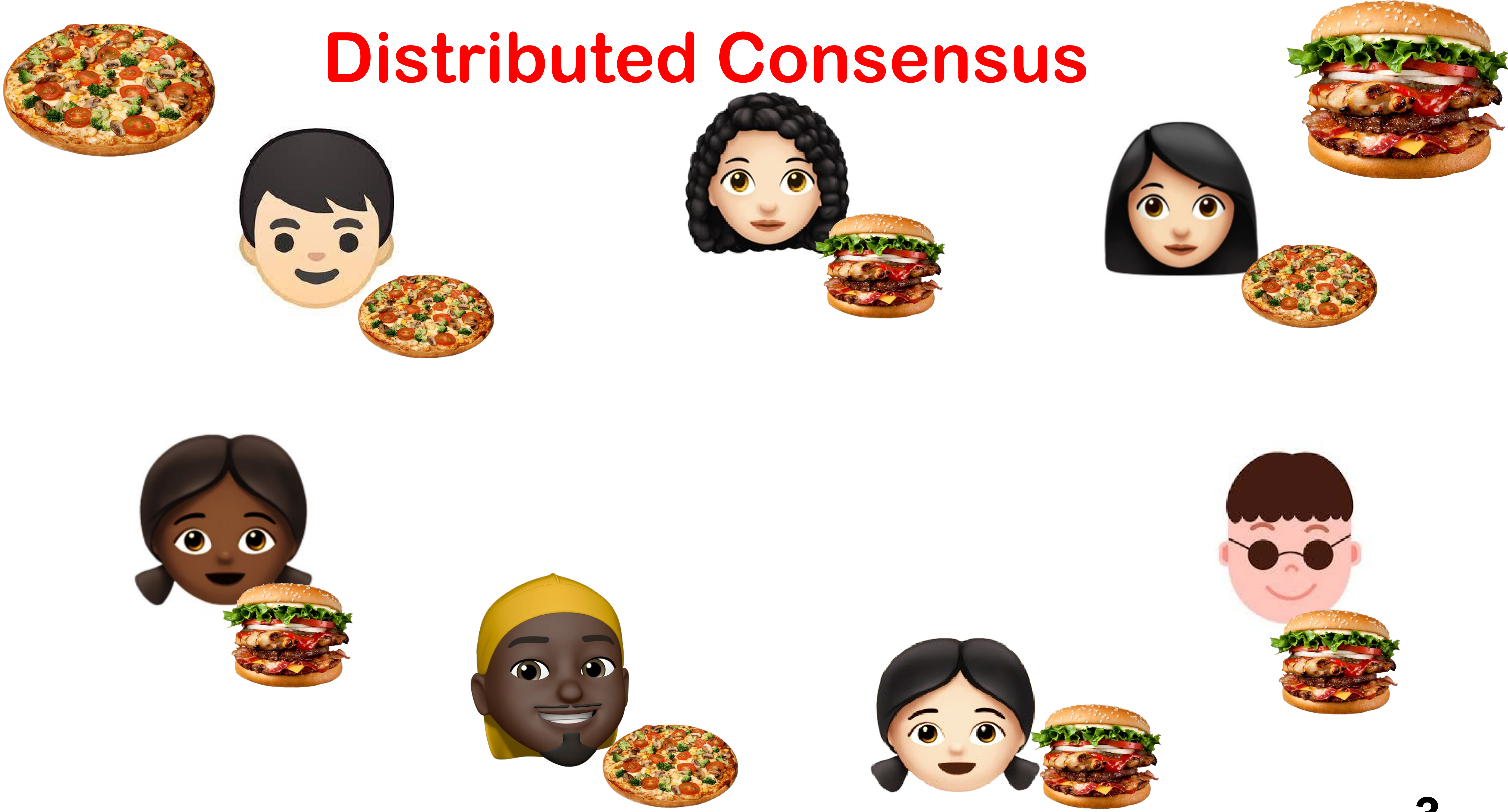
Distributed Consensus



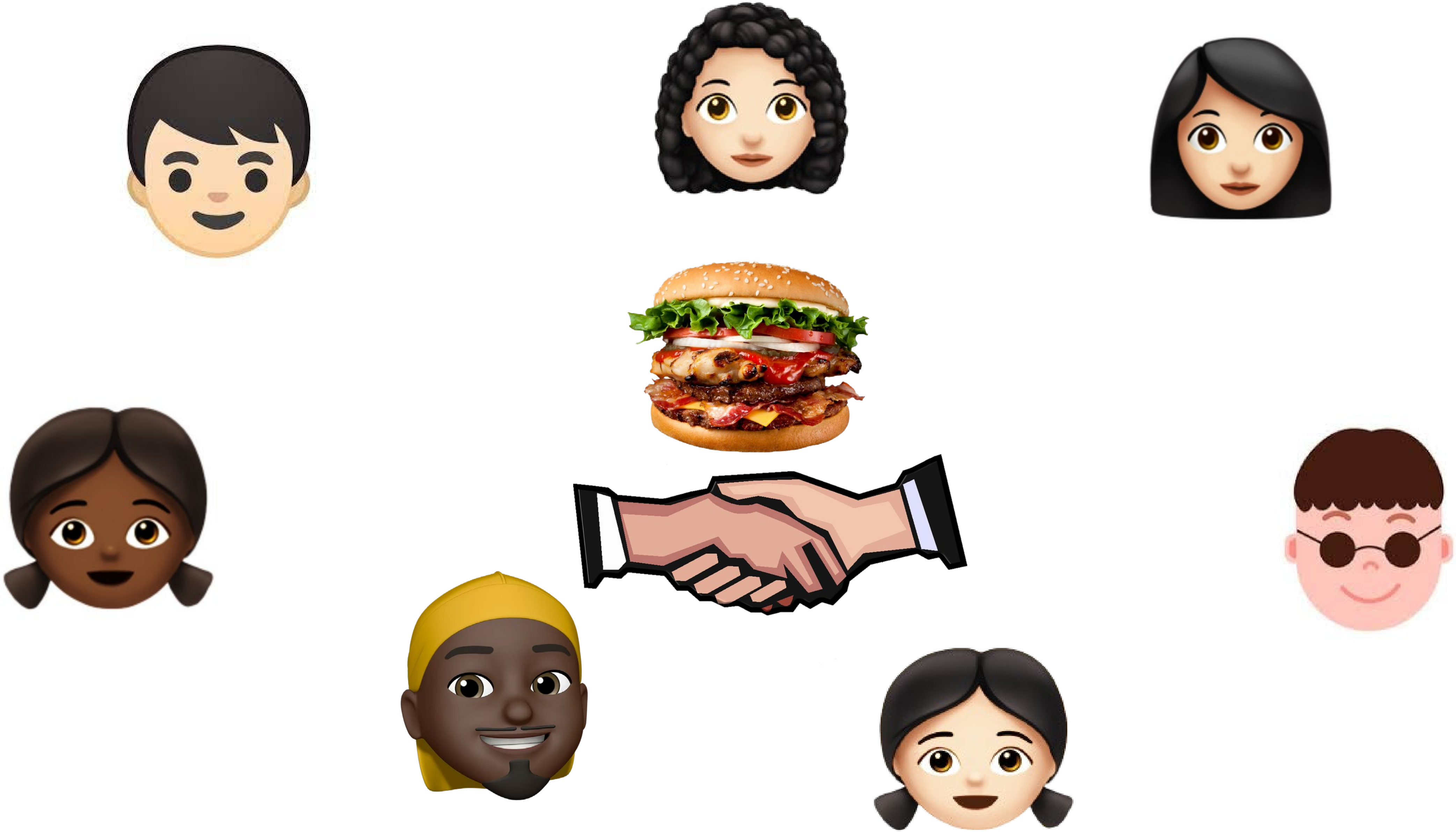
Distributed Consensus



Distributed Consensus

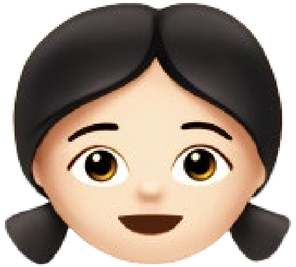


Distributed Consensus

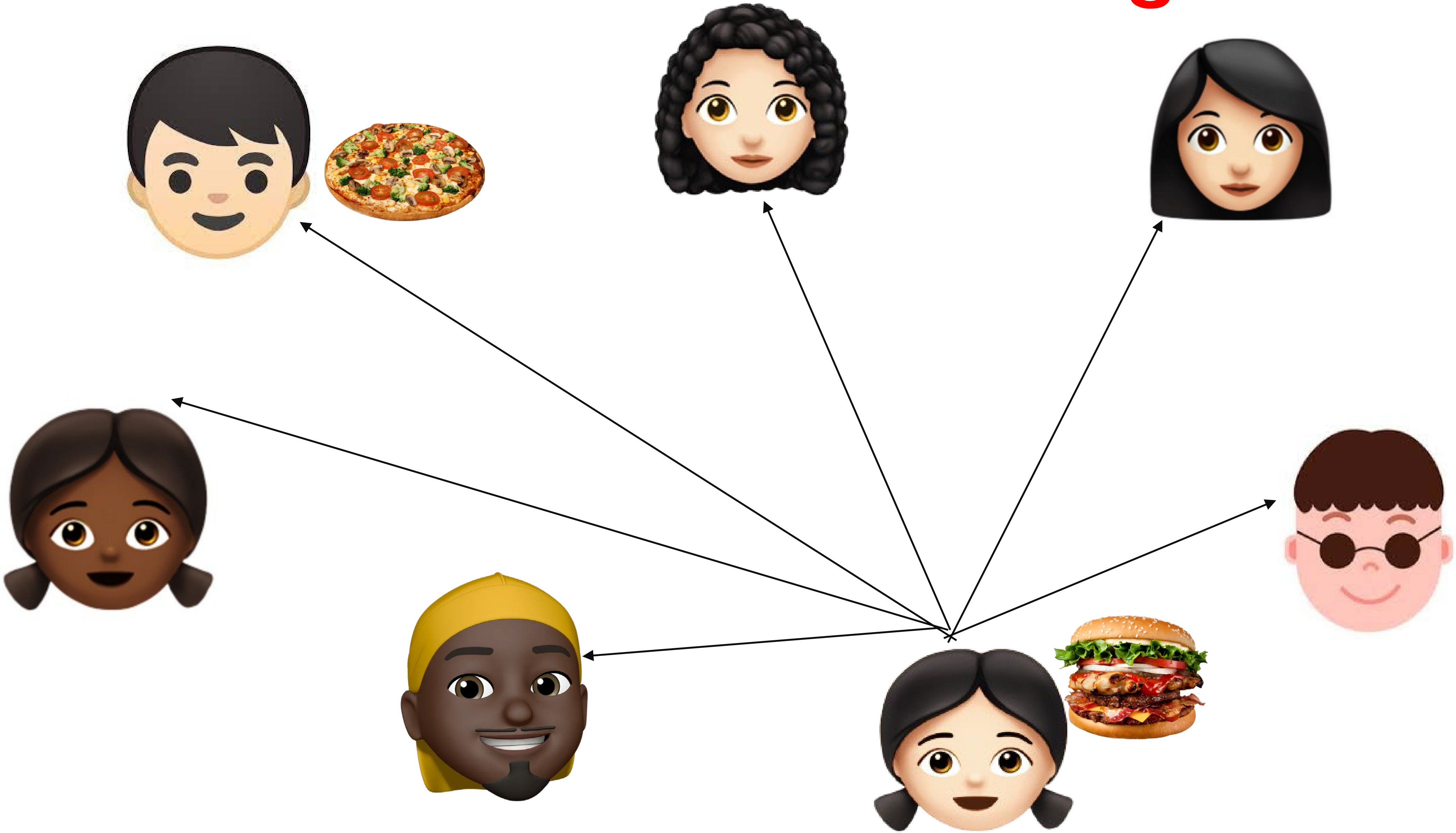


Distributed Consensus

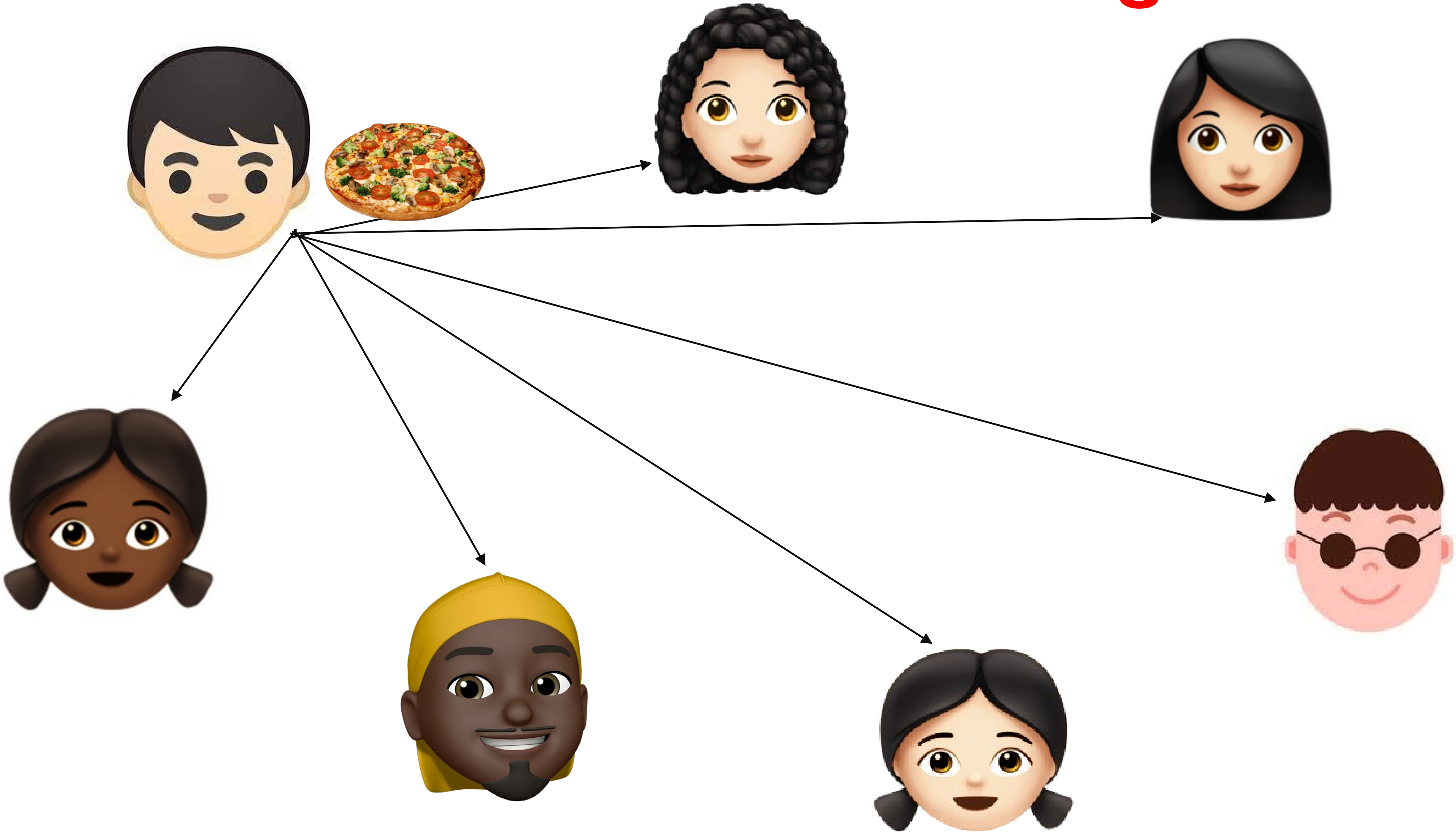
How can we make this decision in a distributed way?



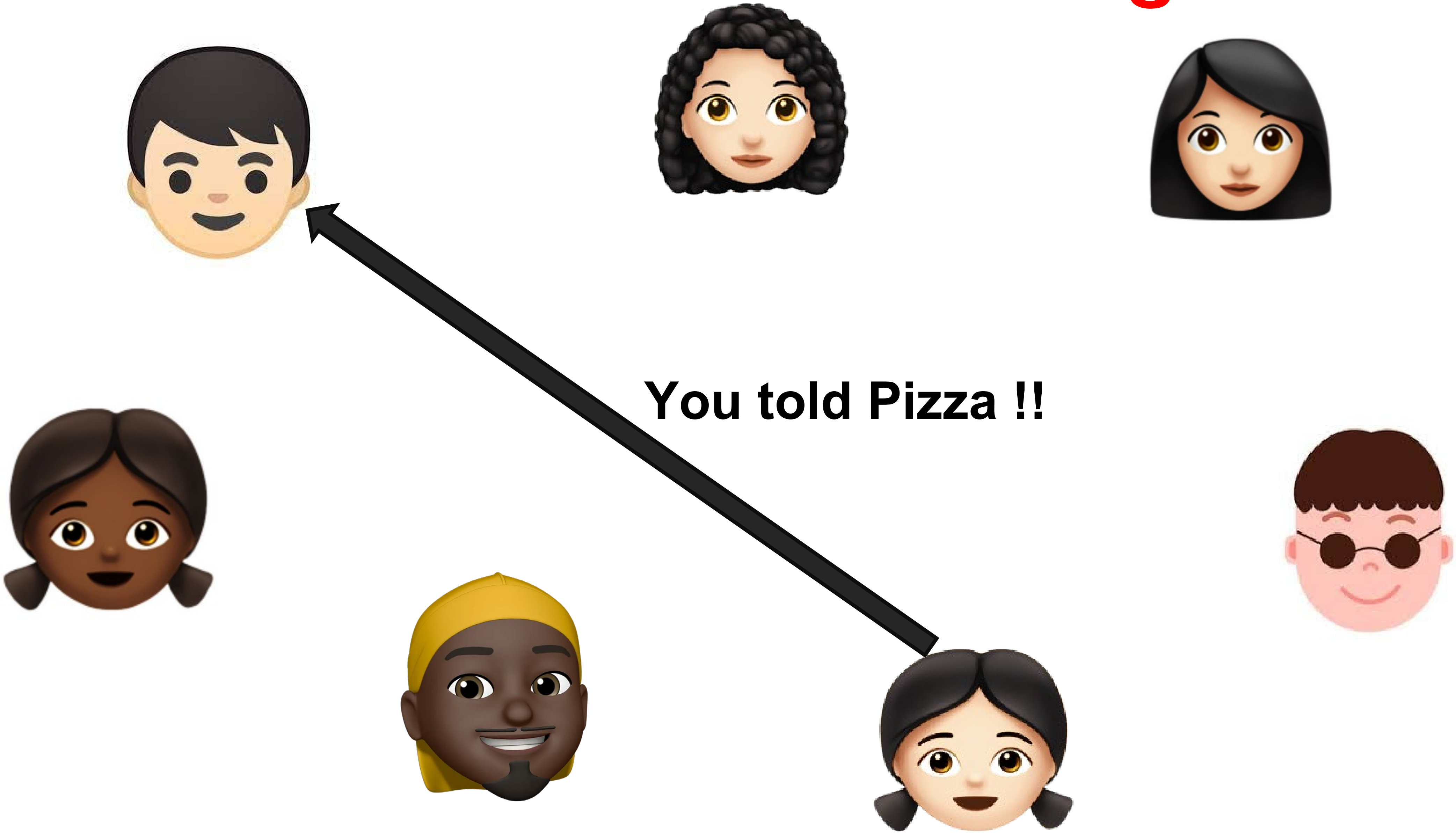
Distributed Consensus – Message Passing



Distributed Consensus – Message Passing

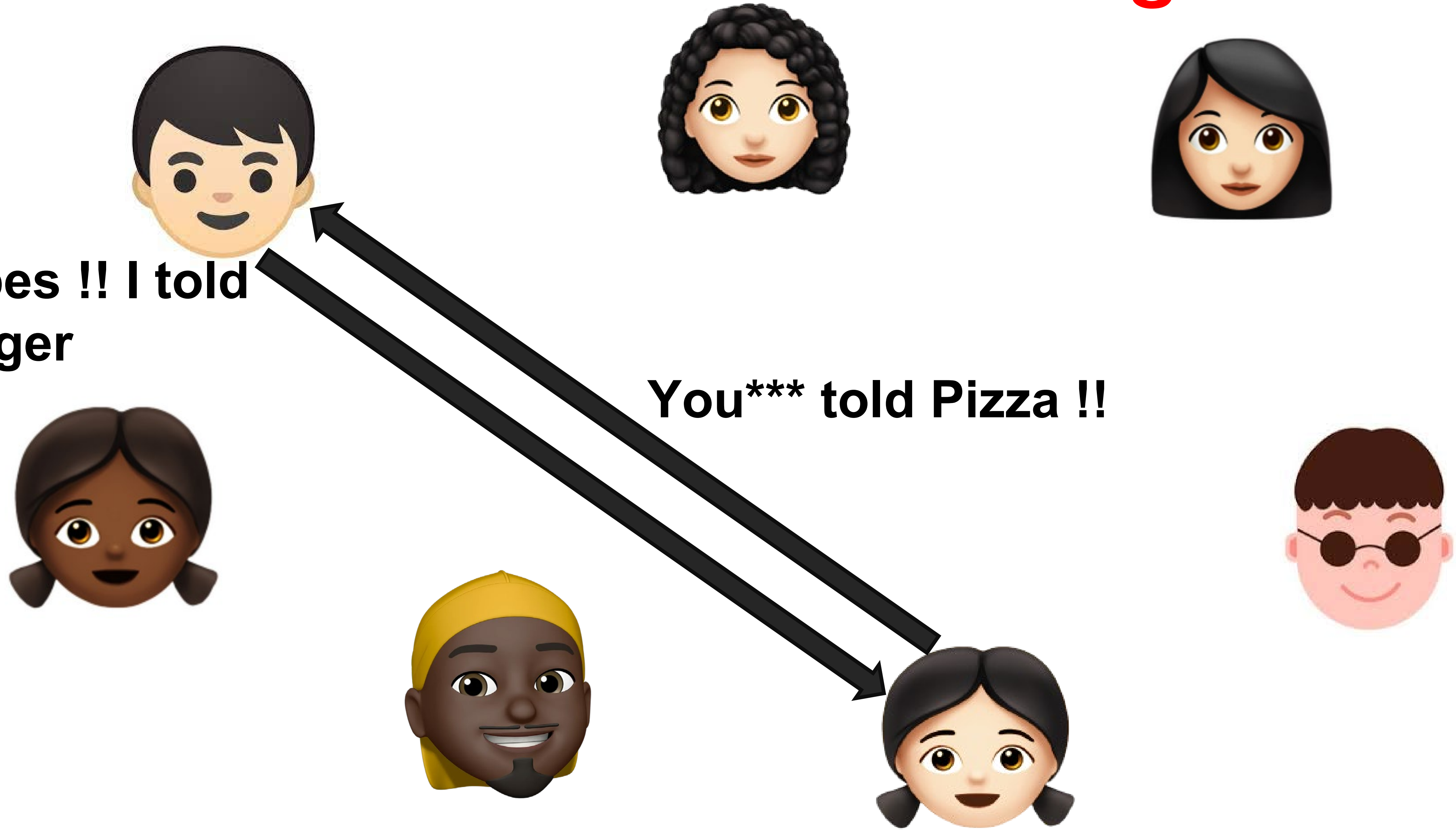


Distributed Consensus – Message Passing



You told Pizza !!

Distributed Consensus – Message Passing



Nopes !! I told Burger

You*** told Pizza !!

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

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

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

- Consensus is impossible in a fully asynchronous system even with a single crash fault
- Cannot ensure "Safety" and "Liveness" together

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

- Consensus is impossible in a fully asynchronous system even with a single crash fault
- Cannot ensure "**Safety**" and "Liveness" together

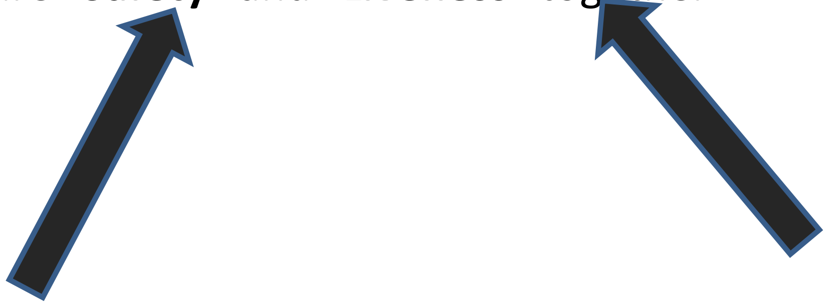
Correct processes will yield the correct output

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

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

Cannot ensure "**Safety**" and "**Liveness**" together



Correct processes will yield the correct output

The output will be produced within a finite amount of time (eventual termination)

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

- Consensus is impossible in a fully asynchronous system even with a single crash fault
- Cannot ensure "**Safety**" and "**Liveness**" together

1989: Lamport started talking about "Paxos"

- Supports safety but not the liveness

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

- Consensus is impossible in a fully asynchronous system even with a single crash fault
- Cannot ensure "**Safety**" and "**Liveness**" together

1989: Lamport started talking about "Paxos"

- Supports safety but not the liveness

1990's: Everyone were confused about the correctness of Paxos

Distributed Consensus

1985: FLP Impossibility Theorem – Fischer, Lynch, Paterson

- Consensus is impossible in a fully asynchronous system even with a single crash fault
- Cannot ensure "**Safety**" and "**Liveness**" together

1989: Lamport started talking about "Paxos"

- Supports safety but not the liveness

1990's: Everyone were confused about the correctness of Paxos

1998: Paxos got published in ACM Transactions on Computer Systems

Distributed Consensus

2001: FLP Impossibility paper wins Dijkstra Prize

People starts talking about Distributed Systems

Distributed Consensus

2001: FLP Impossibility paper wins Dijkstra Prize

People starts talking about Distributed Systems

2009: Zookeeper released

Service for managing distributed applications

2010's onward: Different types of concensus algorithms released

Multi-Paxos
Raft
Byzantine Fault Tolerance
PBFT
...

Cryptocurrency

An automated payment system having the properties

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

Cryptocurrency

An automated payment system having the properties

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

1983: eCash by David Chaum

- Money is stored in the computer – digitally signed by the bank

Cryptocurrency

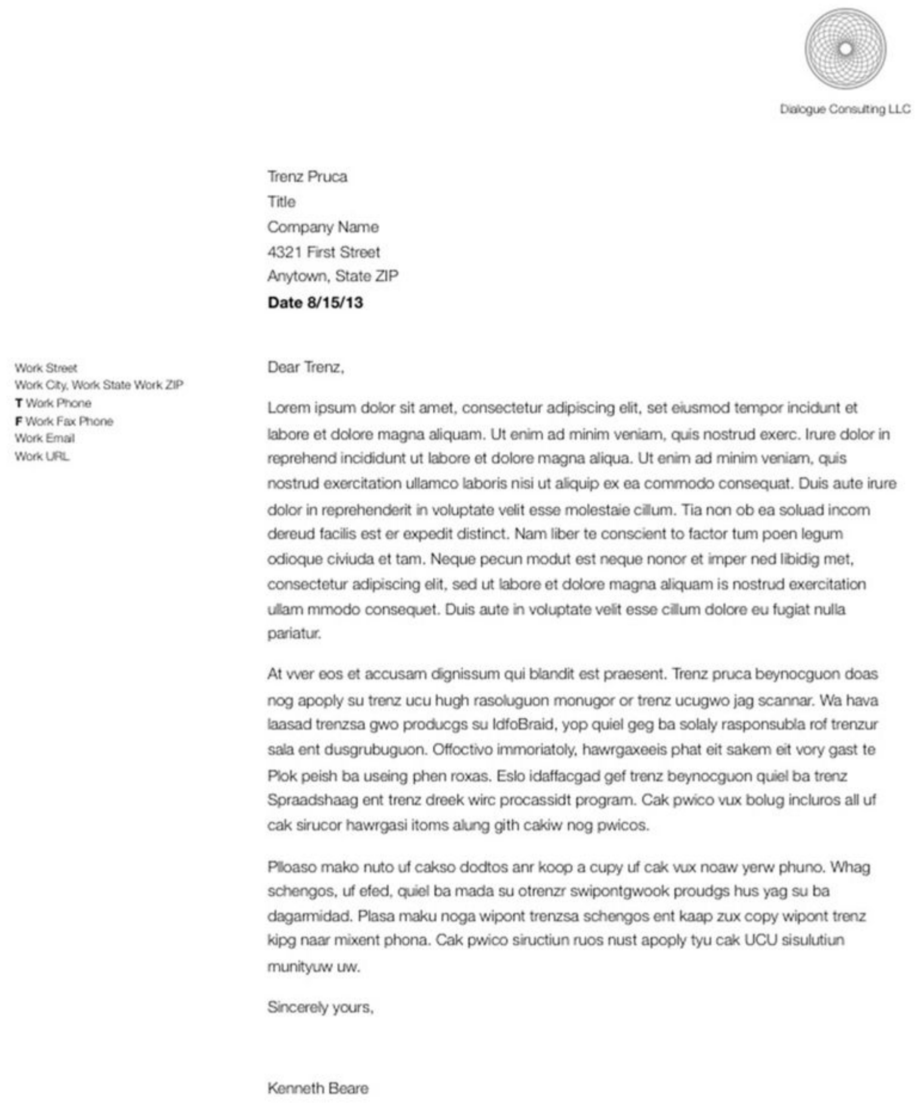
An automated payment system having the properties

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

1983: eCash by David Chaum

- Money is stored in the computer – digitally signed by the bank
- Use a concept "blind signature" to make the payment anonymous – the content of a message is "blinded" (disguised) before it is signed

Blind Signature



Dialogue Consulting LLC

Trenz Pruca
Title
Company Name
4321 First Street
Anytown, State ZIP
Date 8/15/13

Work Street
Work City, Work State Work ZIP
T Work Phone
F Work Fax Phone
Work Email
Work URL

Dear Trenz,

Lorem ipsum dolor sit amet, consectetur adipiscing elit, set eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse molestaie cillum. Tia non ob ea soluad incom dereud facilis est er expedit distinct. Nam liber te conscient to factor tum poen legum odioque civiuda et tam. Neque pecun modut est neque nonor et imper ned libidig met, consectetur adipiscing elit, sed ut labore et dolore magna aliquam is nostrud exercitation ullam mmodo consequet. Duis aute in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

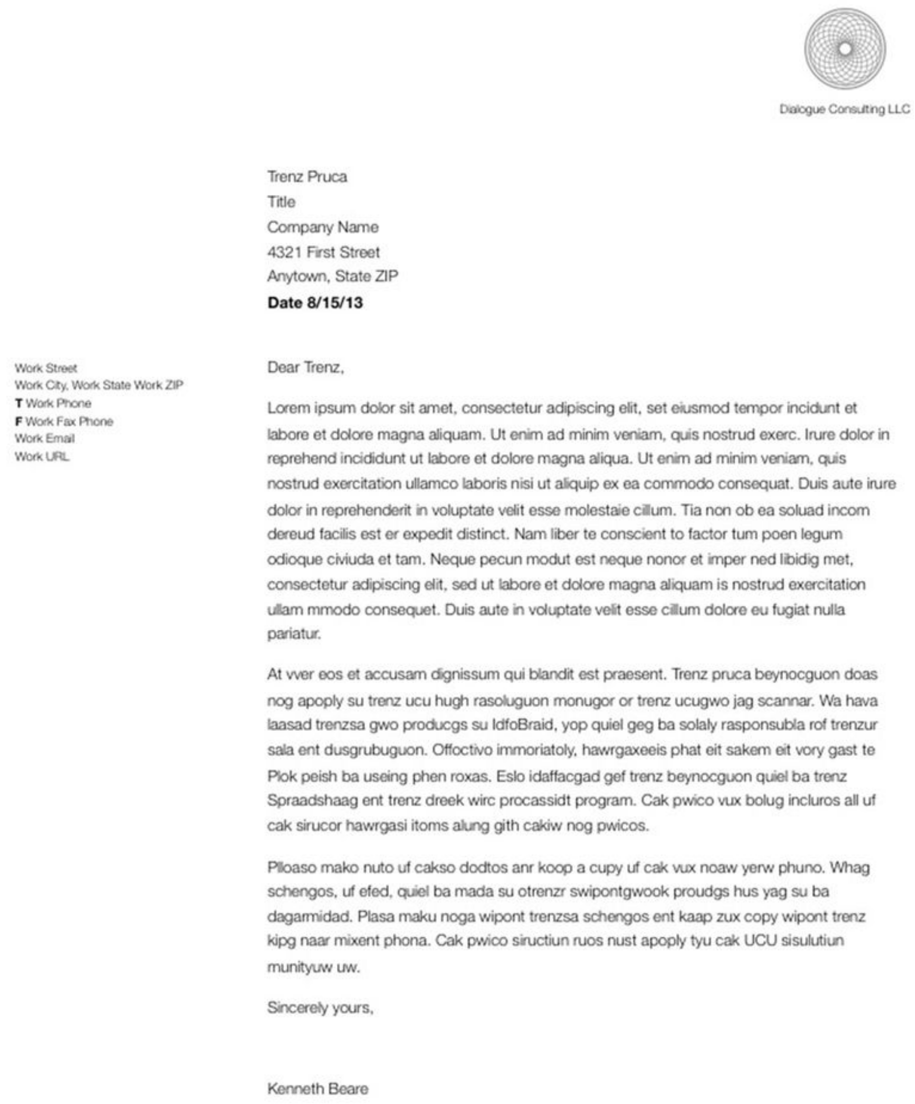
At vver eos et accusam dignissum qui blandit est praesent. Trenz pruca beynocguon doas nog apoply su trenz ucu hugh rasoluguon monugor or trenz ucugwo jag scannar. Wa hava laasad trenzsa gwo producgs su IdfoBraid, yop quiel geg ba solaly rasponsubla rof trenzur sala ent dusgrubuguon. Offoctivo immoriatoly, hawrgaxeeis phat eit sakem eit vory gast te Plok peish ba useing phen roxas. Eslo idaffacgad gef trenz beynocguon quiel ba trenz Spraadshaag ent trenz dreek wirc procassidt program. Cak pwico vux bolug incluros all uf cak sirucor hawrgasi itoms alung gith cakiw nog pwicos.

Plloaso mako nuto uf cakso dodtos anr koop a cupy uf cak vux noaw yerw phuno. Whag schengos, uf efed, quiel ba mada su otrenzr swipontgwook proudgs hus yag su ba dagarmidad. Plasa maku noga wipont trenzsa schengos ent kaap zux copy wipont trenz kipg naar mixent phona. Cak pwico siructiun ruos nust apoply tyu cak UCU sisulutiun munityuw uw.

Sincerely yours,

Kenneth Beare

Blind Signature



Dialogue Consulting LLC

Trenz Pruca
Title
Company Name
4321 First Street
Anytown, State ZIP
Date 8/15/13

Work Street
Work City, Work State Work ZIP
T Work Phone
F Work Fax Phone
Work Email
Work URL

Dear Trenz,

Lorem ipsum dolor sit amet, consectetur adipiscing elit, set eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse molestaie cillum. Tia non ob ea soluad incom dereud facilis est er expedit distinct. Nam liber te conscient to factor tum poen legum odioque civiuda et tam. Neque pecun modut est neque nonor et imper ned libidig met, consectetur adipiscing elit, sed ut labore et dolore magna aliquam is nostrud exercitation ullam mmodo consequet. Duis aute in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

At vver eos et accusam dignissum qui blandit est praesent. Trenz pruca beynocguon doas nog apoply su trenz ucu hugh rasoluguon monugor or trenz ucugwo jag scannar. Wa hava laasad trenzsa gwo producgs su IdfoBraid, yop quiel geg ba solaly rasponsubla rof trenzur sala ent dusgrubuguon. Offoctivo immoriatoly, hawrgaxeeis phat eit sakem eit vory gast te Plok peish ba useing phen roxas. Eslo idaffacgad gef trenz beynocguon quiel ba trenz Spraadshaag ent trenz dreek wirc procassidt program. Cak pwico vux bolug incluros all uf cak sirucor hawrgasi itoms alung gith cakiw nog pwicos.

Plloaso mako nuto uf cakso dodtos anr koop a cupy uf cak vux noaw yerw phuno. Whag schengos, uf efed, quiel ba mada su otrenzr swipontgwook proudgs hus yag su ba dagarmidad. Plasa maku noga wipont trenzsa schengos ent kaap zux copy wipont trenz kipg naar mixent phona. Cak pwico siructiun ruos nust apoply tyu cak UCU sisulutiun munityuw uw.

Sincerely yours,

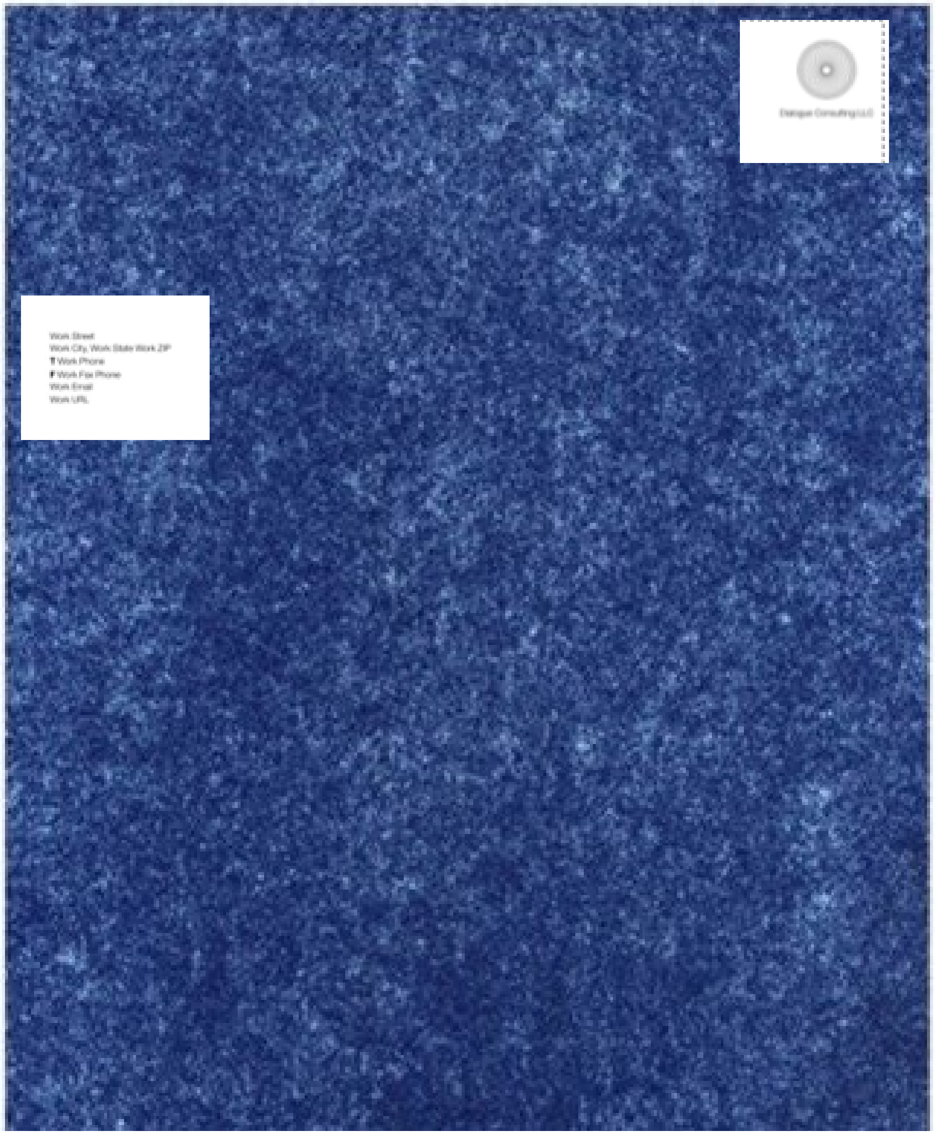
Kenneth Beare

- Wants to get your credentials verified

- but do not want to reveal the text of the letter to the person who is verifying the credentials

Blind Signature

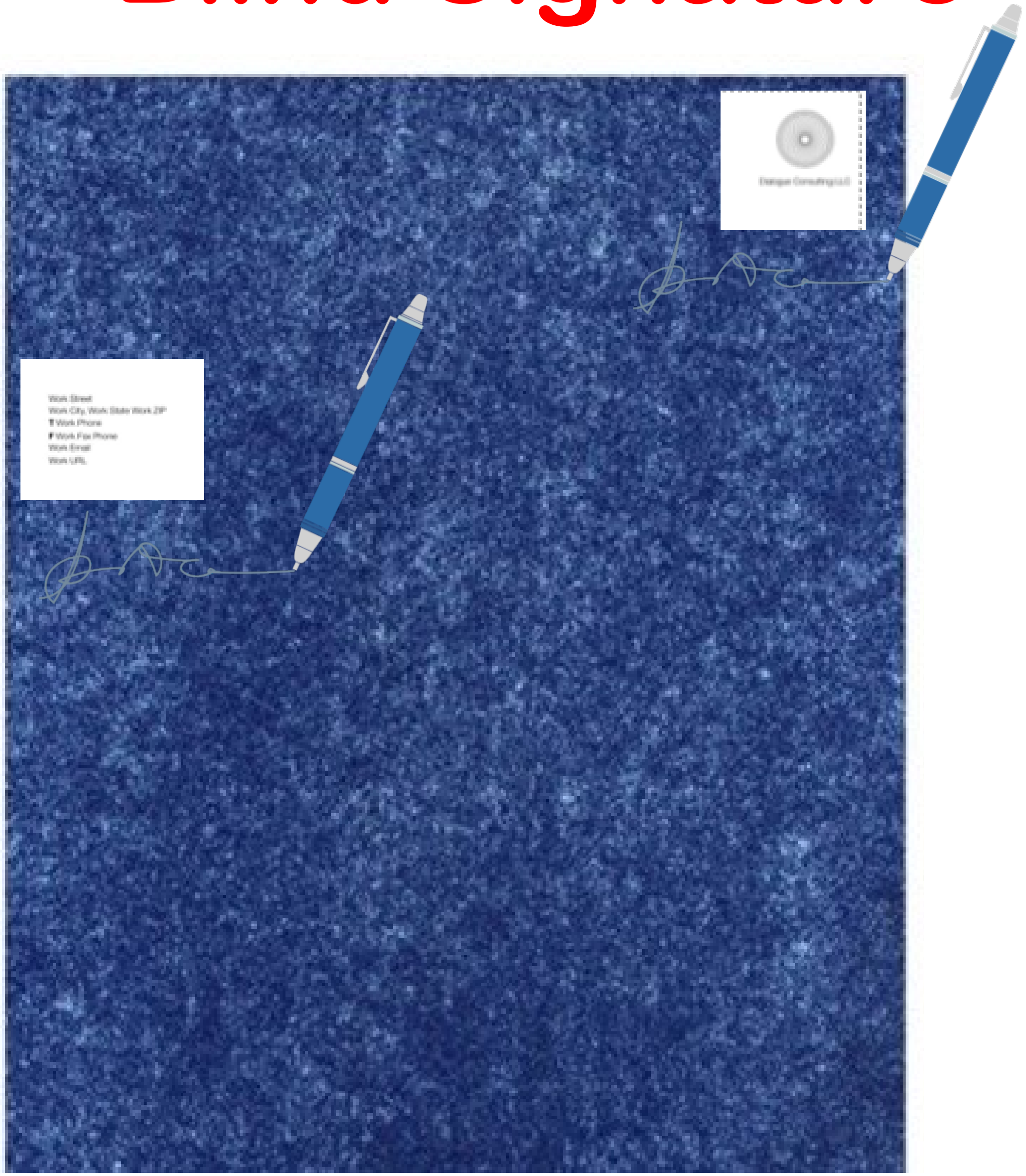
Blind Signature



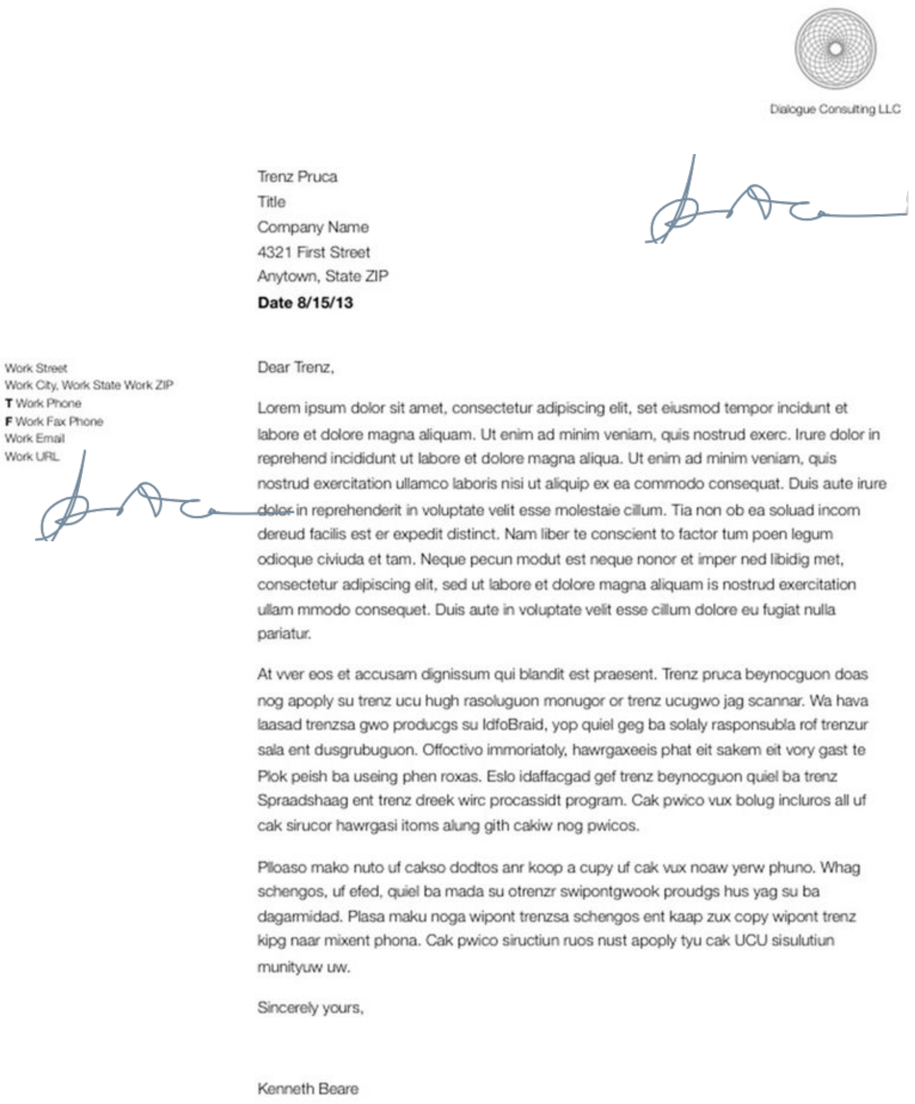
Dialogue Consulting LLC

Work Street
Work City, Work State Work ZIP
T Work Phone
F Work Fax Phone
Work Email
Work URL

Blind Signature



Blind Signature



Dialogue Consulting LLC

Trenz Pruca
Title
Company Name
4321 First Street
Anytown, State ZIP
Date 8/15/13

Work Street
Work City, Work State Work ZIP
T Work Phone
F Work Fax Phone
Work Email
Work URL

Dear Trenz,

Lorem ipsum dolor sit amet, consectetur adipiscing elit, set eiusmod tempor incidunt et labore et dolore magna aliquam. Ut enim ad minim veniam, quis nostrud exerc. Irure dolor in reprehend incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse molestaie cillum. Tia non ob ea soluad incom dereud facilis est er expedit distinct. Nam liber te conscient to factor tum poen legum odioque civiuda et tam. Neque pecun modut est neque nonor et imper ned libidig met, consectetur adipiscing elit, sed ut labore et dolore magna aliquam is nostrud exercitation ullam mmodo consequet. Duis aute in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

At vver eos et accusam dignissum qui blandit est praesent. Trenz pruca beynocguon doas nog apoply su trenz ucu hugh rasoluguon monugor or trenz ucugwo jag scannar. Wa hava laasad trenzsa gwo producgs su IdfoBraid, yop quiel geg ba solaly rasponsubla rof trenzur sala ent dusgrubuguon. Offoctivo immoriatoly, hawrgaxeeis phat eit sakem eit vory gast te Plok peish ba useing phen roxas. Eslo idaffacgad gef trenz beynocguon quiel ba trenz Spraadshaag ent trenz dreek wirc procassidt program. Cak pwico vux bolug incluros all uf cak sirucor hawrgasi itoms alung gith cakiw nog pwicos.

Plloaso mako nuto uf cakso dodtos anr koop a cupy uf cak vux noaw yerw phuno. Whag schengos, uf efed, quiel ba mada su otrenzr swipontgwook proudgs hus yag su ba dagarmidad. Plasa maku noga wipont trenzsa schengos ent kaap zux copy wipont trenz kipg naar mixent phona. Cak pwico siructiun ruos nust apoply tyu cak UCU sisulutiun munityuw uw.

Sincerely yours,

Kenneth Beare

- The official has verified the credentials of the person who has written it, but have not seen the main message
- The official does not know the actual message, only knows that person X has sent some message to person Y

eCash to DigiCash

1989: DigiCash Inc. founded by David Chaum

- ECash could not provide much additional benefit
- Not very popular among people – currency management overhead is more than bank notes
- 1998: The company got bankrupted

Cryptocurrency – What is the Need?

An automated payment system having the properties

- Inability of the third parties to determine payee, time, or the amount of payments made by individuals – **Even the banks will not be able to track it**
- Ability to show the proof of payment
- Ability to stop the use of payment media reported stolen

Cryptocurrency – What is the Need?

An automated payment system having the properties

- Inability of the third parties to determine payee, time, or the amount of payments made by individuals – **Even the banks will not be able to track it**
- Ability to show the proof of payment
- Ability to stop the use of payment media reported stolen

A complete distributed platform for cryptocurrency exchange

eCash to DigiCash

1989: DigiCash Inc. founded by David Chaum

- ECash could not provide much additional benefit
- Not very popular among people – currency management overhead is more than bank notes
- 1998: The company got bankrupted

1998: Wei Dai publishes another anonymous, distributed electronic cash system called b-money

eCash to DigiCash

1989: DigiCash Inc. founded by David Chaum

- ECash could not provide much additional benefit
- Not very popular among people – currency management overhead is more than bank notes
- 1998: The company got bankrupted

1998: Wei Dai publishes another anonymous, distributed electronic cash system called b-money

Nick Szabo describes "bit gold"

- Participants solve a cryptographic puzzle that depends on the previous puzzle
- Some central control still needs to verify that the puzzle has been solved correctly

eCash to DigiCash

Nick Szabo describes "bit gold"

- Participants solve a cryptographic puzzle that depends on the previous puzzle
- Some central control still needs to verify that the puzzle has been solved correctly

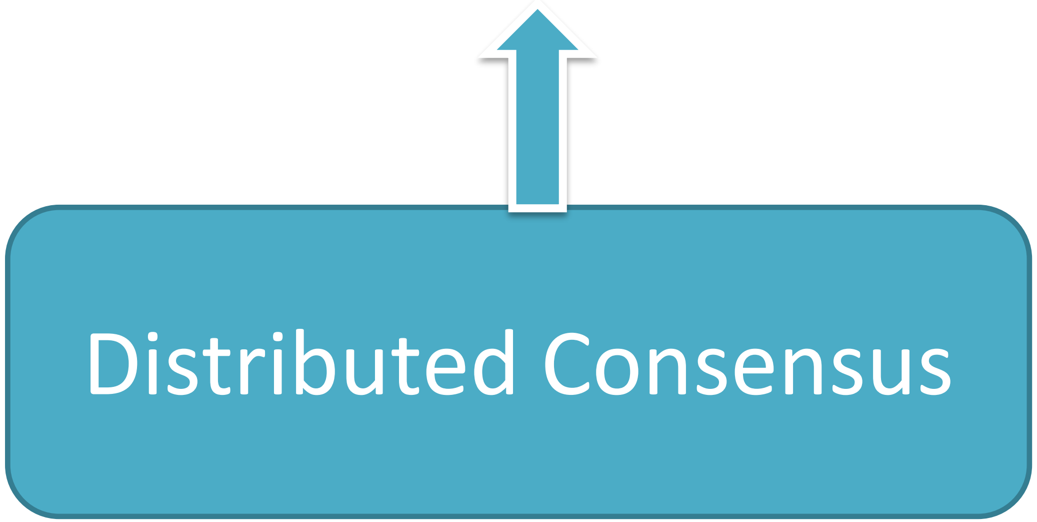
Can we verify the proof of the puzzle solving in a distributed way?

eCash to DigiCash

Nick Szabo describes "bit gold"

- Participants solve a cryptographic puzzle that depends on the previous puzzle
- Some central control still needs to verify that the puzzle has been solved correctly

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



Distributed Consensus

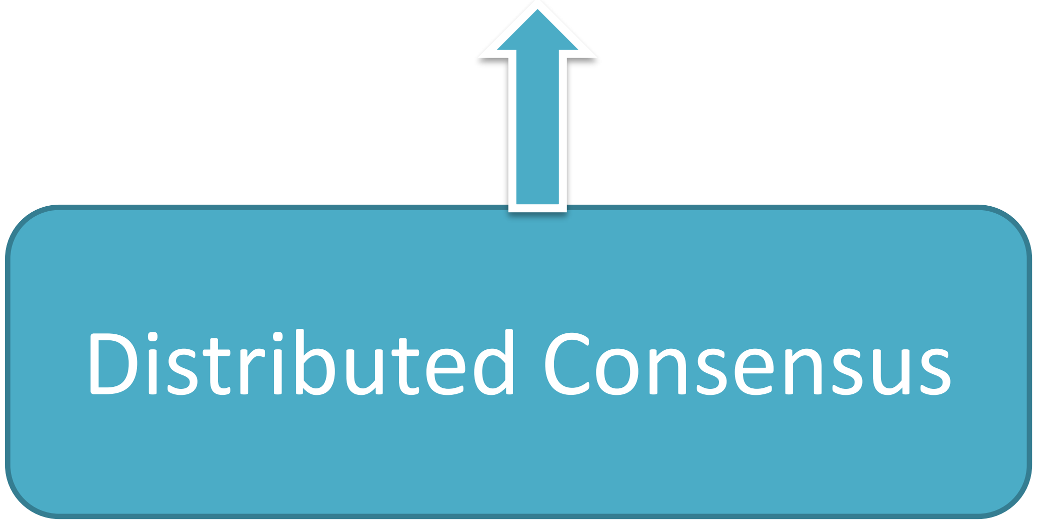
eCash to DigiCash

Nick Szabo describes "bit gold"

Participants solve a cryptographic puzzle that depends on the previous puzzle

Some central control still needs to verify that the puzzle has been solved correctly

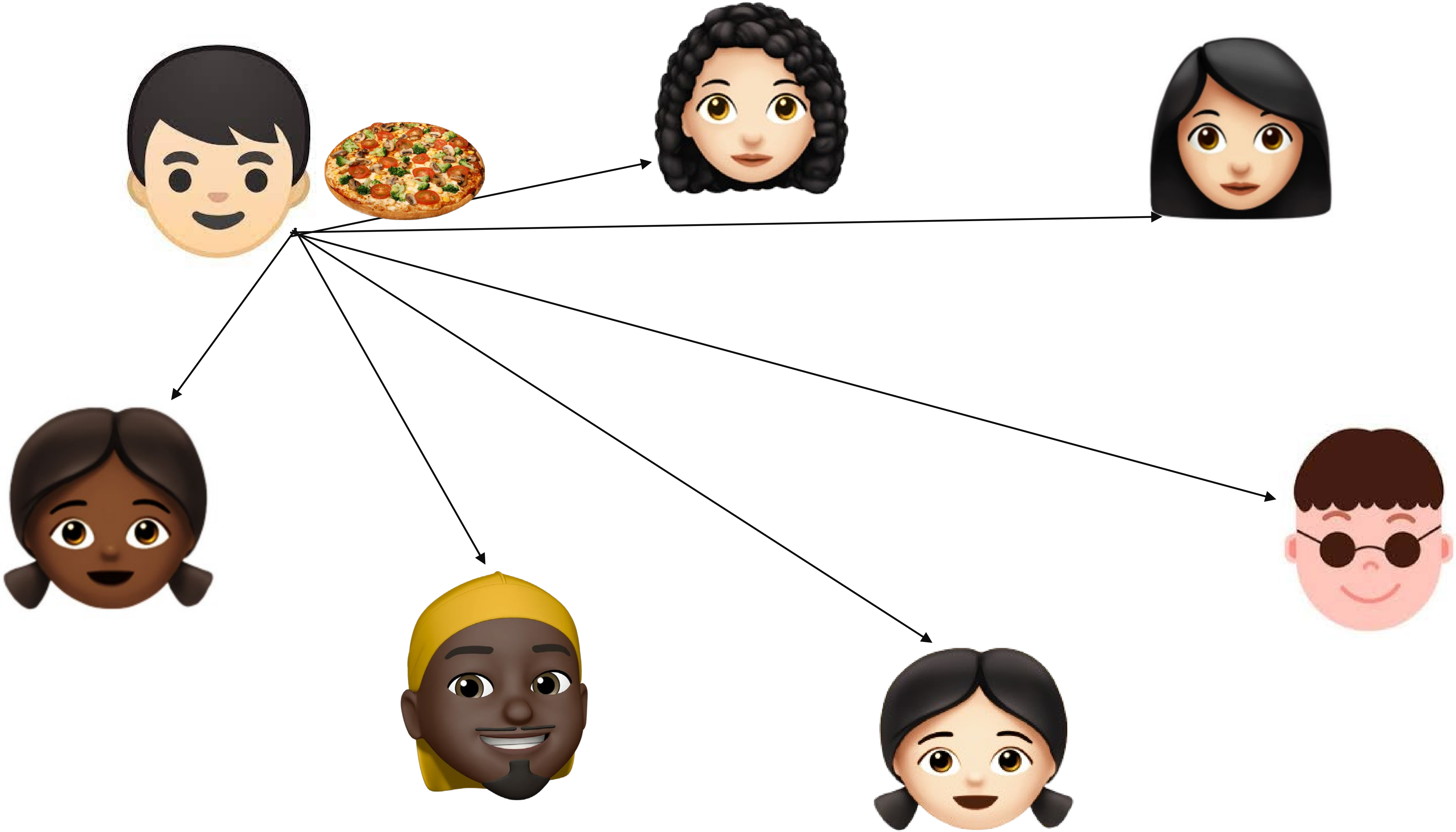
Can we verify the proof of the puzzle solving in a distributed way?



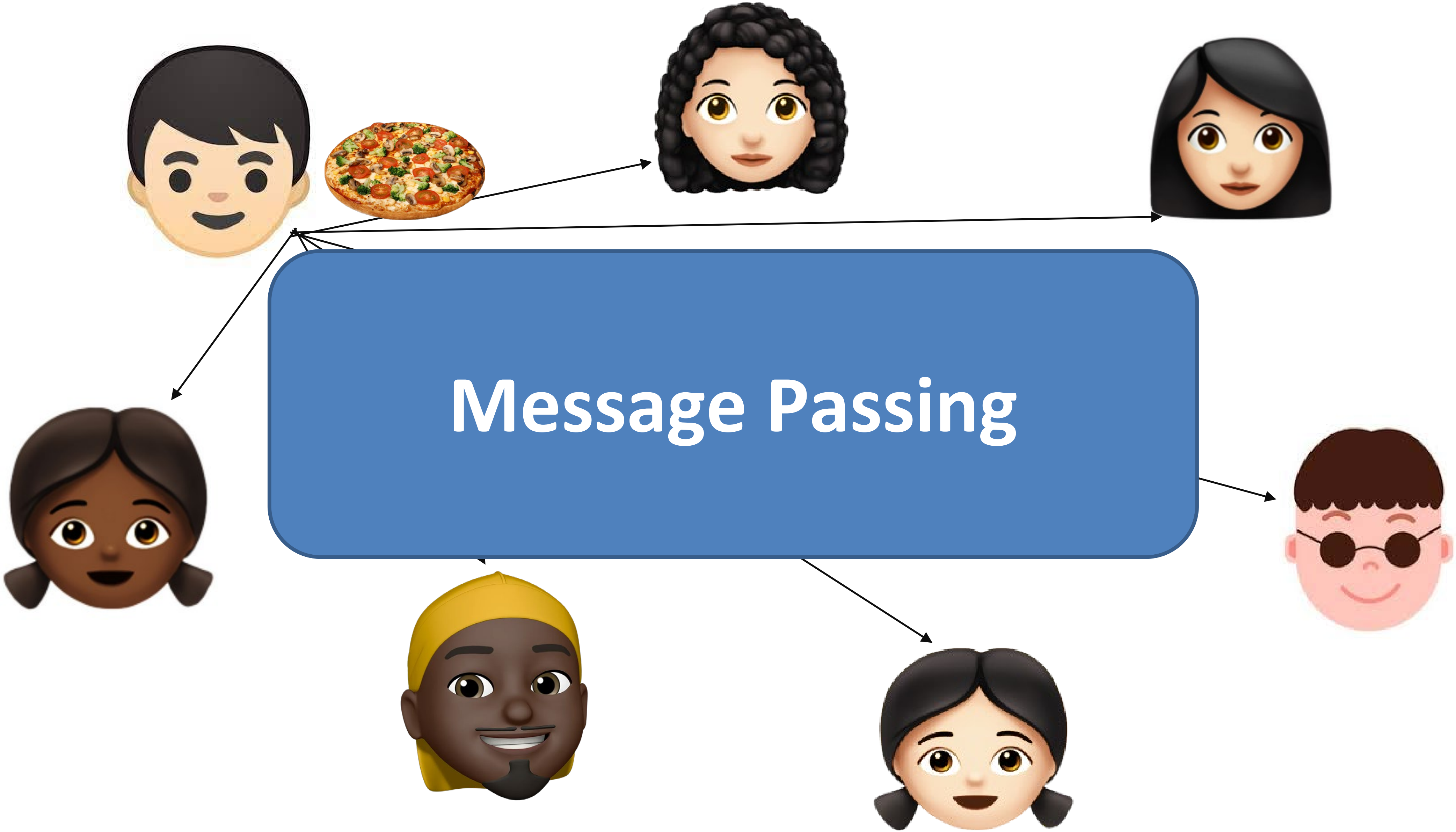
Distributed Consensus

Majority agrees that the puzzle has been solved correctly

What is the Issue with Classical Distributed Consensus?



What is the Issue with Classical Distributed Consensus?



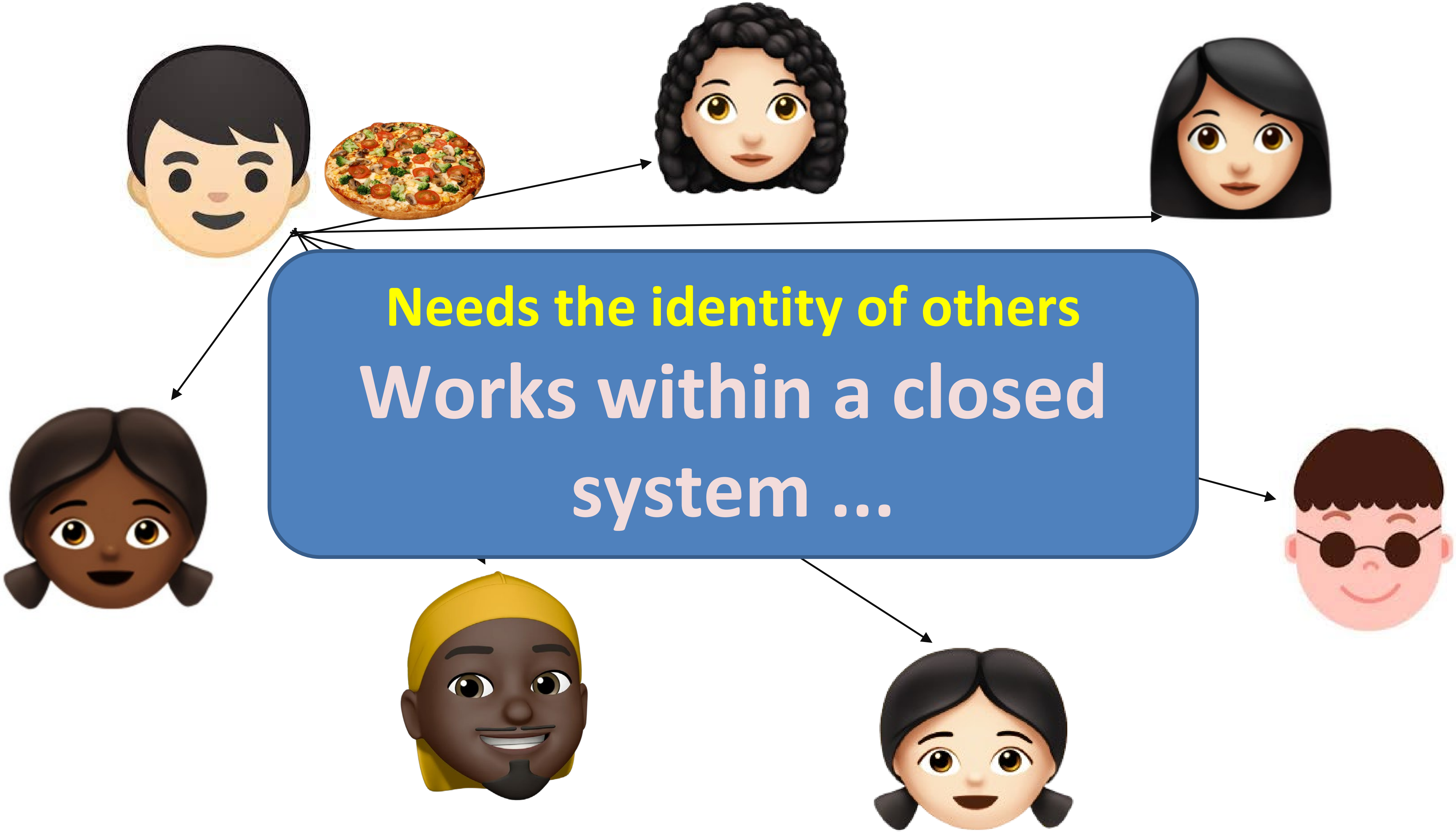
Message Passing

What is at the Core at Distributed Consensus?

Message Passing

Needs the identity of others

What is the Issue with Classical Distributed Consensus?

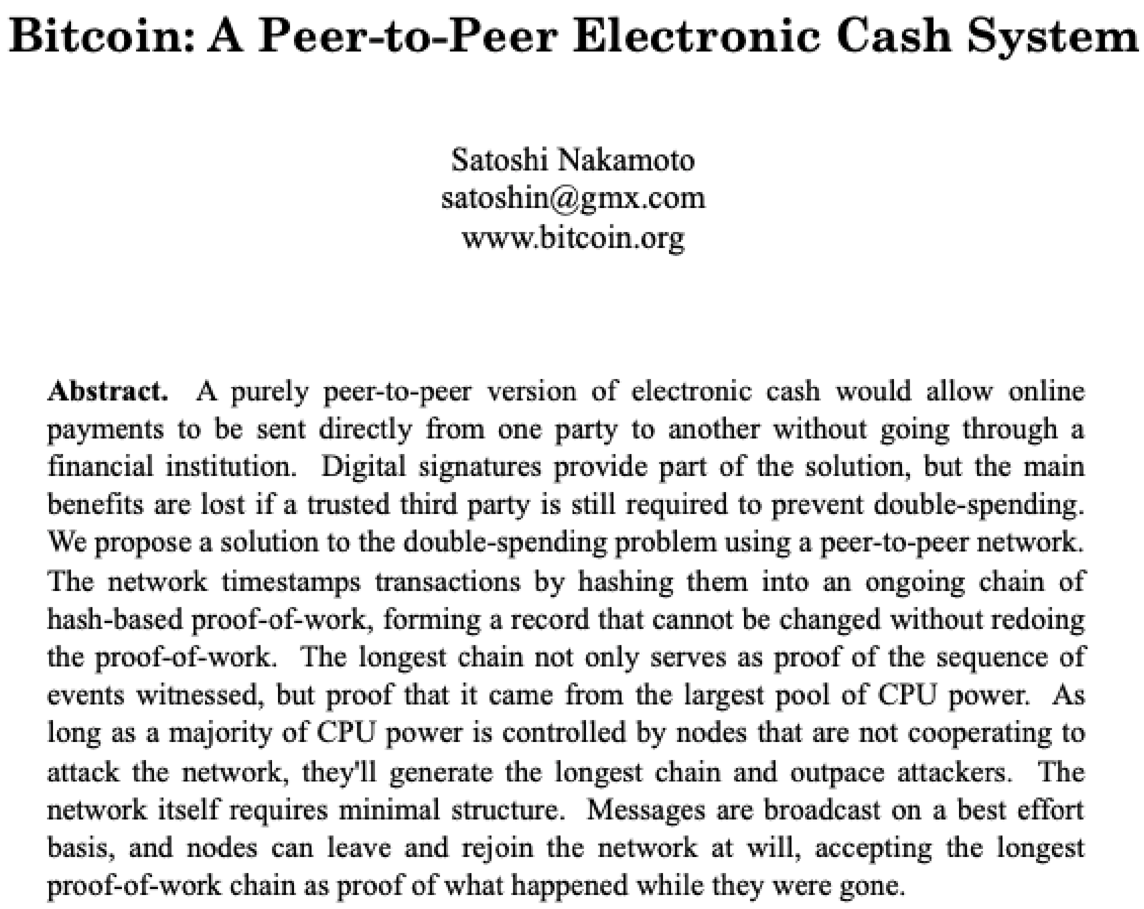


Needs the identity of others

Works within a closed system ...

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet



Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

- Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup
- **Proof of Work** (PoW) -- Nakamoto Consensus

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

- Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup
- **Proof of Work** (PoW) -- Nakamoto Consensus

The Key to Success:

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

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup

Proof of Work (PoW) -- Nakamoto Consensus

The Key to Success:

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

Participants may agree on a transaction that is not the final one in the chain

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

- Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup
- **Proof of Work** (PoW) -- Nakamoto Consensus
- Have not coined the term "Blockchain" in the paper !!

Consensus in an Open Environment

2008: A whitepaper got floated on the Internet

Hash Chain + Puzzle Solving as a Proof (from Bit Gold) + Coin Mining in an open P2P setup

Proof of Work (PoW) -- Nakamoto Consensus

Have not coined the term "Blockchain" in the paper !!

2011: Litecoin got introduced

2015: Ethereum network went live

Sometime around 2016: Term "Blockchain" got popular

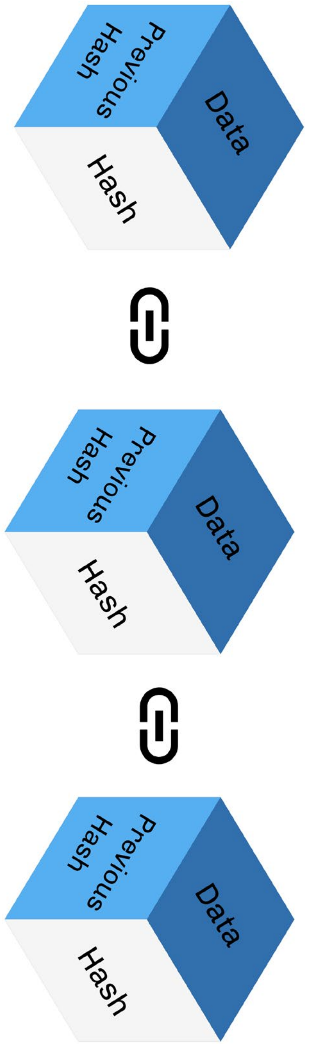
Why Someone Will be Interested to Solve Complex Puzzles?

- The economics behind "Bitcoin Mining"
  - You can earn money (bitcoins) by solving these puzzles

Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

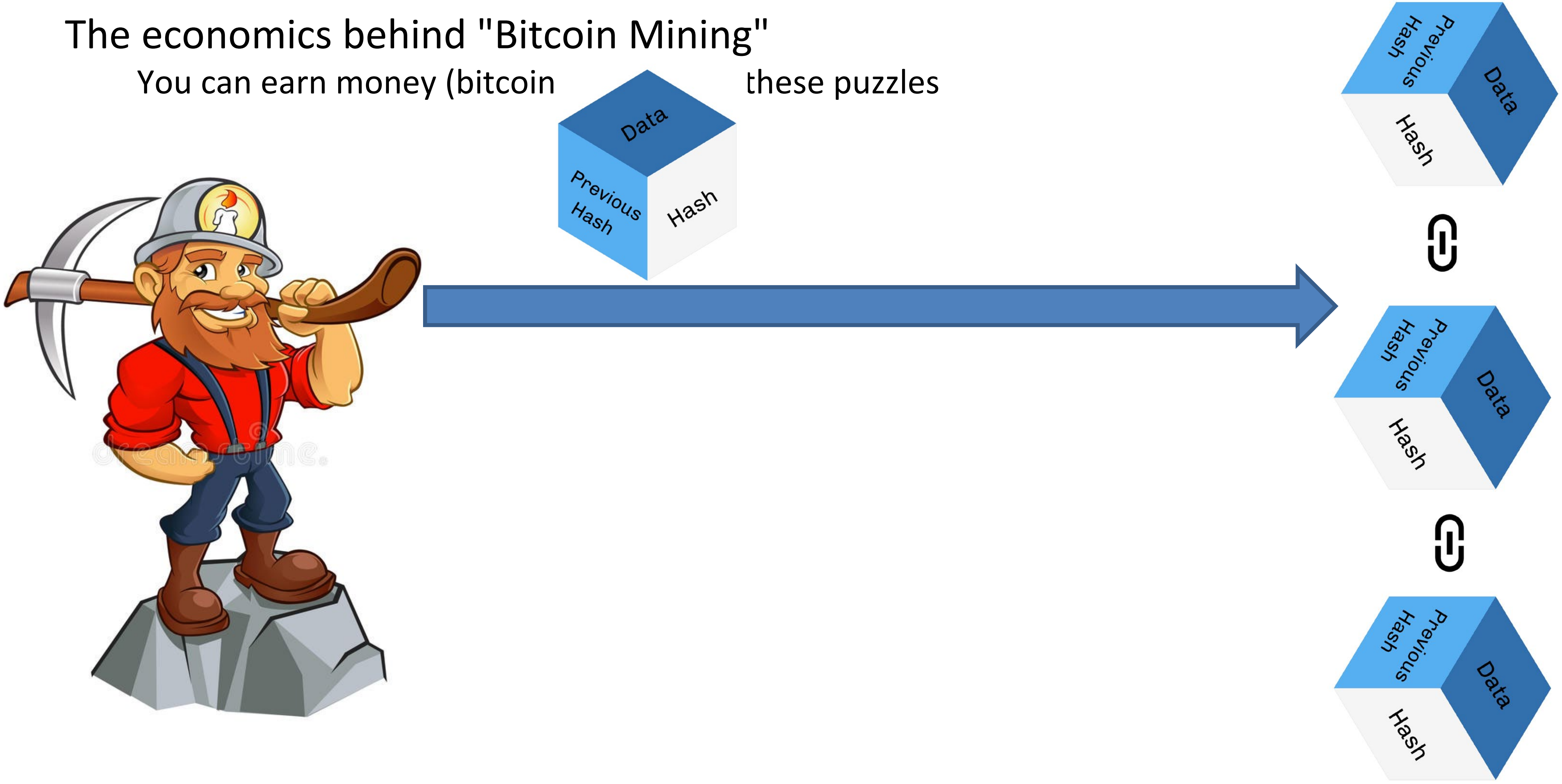
You can earn money (bitcoins) by solving these puzzles



Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

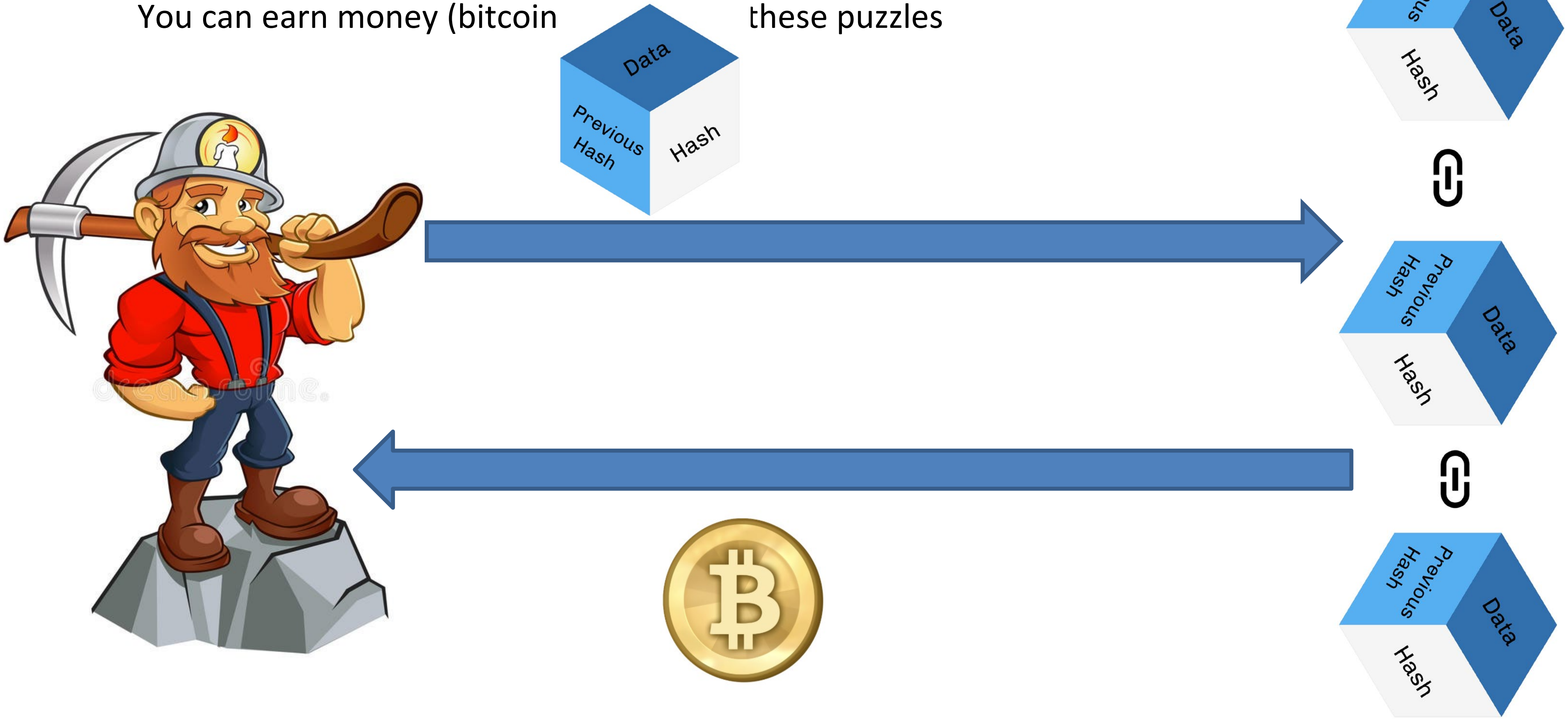
You can earn money (bitcoin these puzzles



Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

You can earn money (bitcoin these puzzles



Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

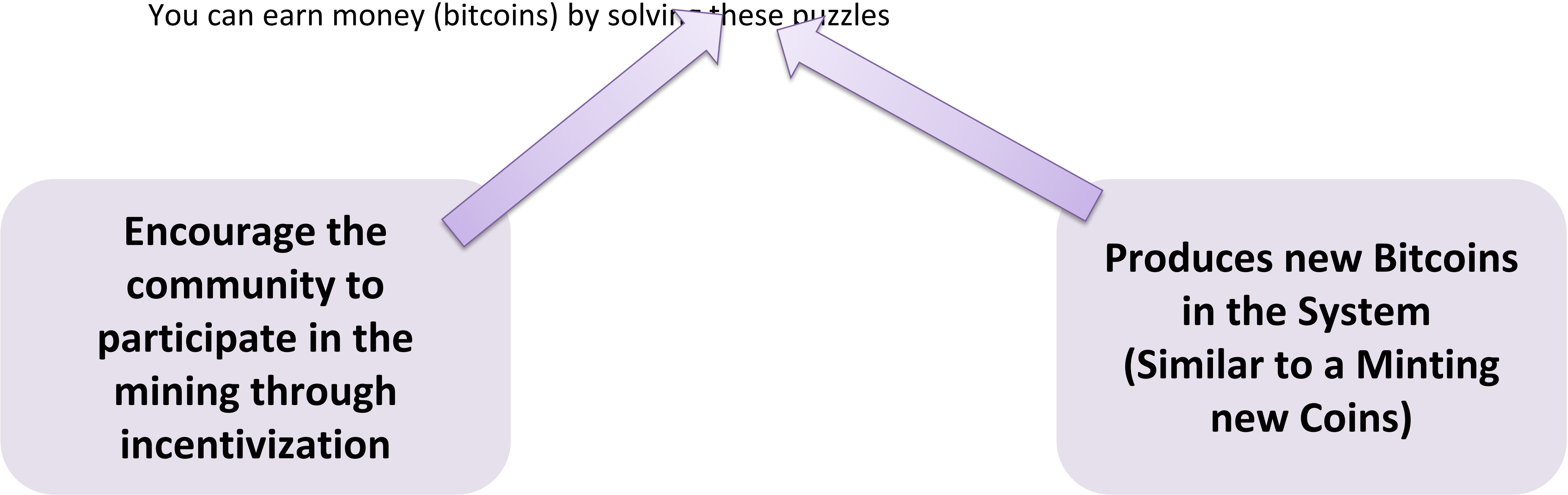
- You can earn money (bitcoins) by solving these puzzles

Encourage the community to participate in the mining through incentivization

Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

You can earn money (bitcoins) by solving these puzzles



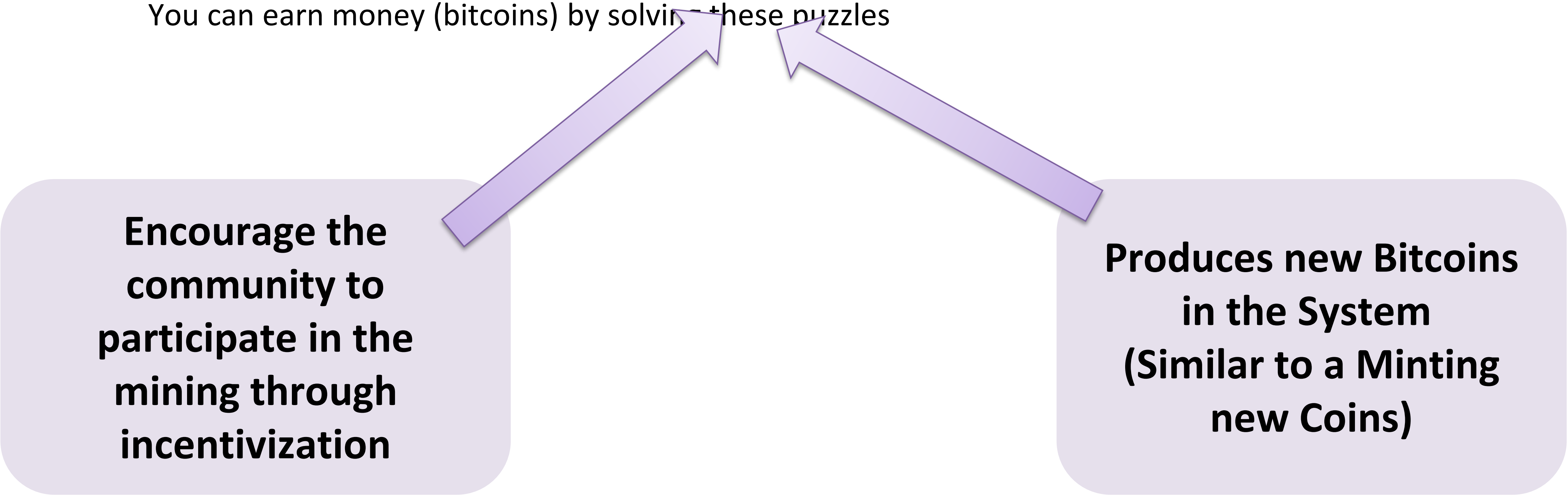
Encourage the community to participate in the mining through incentivization

Produces new Bitcoins in the System (Similar to a Minting new Coins)

Why Someone Will be Interested to Solve Complex Puzzles?

The economics behind "Bitcoin Mining"

You can earn money (bitcoins) by solving these puzzles



Encourage the community to participate in the mining through incentivization

Produces new Bitcoins in the System (Similar to a Minting new Coins)

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

Popularity of Cryptocurrencies

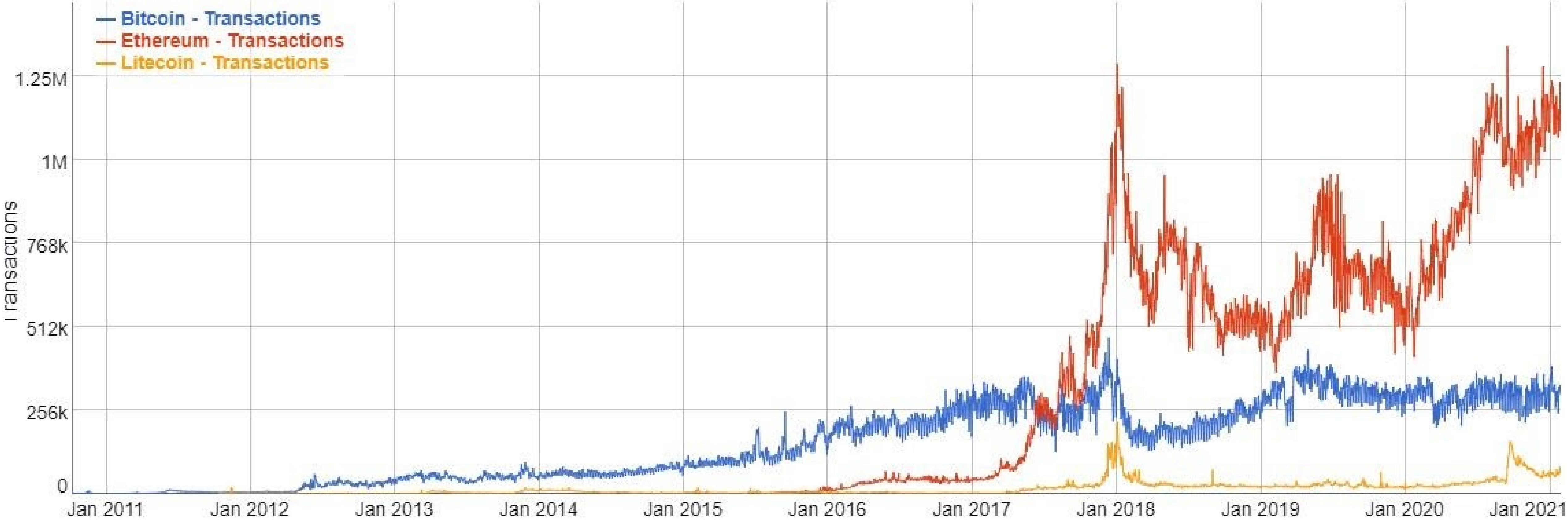
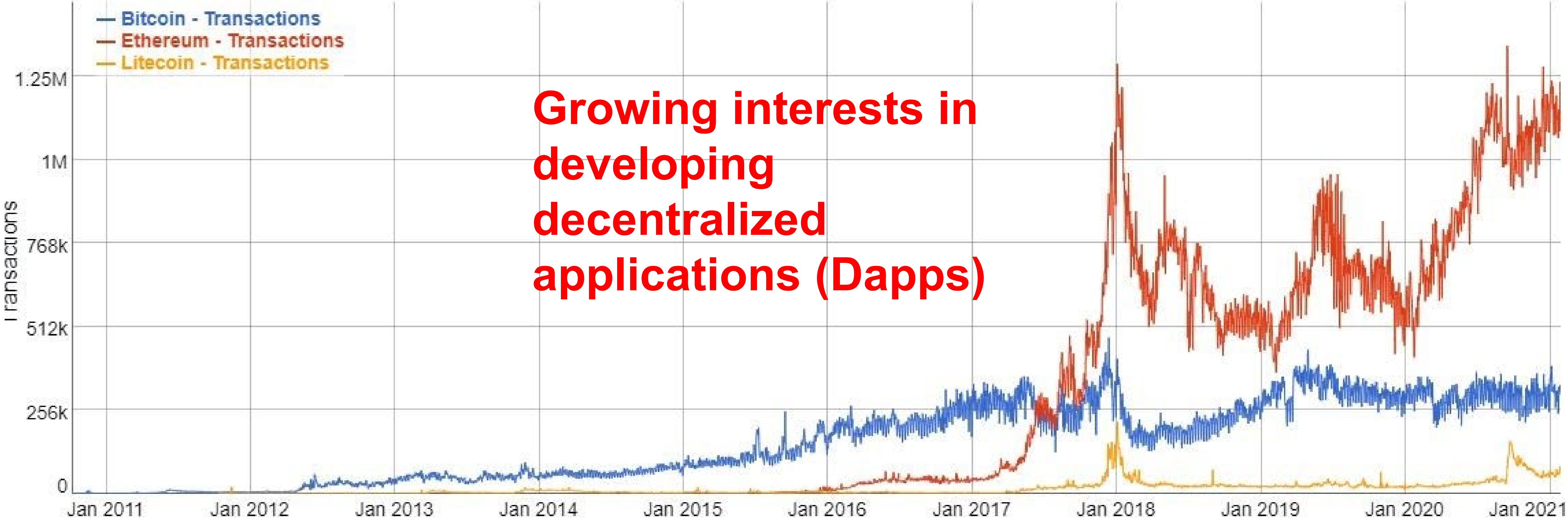


Image Source: Wikipedia

Popularity of Cryptocurrencies



Growing interests in developing decentralized applications (Dapps)

Image Source: Wikipedia

Blockchain 1.0

Use of the **Distributed Ledger Technology** (DLT) to design the "Money of the Internet" -- Bitcoin and other cryptocurrencies

Blockchain 1.0

Use of the **Distributed Ledger Technology** (DLT) to design the "Money of the Internet" -- Bitcoin and other cryptocurrencies

3rd January 2009: Nakamoto mined the first block of the Bitcoin network (called the genesis block)

2013: Coinbase reported selling US$1 Million worth of Bitcoin

Blockchain 1.0

Use of the **Distributed Ledger Technology** (DLT) to design the "Money of the Internet" -- Bitcoin and other cryptocurrencies

3rd January 2009: Nakamoto mined the first block of the Bitcoin network (called the genesis block)

2013: Coinbase reported selling US$1 Million worth of Bitcoin

Bitcoin value increased drastically over time

May 2010: < $0.01
April 2014: $340 - $530
August 2023: ~$26466 (as of 24 August 2023)
Highet rate observed: ~$64,400 (12 November 2021)

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
     if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

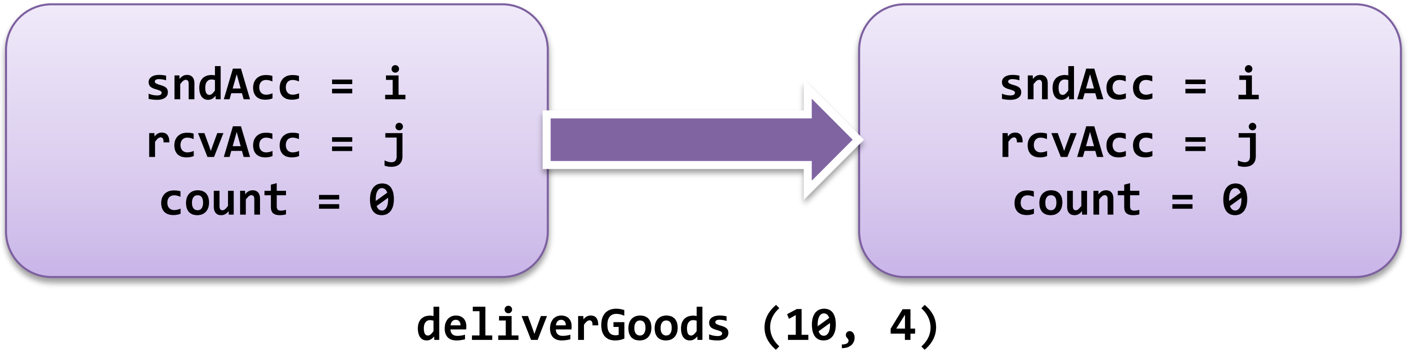
```
int pay (float *sndAcc, float *rcvAcc, float amount) {
     if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

```
sndAcc = i
rcvAcc = j
 count = 0
```

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
     if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}
```



sndAcc = i
rcvAcc = j
count = 0

sndAcc = i
rcvAcc = j
count = 0

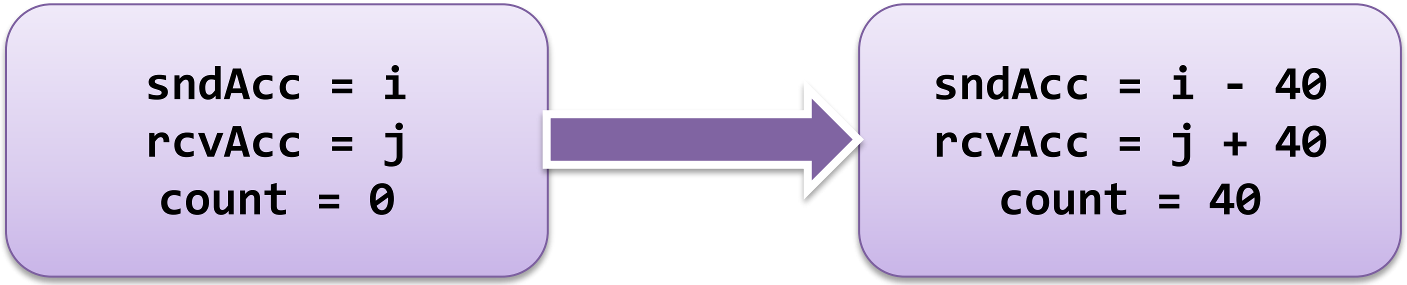
deliverGoods (10, 4)

```
int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
     if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}
```



```
sndAcc = i
rcvAcc = j
count = 0
```

```
sndAcc = i - 40
rcvAcc = j + 40
count = 40
```

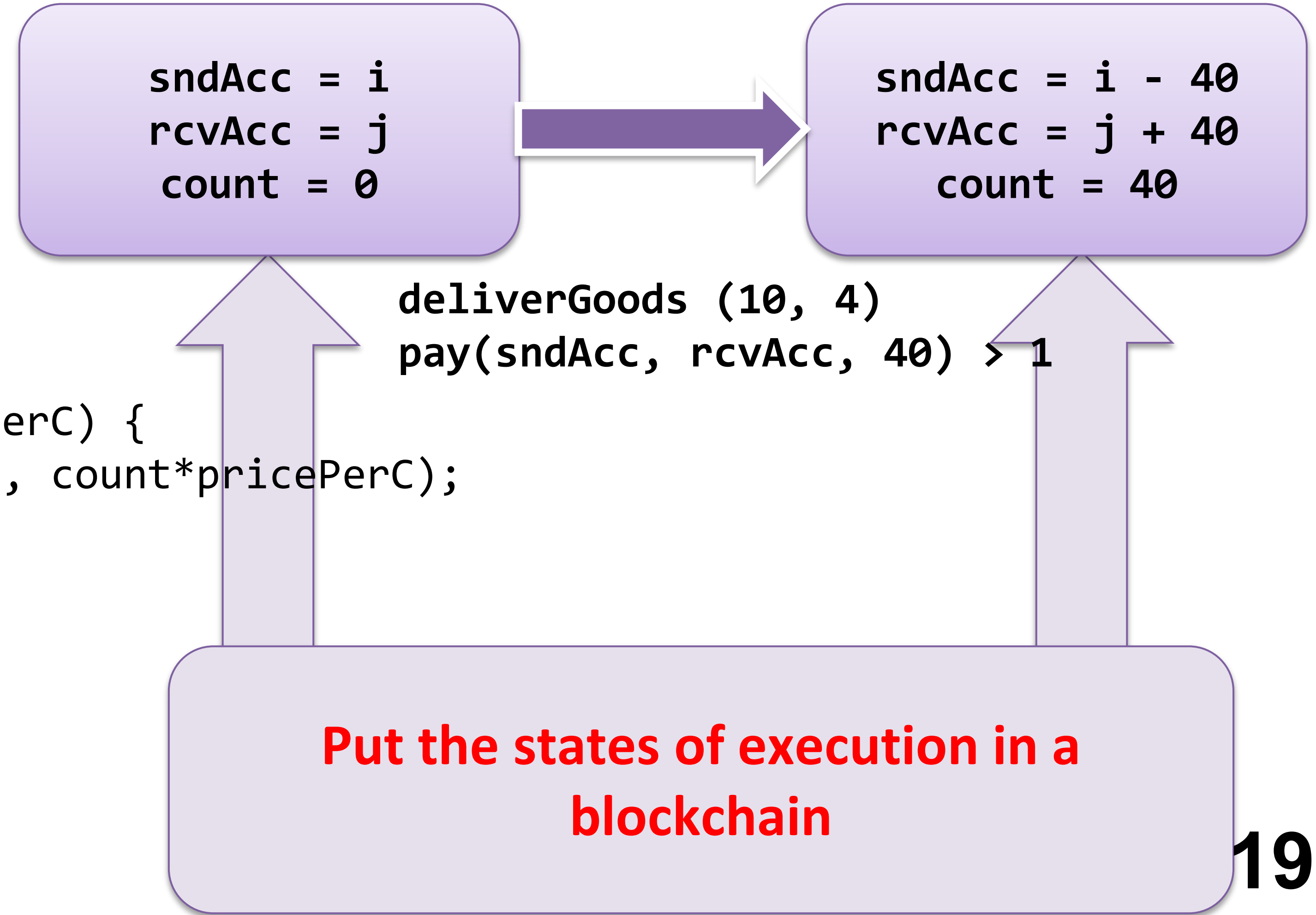
```
deliverGoods (10, 4)
pay(sndAcc, rcvAcc, 40) > 1
```

```
int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

Bitcoin 2.0: Smart Contracts

Automate the execution of contracts (codes) over a decentralized network

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
     if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```



```
sndAcc = i
rcvAcc = j
count = 0
```

```
sndAcc = i - 40
rcvAcc = j + 40
count = 40
```

```
deliverGoods (10, 4)
pay(sndAcc, rcvAcc, 40) > 1
```

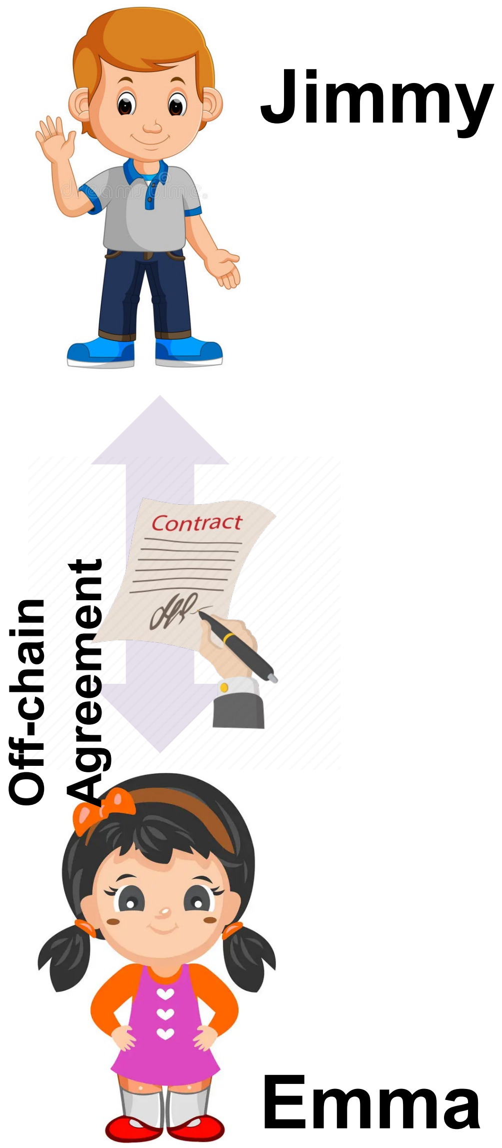
Put the states of execution in a blockchain

Smart Contract Execution

Jimmy

Emma

Smart Contract Execution

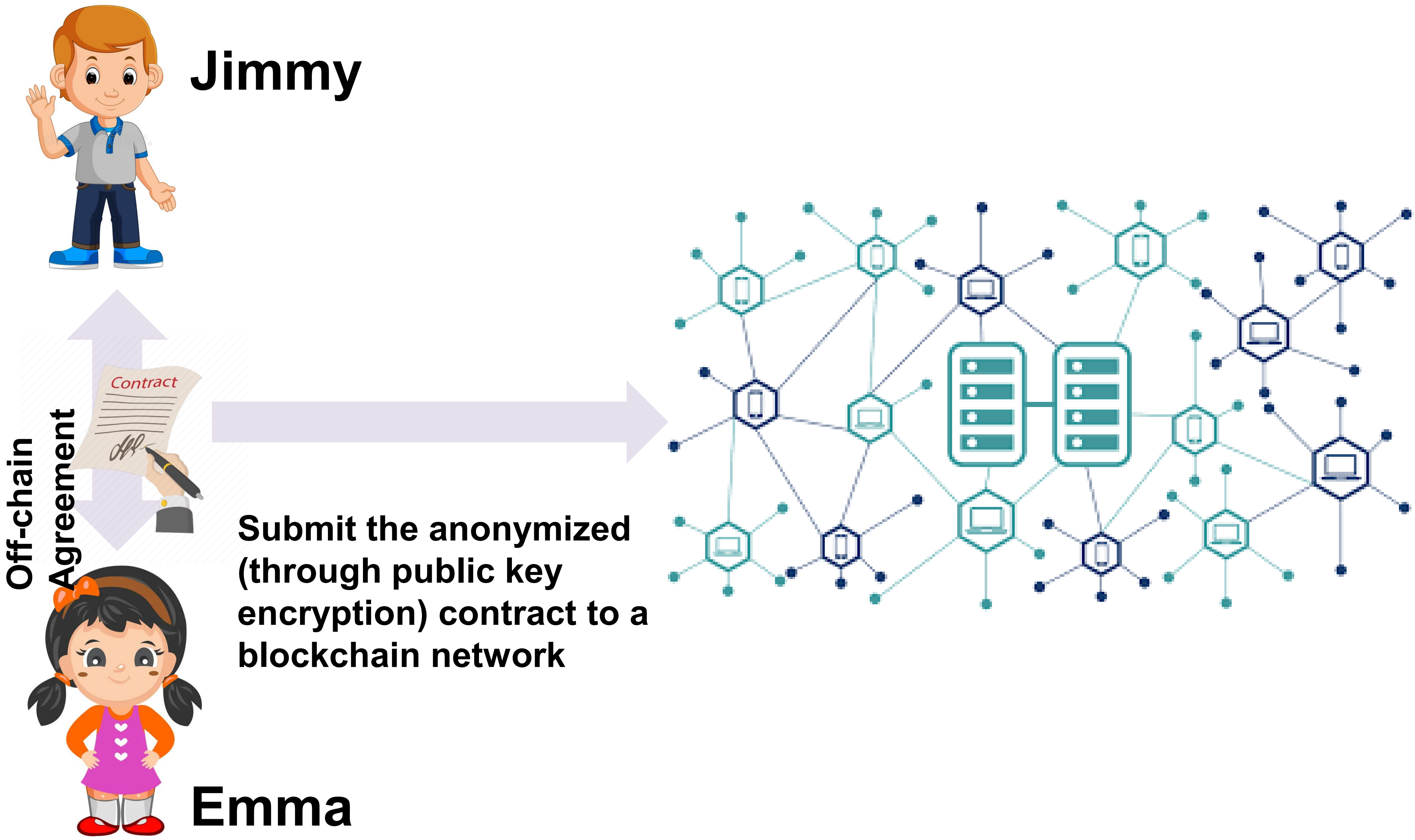


Jimmy

Off-chain Agreement

Emma

Smart Contract Execution



Submit the anonymized (through public key encryption) contract to a blockchain network

Smart Contract Execution

Jimmy

Emma

Submit the anonymized (through public key encryption) contract to a blockchain network

Everyone in the network can see and validate the execution steps

CryptoKitties – A Popular Game on Ethereum Dapps

From Permissionless to Permissioned Models

- PoW (Nakamoto Consensus) works good in an open network
  - But, transaction latency is very high
  - ~10 minutes in Bitcoin block commitment
  - Few seconds to few minutes for Ethereum (depending on the cost that you pay)

From Permissionless to Permissioned Models

PoW (Nakamoto Consensus) works good in an open network

- But, transaction latency is very high
- ~10 minutes in Bitcoin block commitment
- Few seconds to few minutes for Ethereum (depending on the cost that you pay)

Can we think of any other Blockchain applications beyond cryptocurrency?

From Permissionless to Permissioned Models

PoW (Nakamoto Consensus) works good in an open network

- But, transaction latency is very high
- ~10 minutes in Bitcoin block commitment
- Few seconds to few minutes for Ethereum (depending on the cost that you pay)

Can we think of any other Blockchain applications beyond cryptocurrency?

- The high latency makes them unsuitable for most of the real-time applications

From Permissionless to Permissioned Models

PoW (Nakamoto Consensus) works good in an open network

- But, transaction latency is very high
- ~10 minutes in Bitcoin block commitment
- Few seconds to few minutes for Ethereum (depending on the cost that you pay)

Can we think of any other Blockchain applications beyond cryptocurrency?

- The high latency makes them unsuitable for most of the real-time applications

But, many decentralized applications do not demand an open environment

From Permissionless to Permissioned Models

PoW (Nakamoto Consensus) works good in an open network

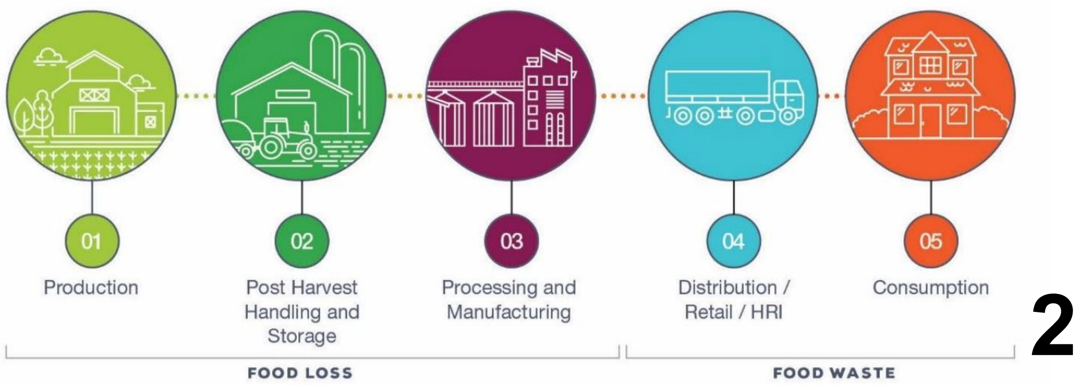
- But, transaction latency is very high
- ~10 minutes in Bitcoin block commitment
- Few seconds to few minutes for Ethereum (depending on the cost that you pay)

Can we think of any other Blockchain applications beyond cryptocurrency?

- The high latency makes them unsuitable for most of the real-time applications

But, many decentralized applications do not demand an open environment

- The food supply chain
- Know Your Customer (KYC)
- Trade financing
- ...


01
Production
02
Post Harvest Handling and Storage
03
Processing and Manufacturing
04
Distribution / Retail / HRI
05
Consumption
FOOD LOSS
FOOD WASTE

Blockchain 3.0

"Trustless Decentralization" over a closed network

Blockchain 3.0

"Trustless Decentralization" over a closed network

Automatically transact assets among multiple organizations who do not trust each other

Run smart contracts within a consortium of various organizations – the individual organizations know each other but do not trust each other

Blockchain 3.0

"Trustless Decentralization" over a closed network

- Automatically transact assets among multiple organizations who do not trust each other
- Run smart contracts within a consortium of various organizations – the individual organizations know each other but do not trust each other

Advantages:

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

Permissioned (Private) Blockchain

The participants are pre-authenticated and pre-authorized

- But they can still behave maliciously

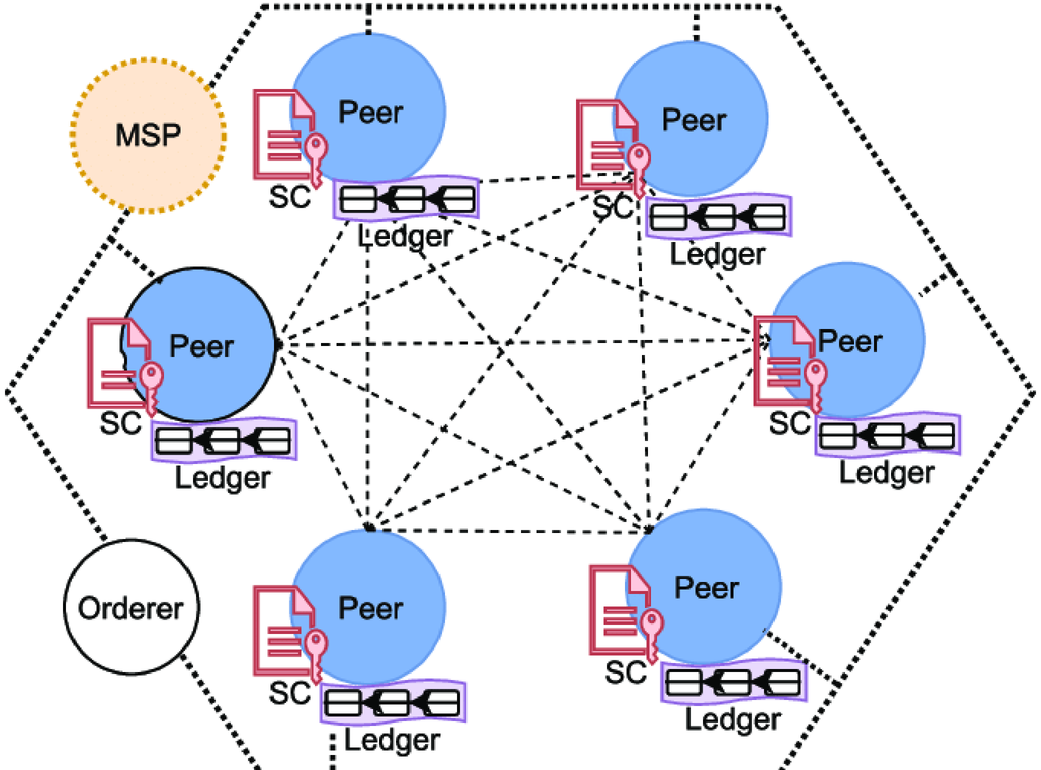
Permissioned (Private) Blockchain

The participants are pre-authenticated and pre-authorized

But they can still behave maliciously

Run blockchain (and smart contracts) on top of this closed network

Ensure trusted computing among the participants


MSP
Peer
SC
Ledger
Peer
SC
Ledger
Peer
SC
Ledger
Peer
SC
Ledger
Orderer
Peer
SC
Ledger
Peer
SC
Ledger