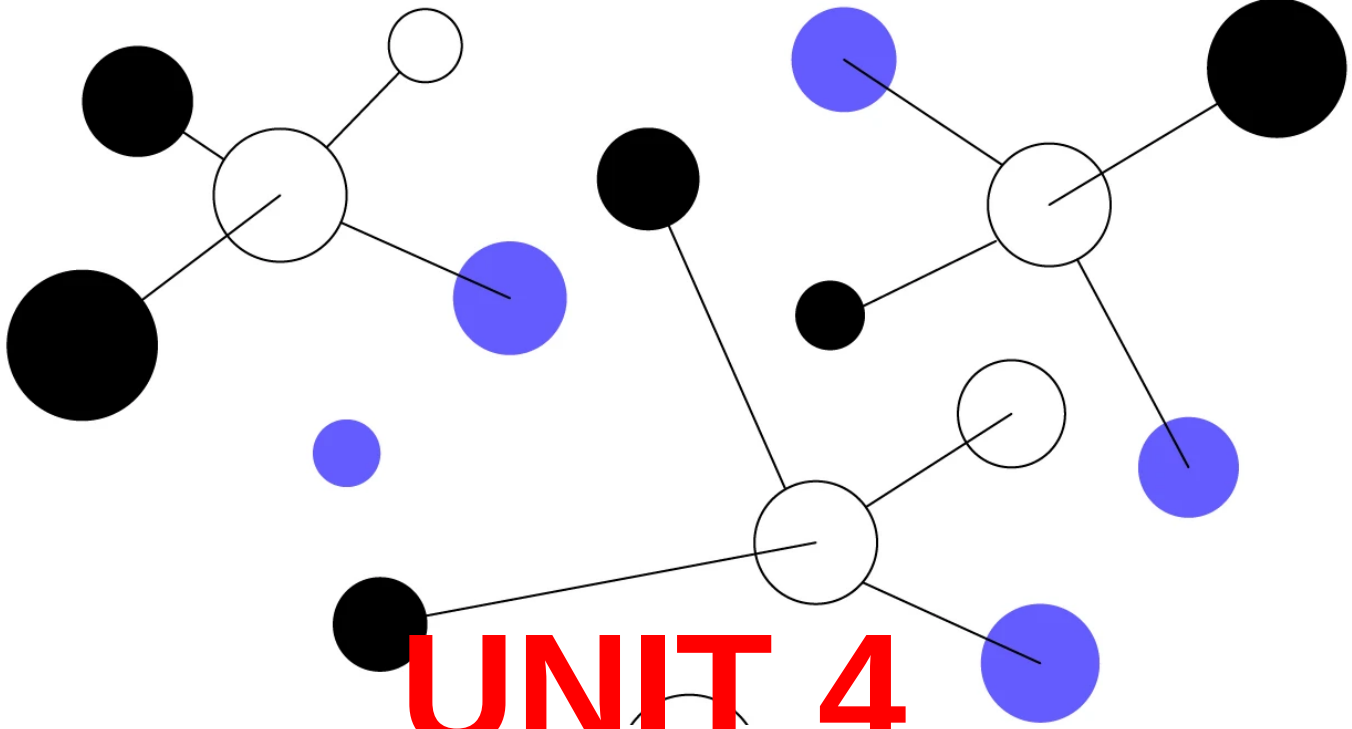


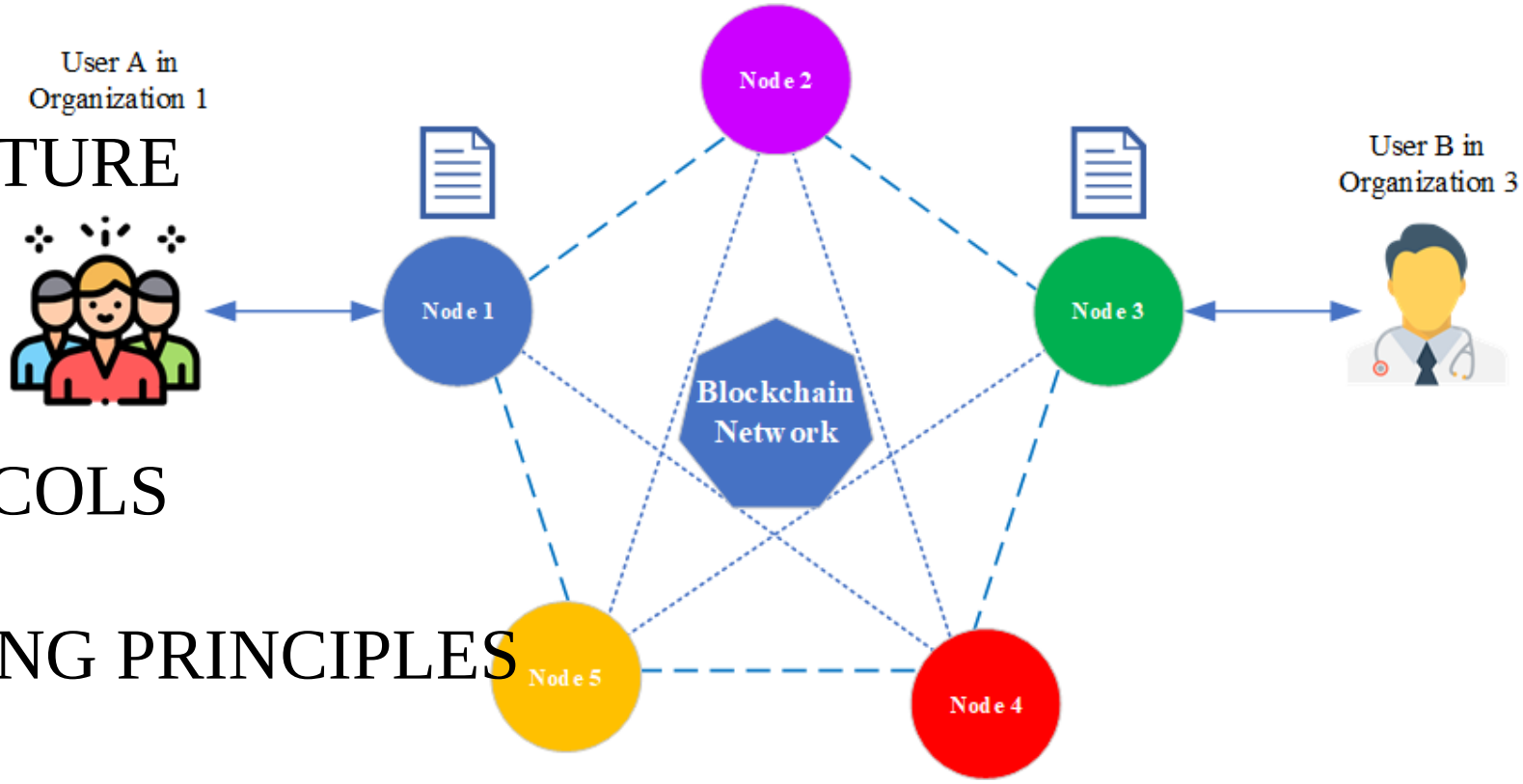
# UNIT 4

# BLOCKCHAIN NETWORK

Lecturer: Ph.D Lê Quang Huy

# CONTENTS

1. INTRODUCTION
2. NETWORK ARCHITECTURE
3. BLOCKCHAIN NODES
4. BLOCKCHAIN PROTOCOLS
5. BLOCKCHAIN WORKING PRINCIPLES
6. SUMMARY
7. DISCUSSION



# 1. INTRODUCTION

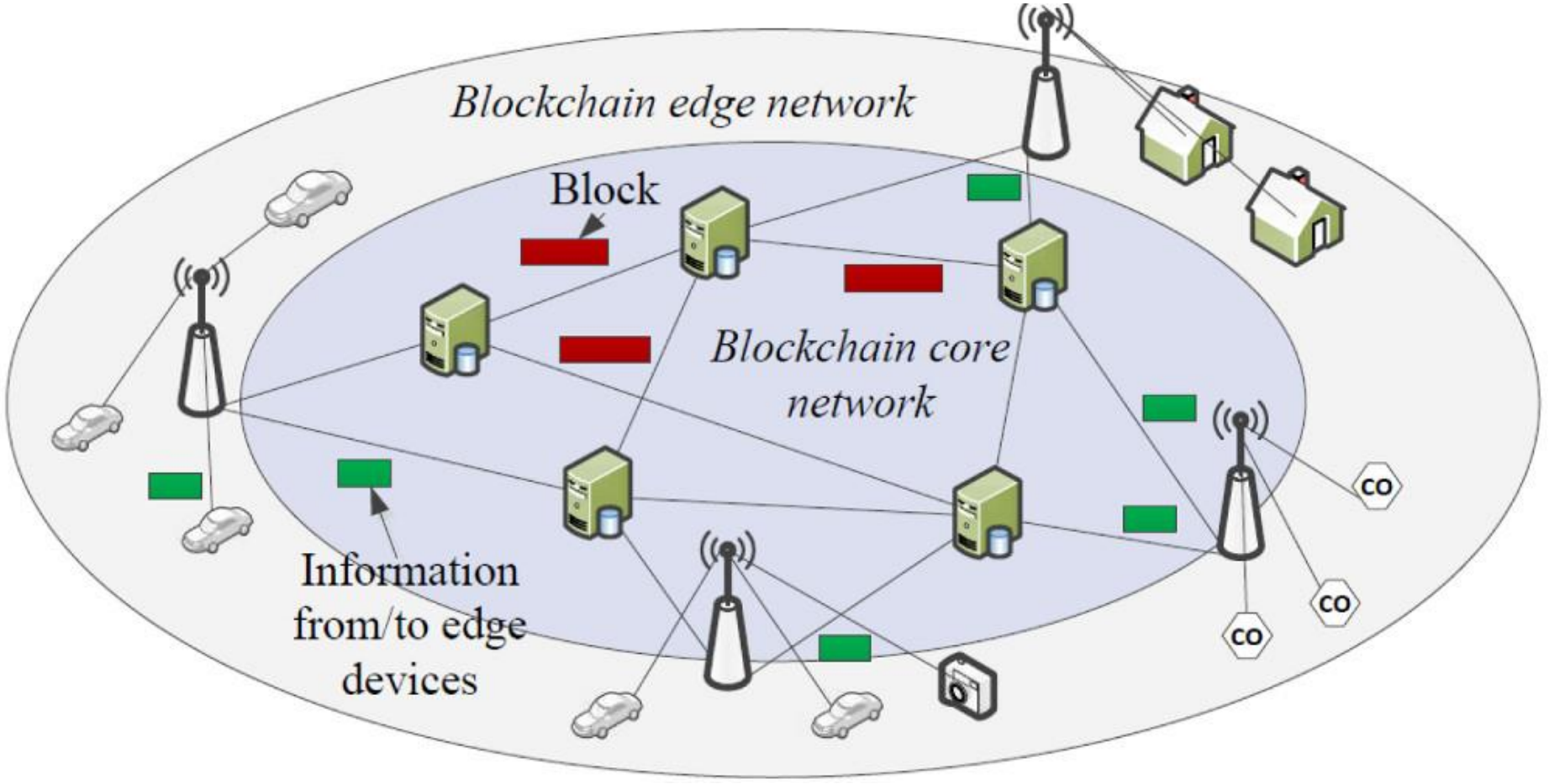
Blockchain network:

- Technical infrastructure

- Access ledger (transaction, smart contract)

Components:

- Nodes
- Architecture
- Protocol



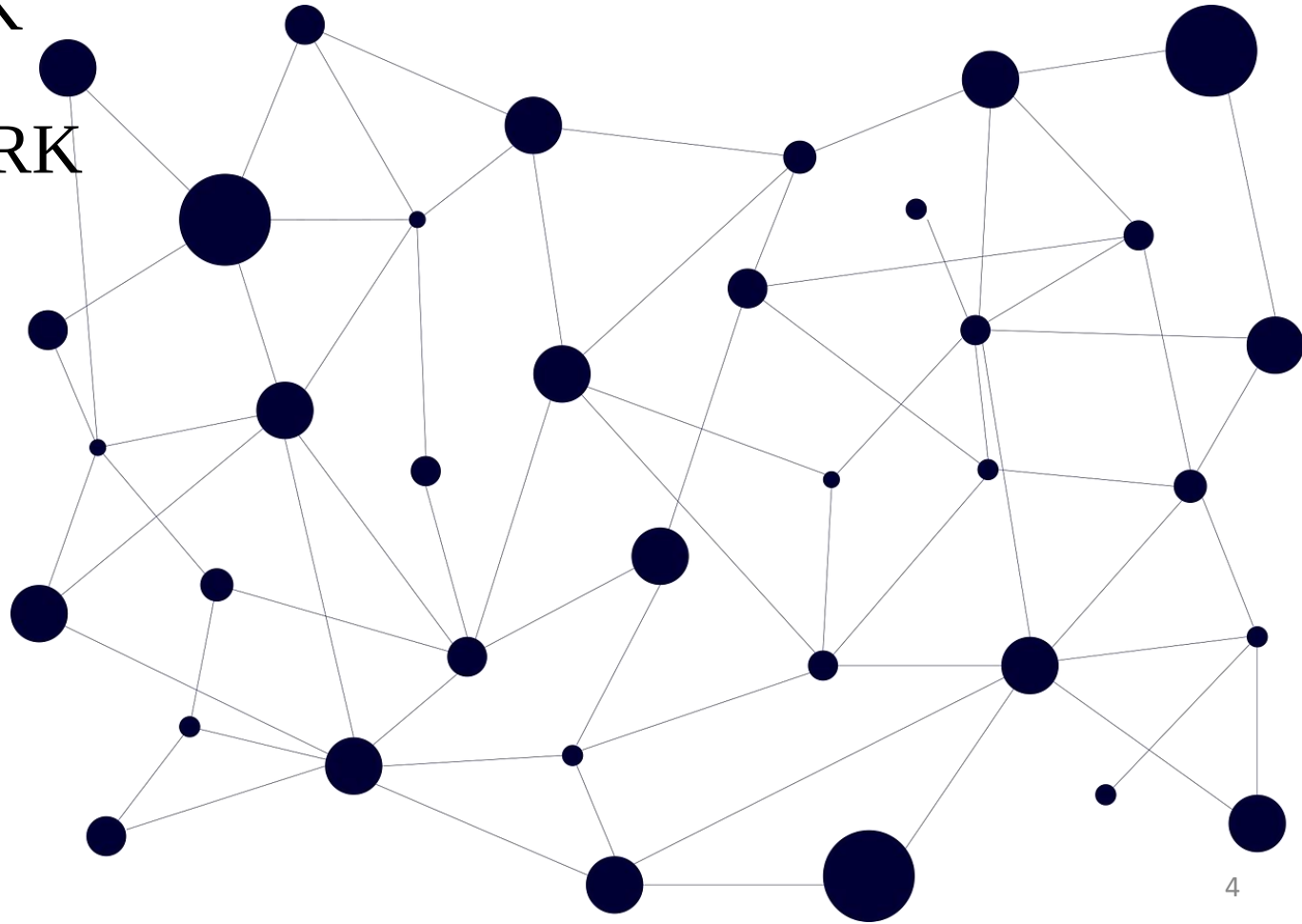
# 2. NETWORK ARCHITECTURE

2.1. NETWORK LAYOUT

2.2. PEER TO PEER NETWORK

2.3. CLIENT SERVER NETWORK

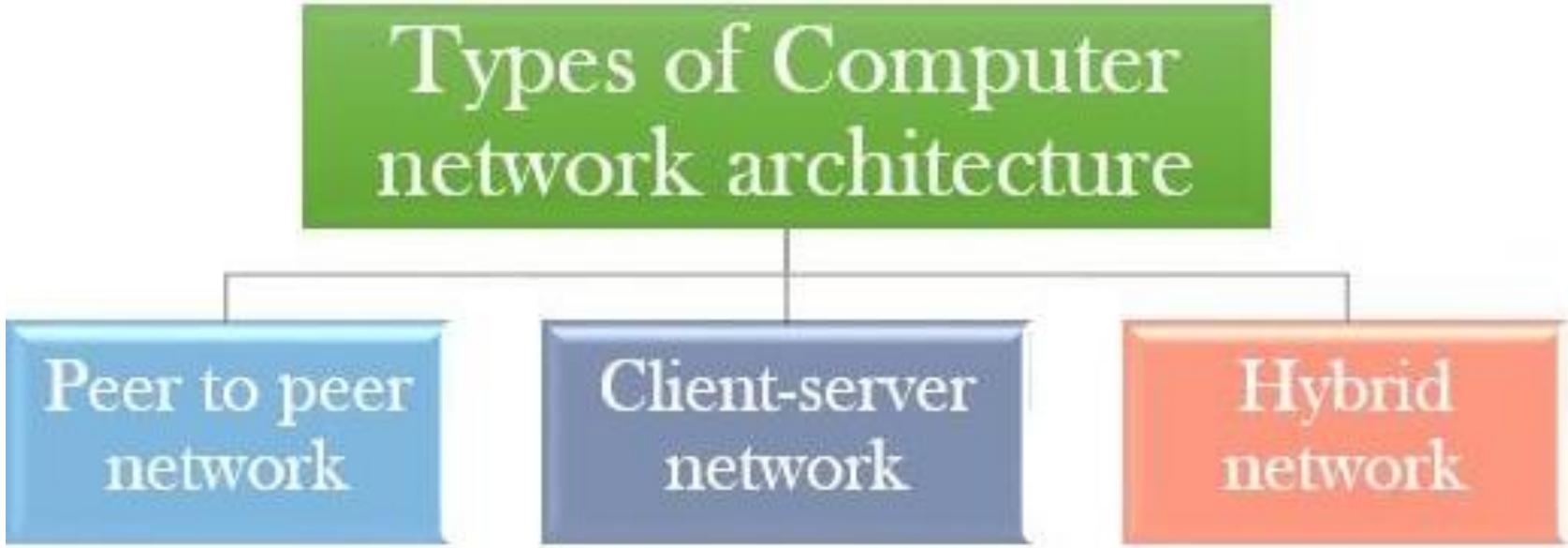
2.4. HYBRID NETWORK



# 2.1. NETWORK LAYOUT

Network architecture:

- Logical/structural layout
- equipment software, protocols, infrastructure of data and connectivity between components.




Network architecture: design of network framework of:

- physical components
- functional organization and configuration,
- operational principles and procedures
- communication protocols.

# 2.2. PEER TO PEER NETWORK

Peer to Peer (P2P) network:

- All computers are linked together, equal privilege/responsibilities for processing data.

P2P network type:

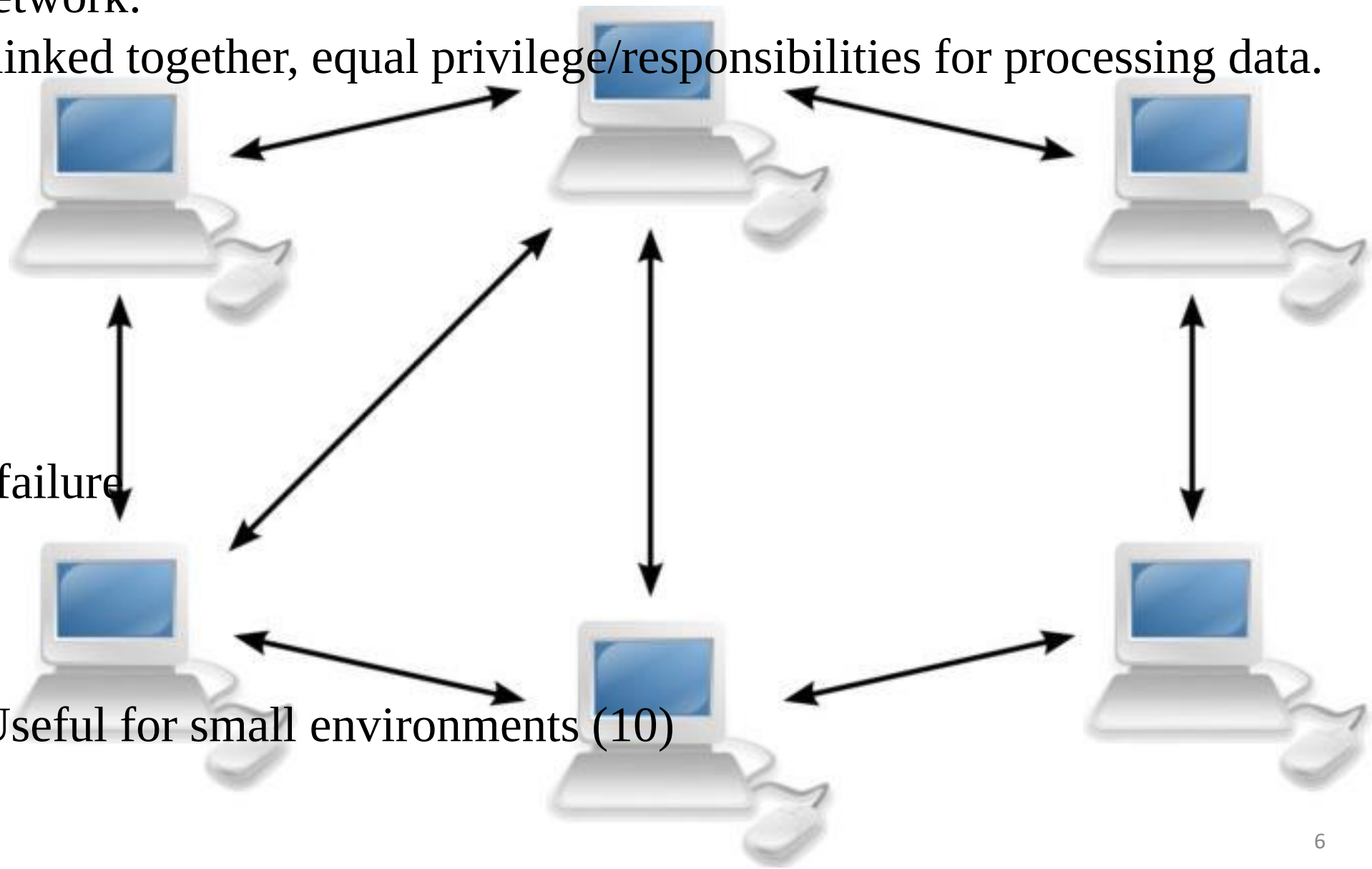
- Unstructured
- Structured
- Hybrid

Advantages:

- Low cost
- No single point of failure
- Easy installation

Disadvantages:

- Backup each node
- Scalability issue (Useful for small environments (10)



# 2.3. CLIENT SERVER NETWORK

Client-server network:

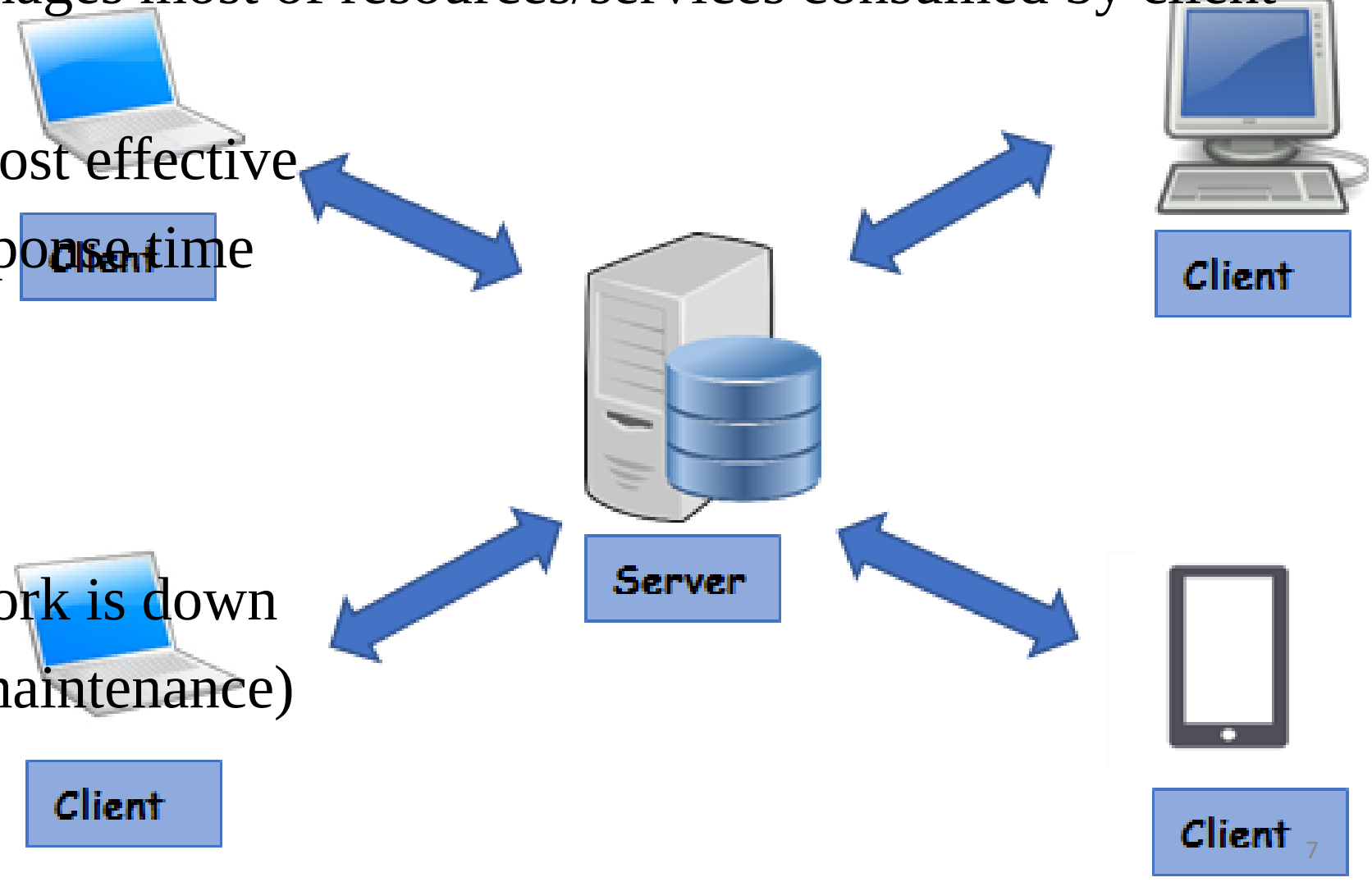
- server hosts, delivers, manages most of resources/services consumed by client

Advantages:

- Data backup is easy and cost effective
- Performance is better, response time
- Security is better.
- Scalability is easy.

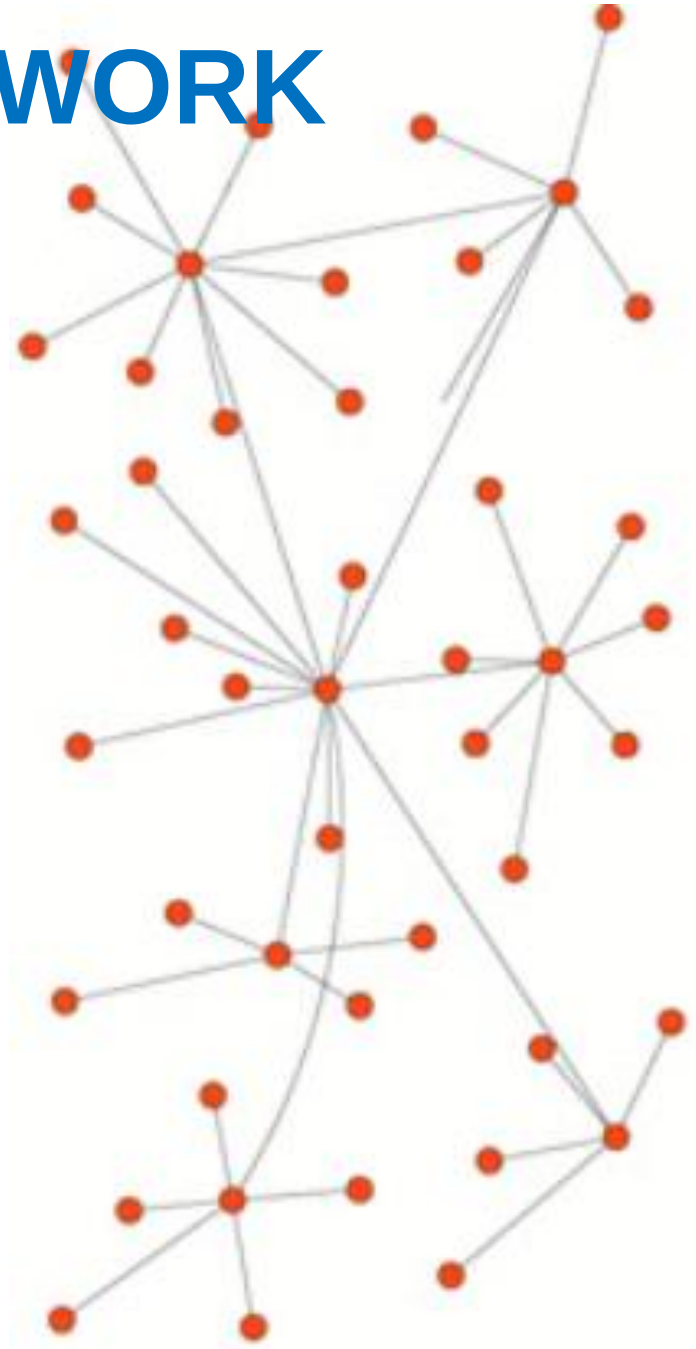
Disadvantages:

- Server failure entire network is down
- High cost for resources, maintenance)

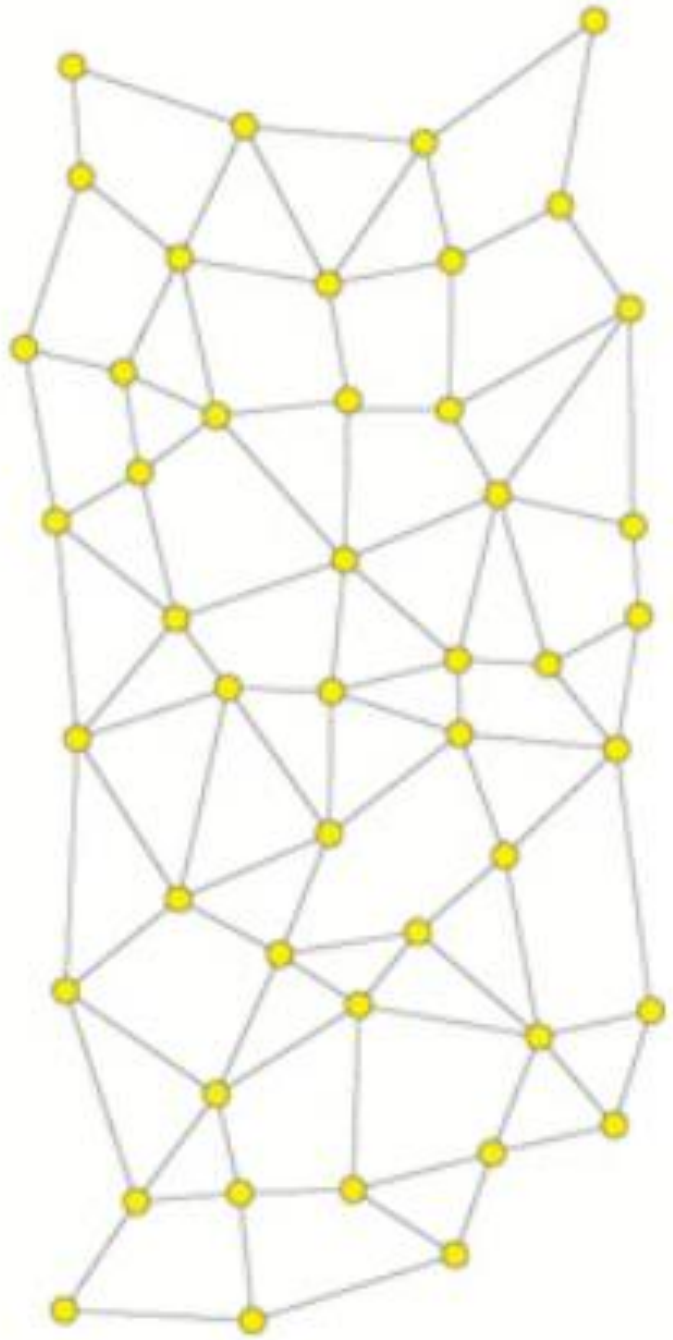


# 2.4. HYBRID NETWORK

- Decentralized network
- Distributed network



Decentralized network



Distributed network

# 3. BLOCKCHAIN NODES

3.1. INTRODUCTION

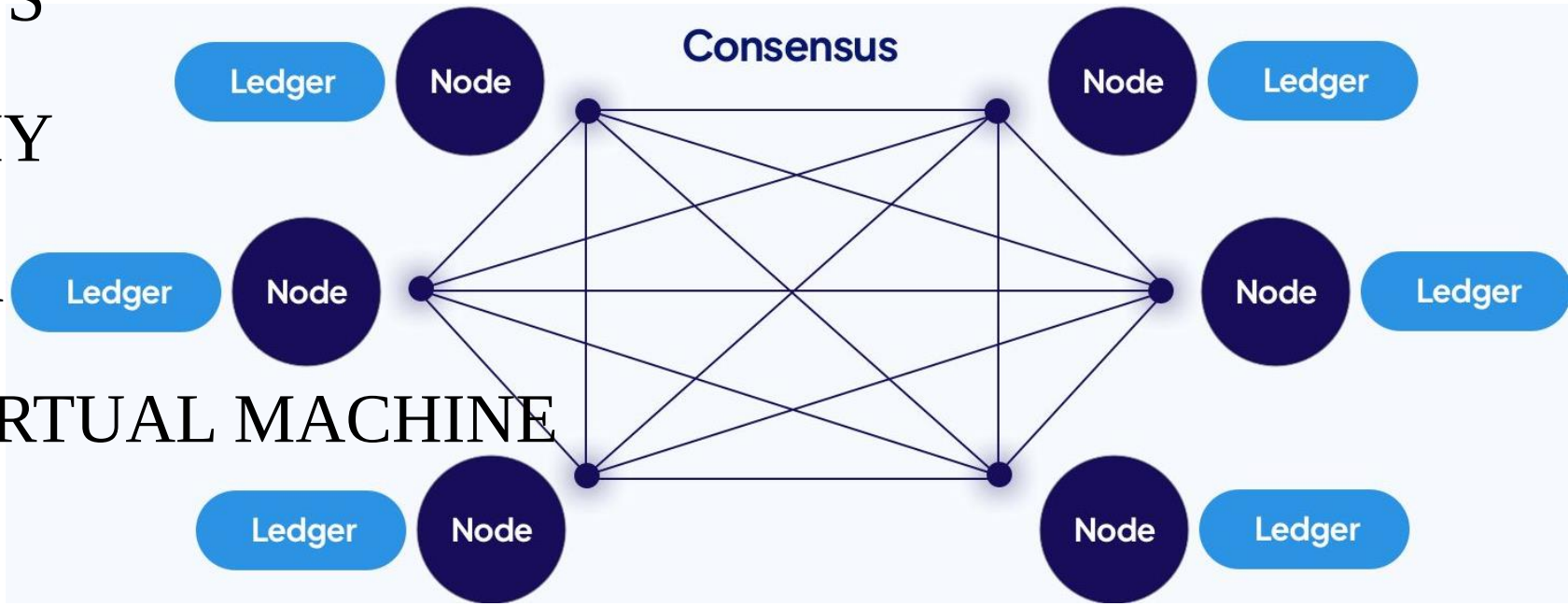
3.2. NODE ARCHITECTURE

3.3. NODE FUNCTIONS

3.4. NODE TAXONOMY

3.5. NODE PROVIDER

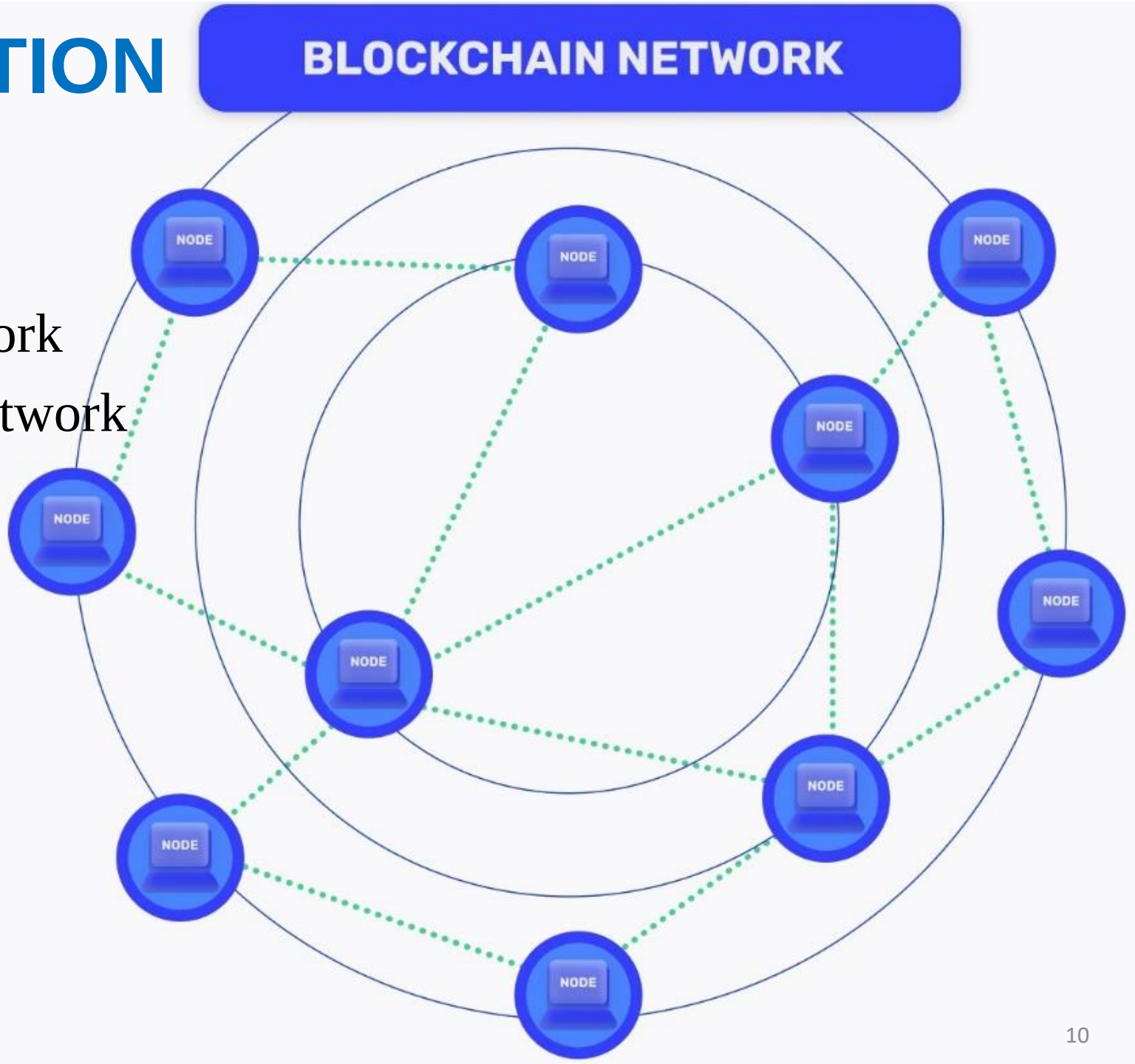
3.6. BLOCKCHAIN VIRTUAL MACHINE



# 3.1. INTRODUCTION

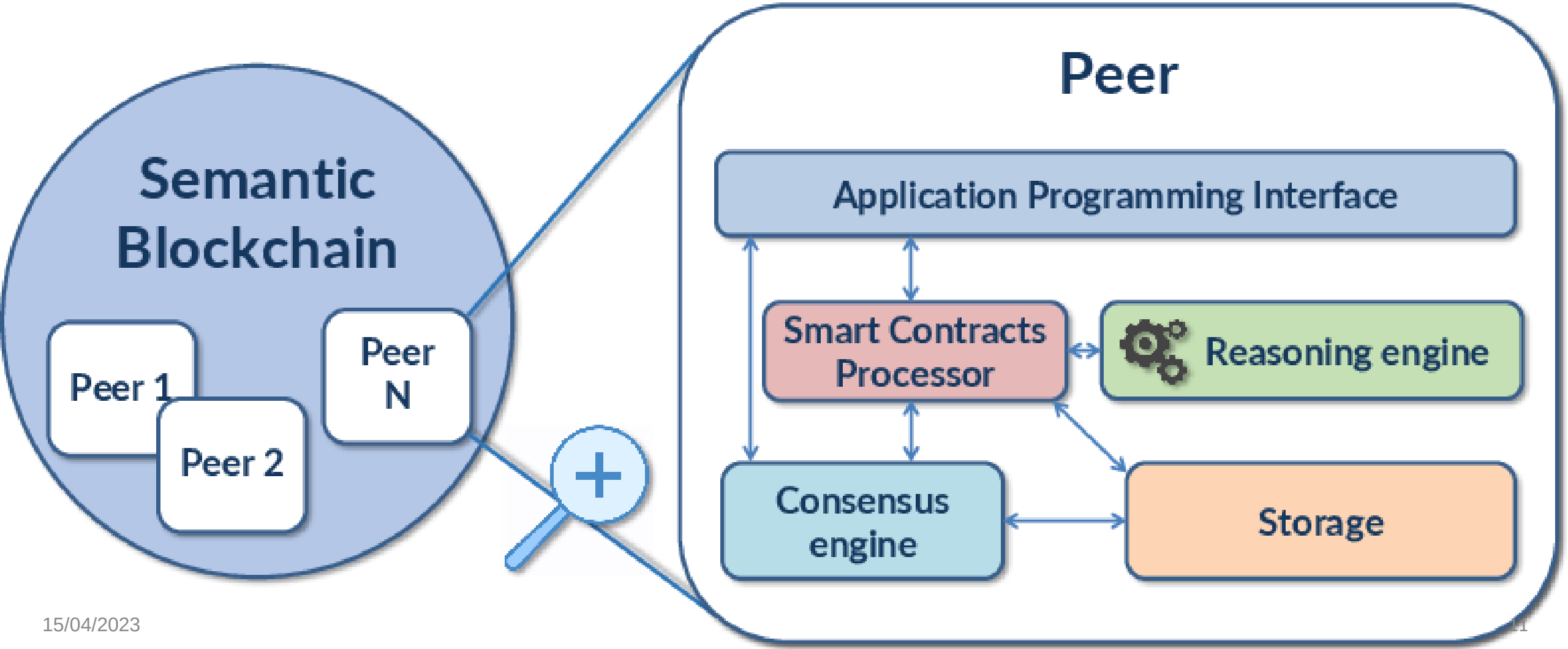
Node: devices & protocols:

- program running on computer
- connect with blockchain network
- carries out key functions of network
- allow application interact with blockchain



# 3.2. NODE ARCHITECTURE

Node layer:

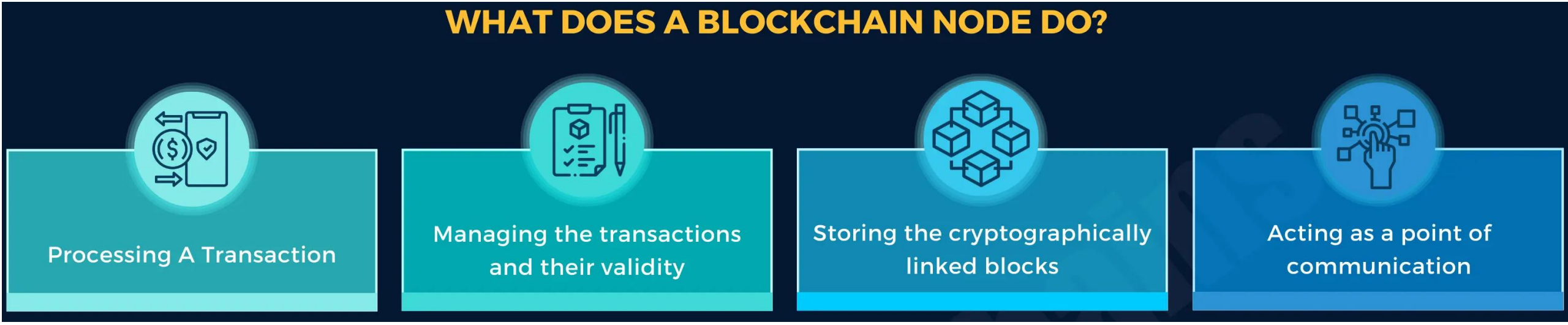


# 3.3. NODE FUNCTIONS

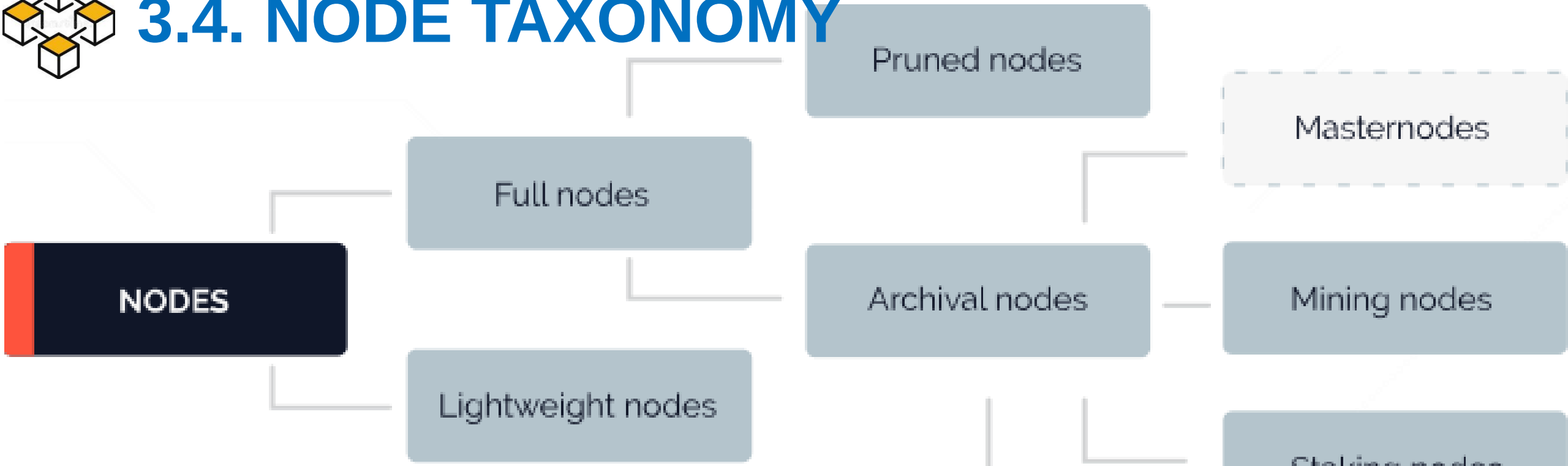
Node functions: maintain consensus

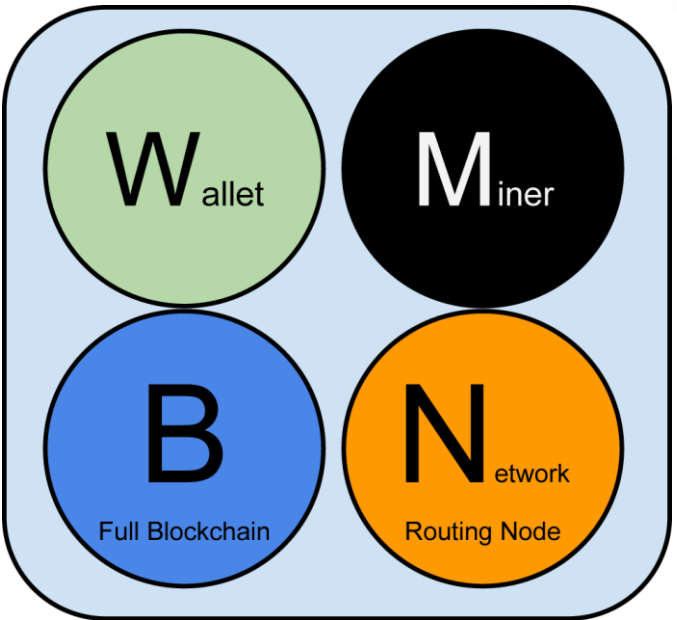
- Manage (keep, sync) ledger
- Transaction processing: accept, reject proposals, block creation
- Accessibility: access ledger data.

Client: request to perform node functions

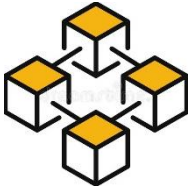


# 3.4. NODE TAXONOMY




- Light nodes: sync block headers
- Full nodes: keep all transactions



# 3.4. NODE TAXONOMY

- Light nodes: sync block headers
- Full nodes: keep all transactions

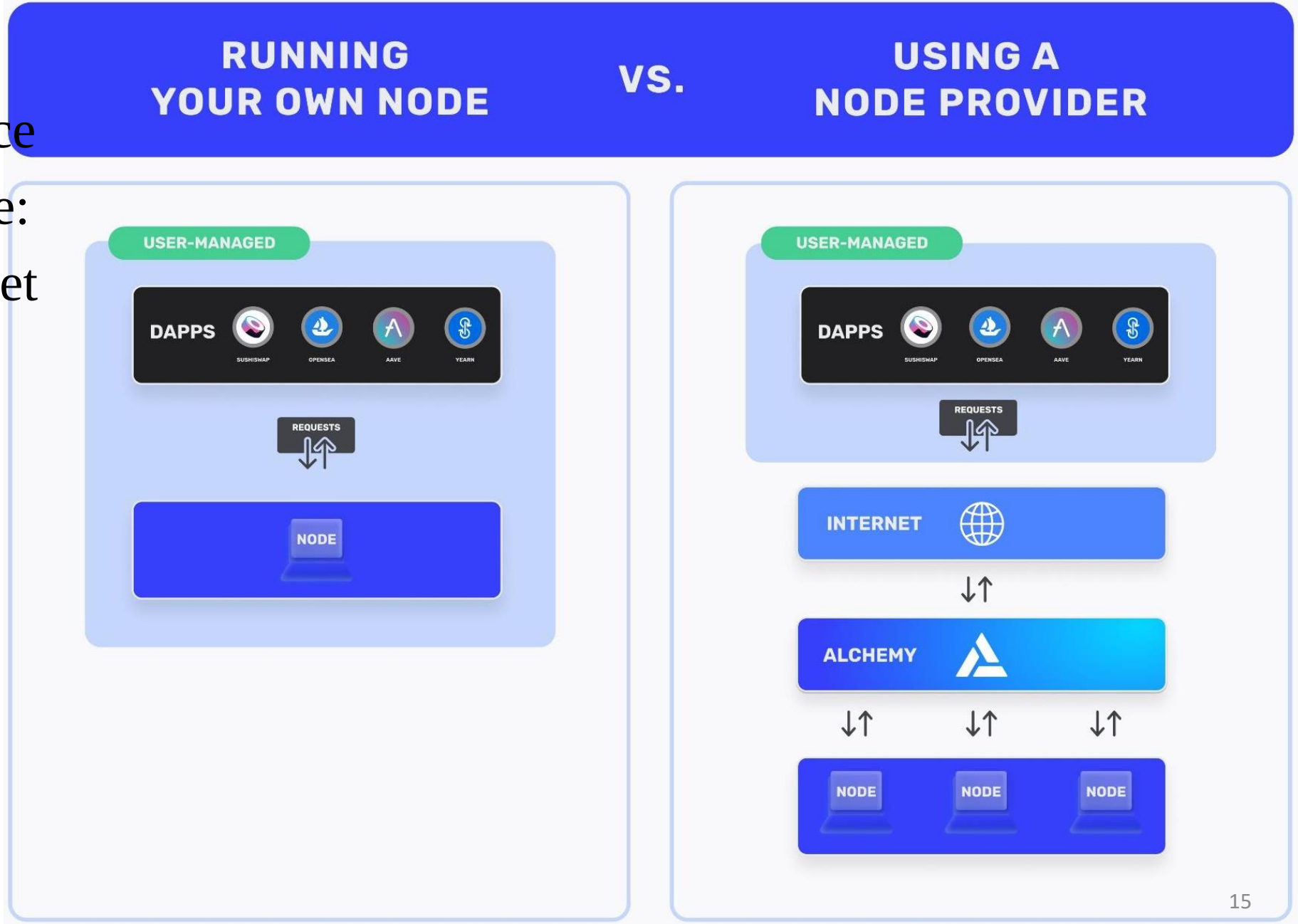
| | Pros | Cons |
|---|---|---|
| **Light nodes** | • Portable<br>• Resource-efficient<br>• User-friendly | • Do not validate the network<br>• Do not propagate blocks<br>• Do not maintain consensus<br>• Less secure |
| **Full nodes** | • Validate the network<br>• Propagate blocks<br>• Maintain consensus<br>• More secure | • Resource-heavy<br>• Harder to maintain<br>• Less user-friendly |
| **Pruned** | Flexible storage | Need to revalidate old blocks |
| **Archive** | Carry full history | Resource and storage heavy |
| **Mining** | • Easily trackable involvement<br>• Can pool with others to increase reward rate | • High and wasteful energy consumption<br>• High equipment cost and barrier to entry |
| **Staking** | • Low barrier to entry<br>• Low energy consumption | • Reward system based on luck<br>• Low transparency in staking pools |
| **Masternodes** | • Balanced network benefits and rewards<br>• Lower maintenance costs | • High initial investment<br>• Difficult setup process |

# 3.5. NODE PROVIDER

Node provider: Blockchain-as-a-service

Problems to run a node:

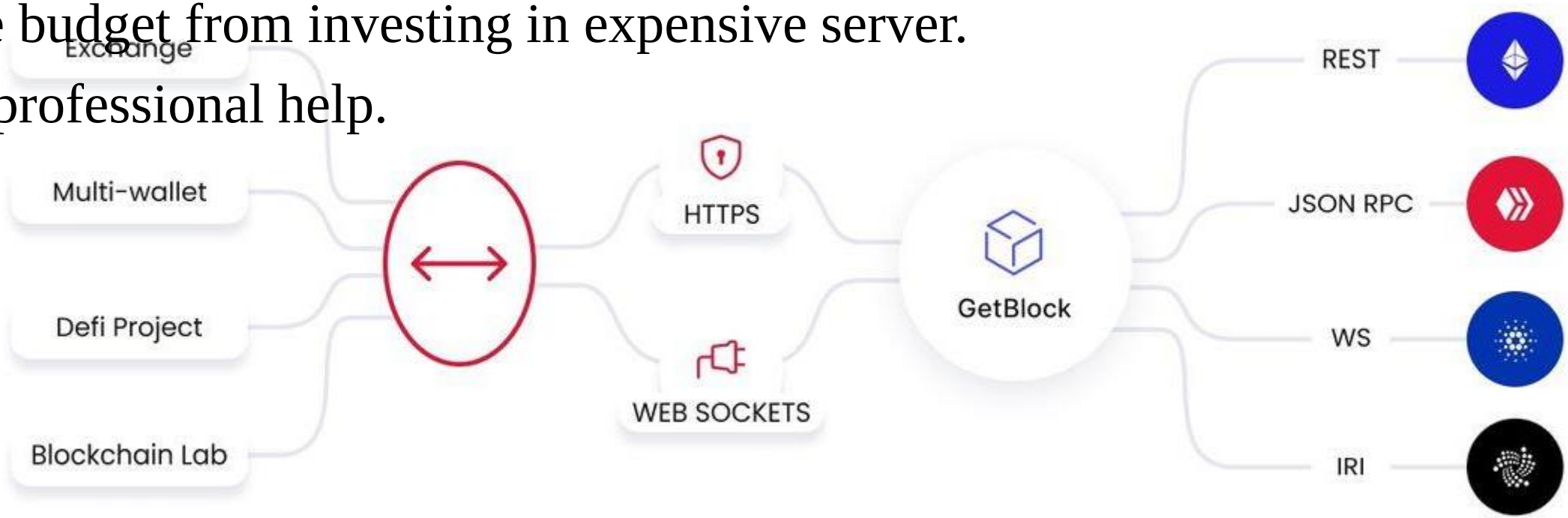
- Take a long time to set up (weeks)
- Hard to manage.
- Hard to scale



# 3.5. NODE PROVIDER

Node provider benefits:

- Deploy well-maintained and regularly-updated nodes.
- Leverage historical transaction data
- Scale reliably
- Forget about exceptions/desynchronization issues.
- Save budget from investing in expensive server.
- Get professional help.



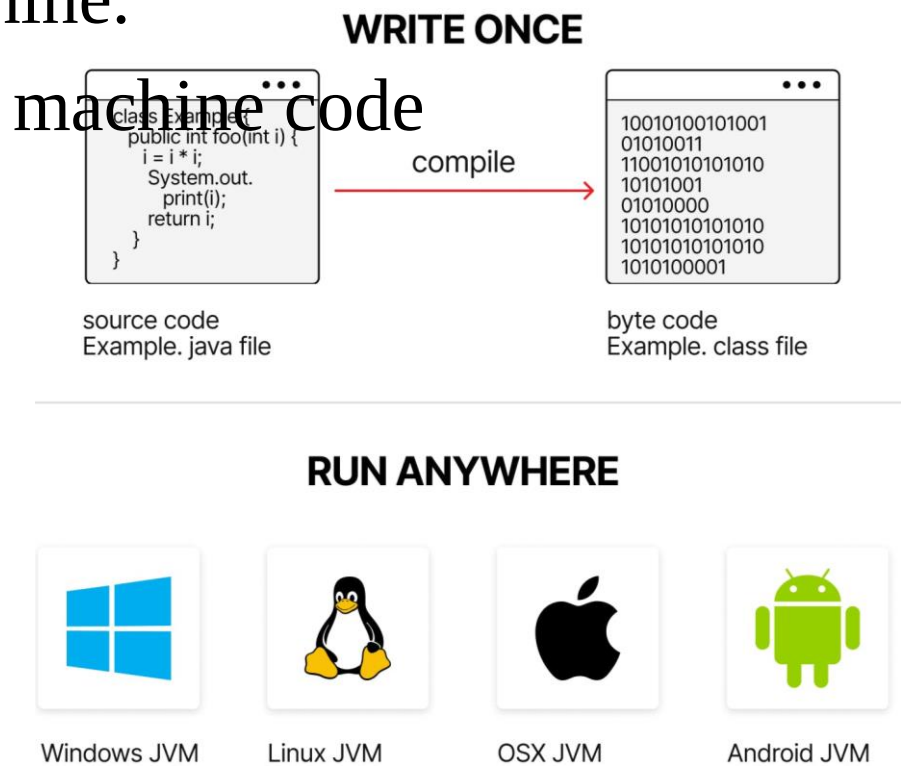
# 3.6. BLOCKCHAIN VIRTUAL MACHINE

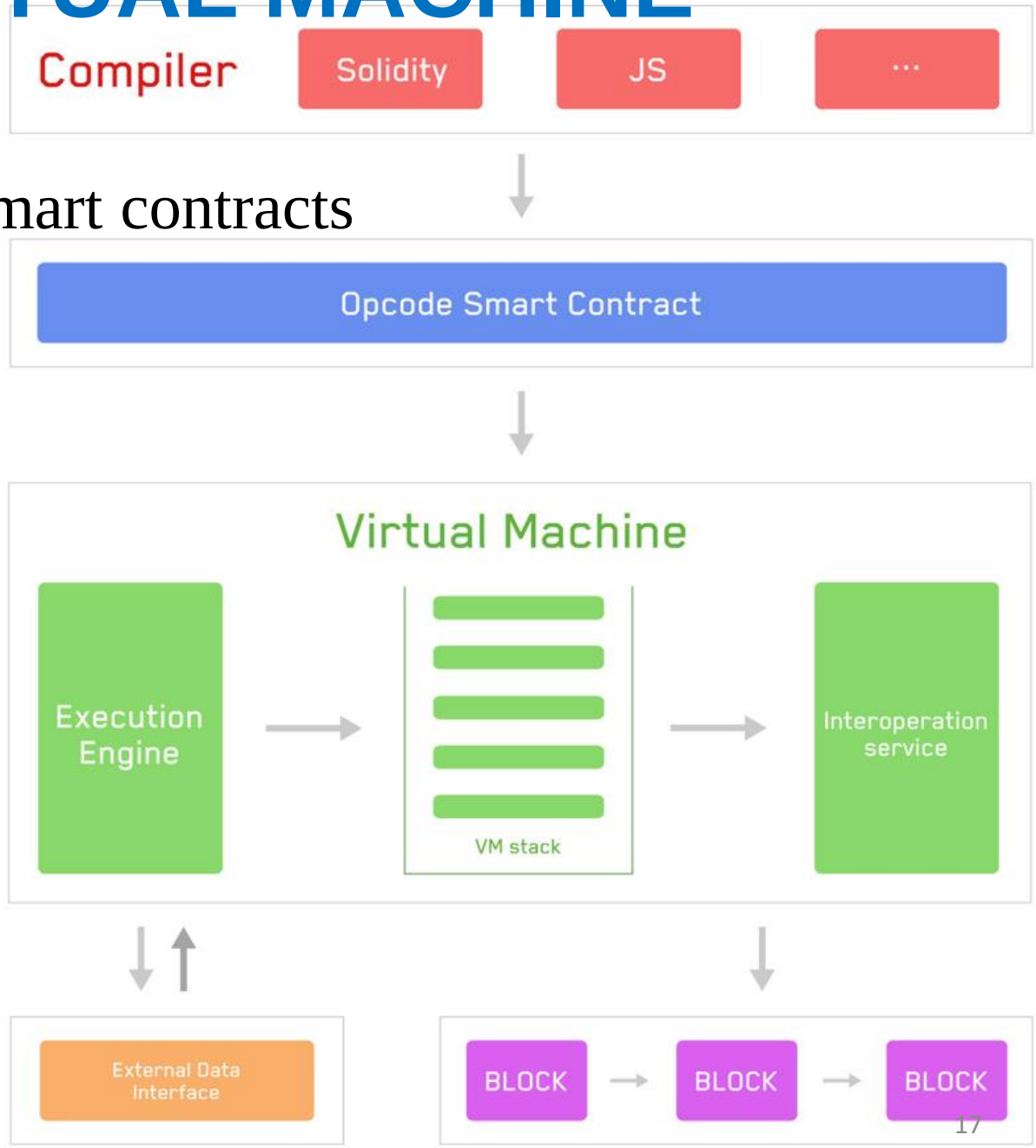
Blockchain Virtual Machine:

- environment for creating and deploying smart contracts
- as ‘virtual computer’ / software platform.

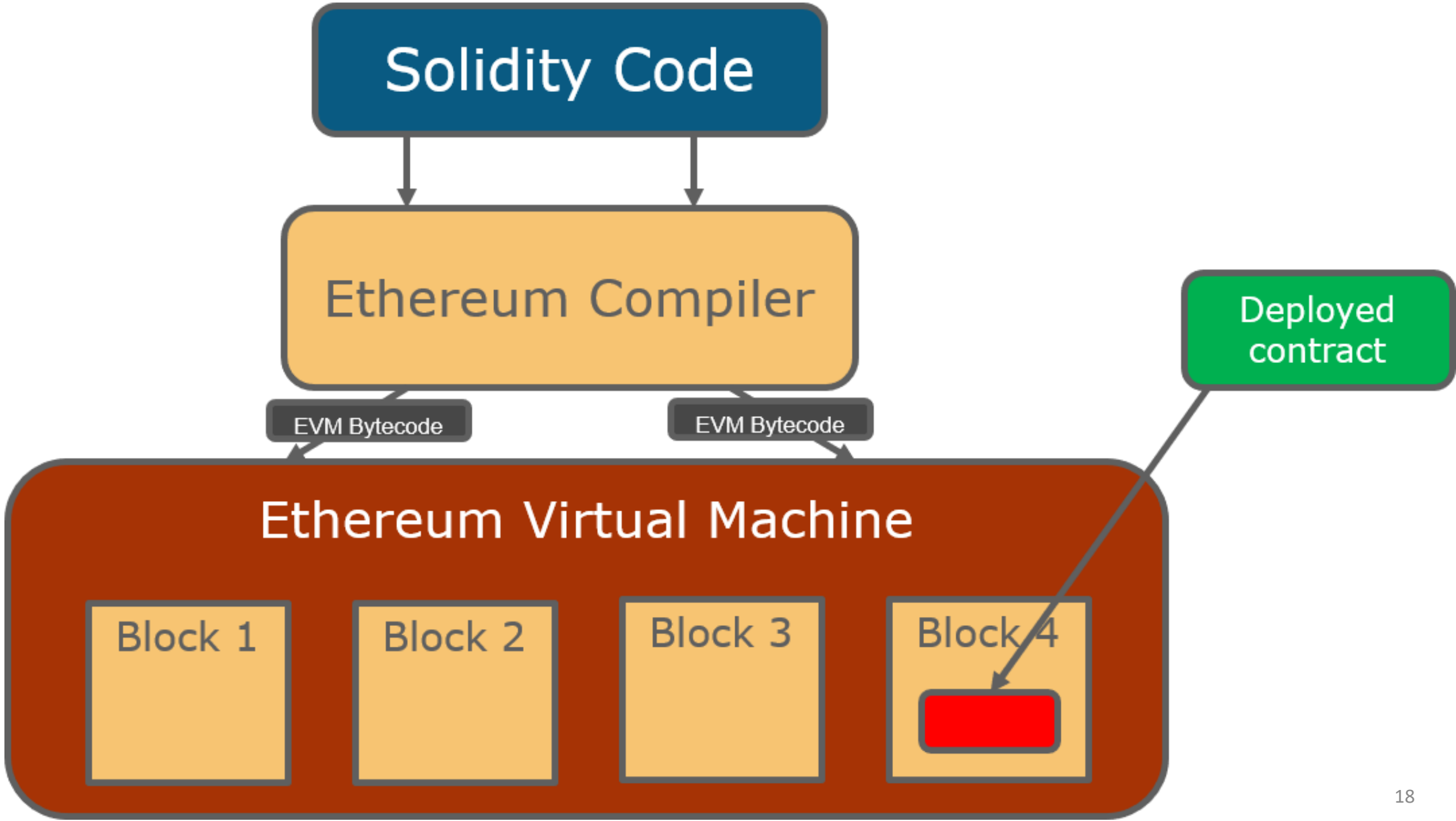
Functions:

- State machine.
- Executing machine code



# 3.6. BLOCKCHAIN VIRTUAL MACHINE



# 4. BLOCKCHAIN PROTOCOLS

4.1. INTRODUCTION

4.2. BLOCKCHAIN PROTOCOLS

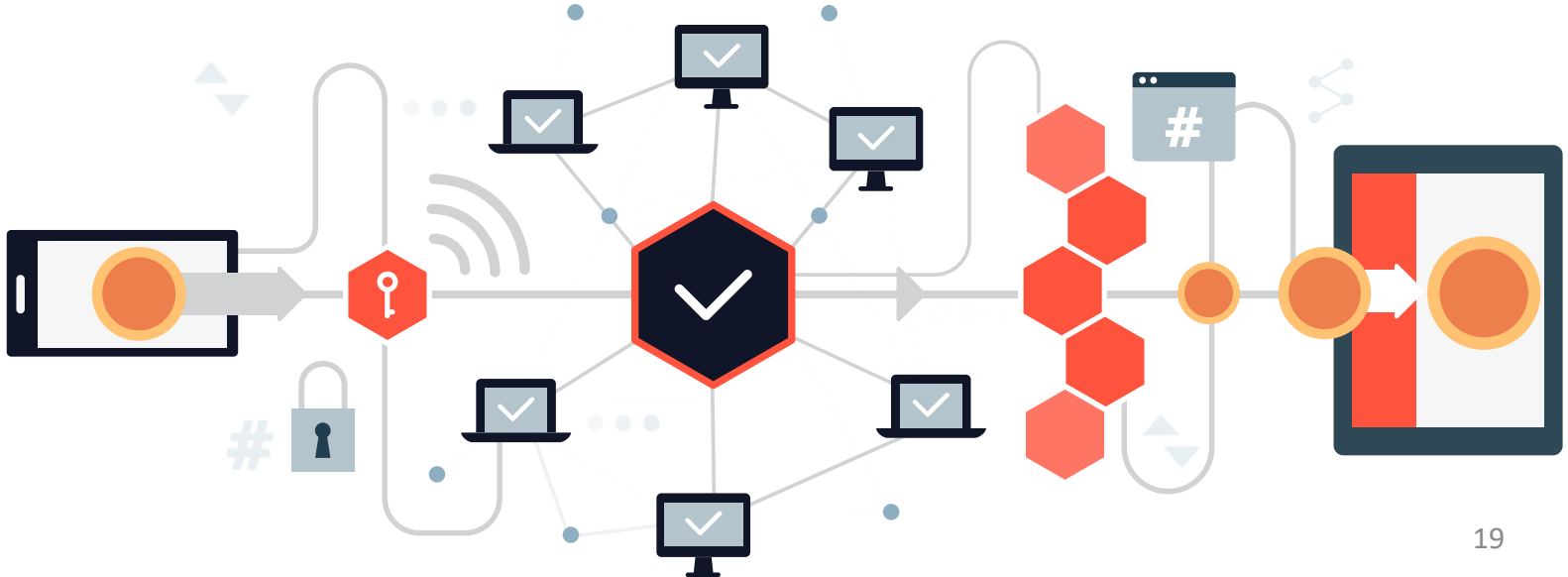
4.3. BLOCKCHAIN CONCENSUS

4.4. TYPE OF CONCENSUS

4.5. PROOF OF WORK

4.6. PROOF OF STAKE

4.7. BLOCKCHAIN FORK



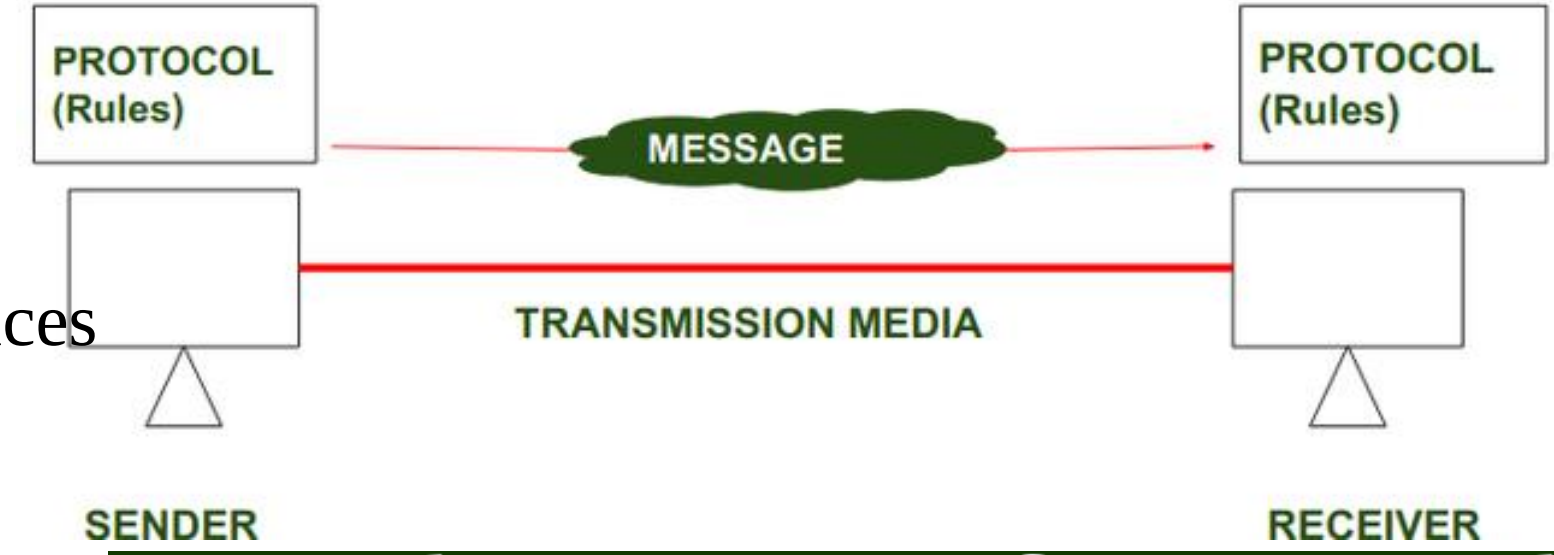
# 4.1. INTRODUCTION

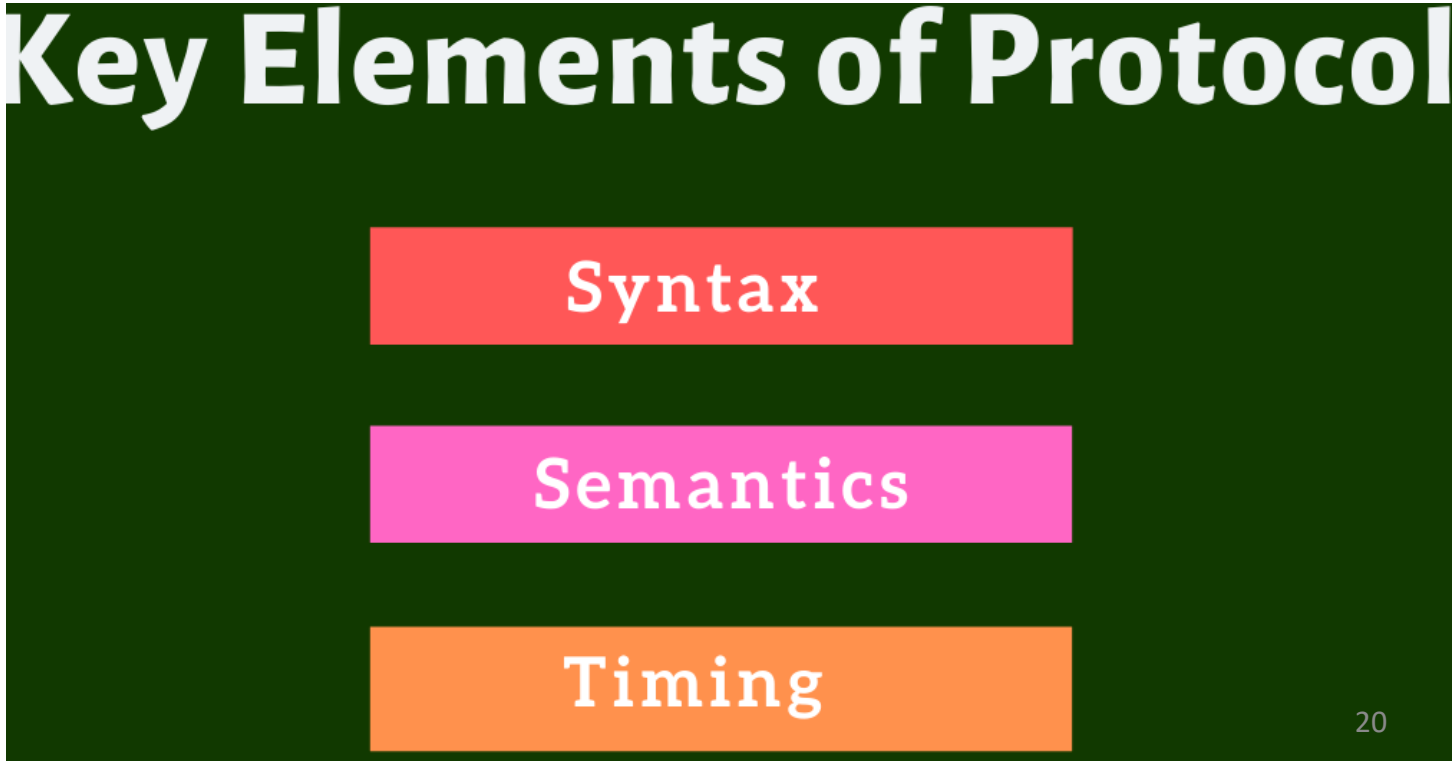
Protocols:

- Set of rules
- Communication between devices
- Exchange of data.

Levels of a Protocol:

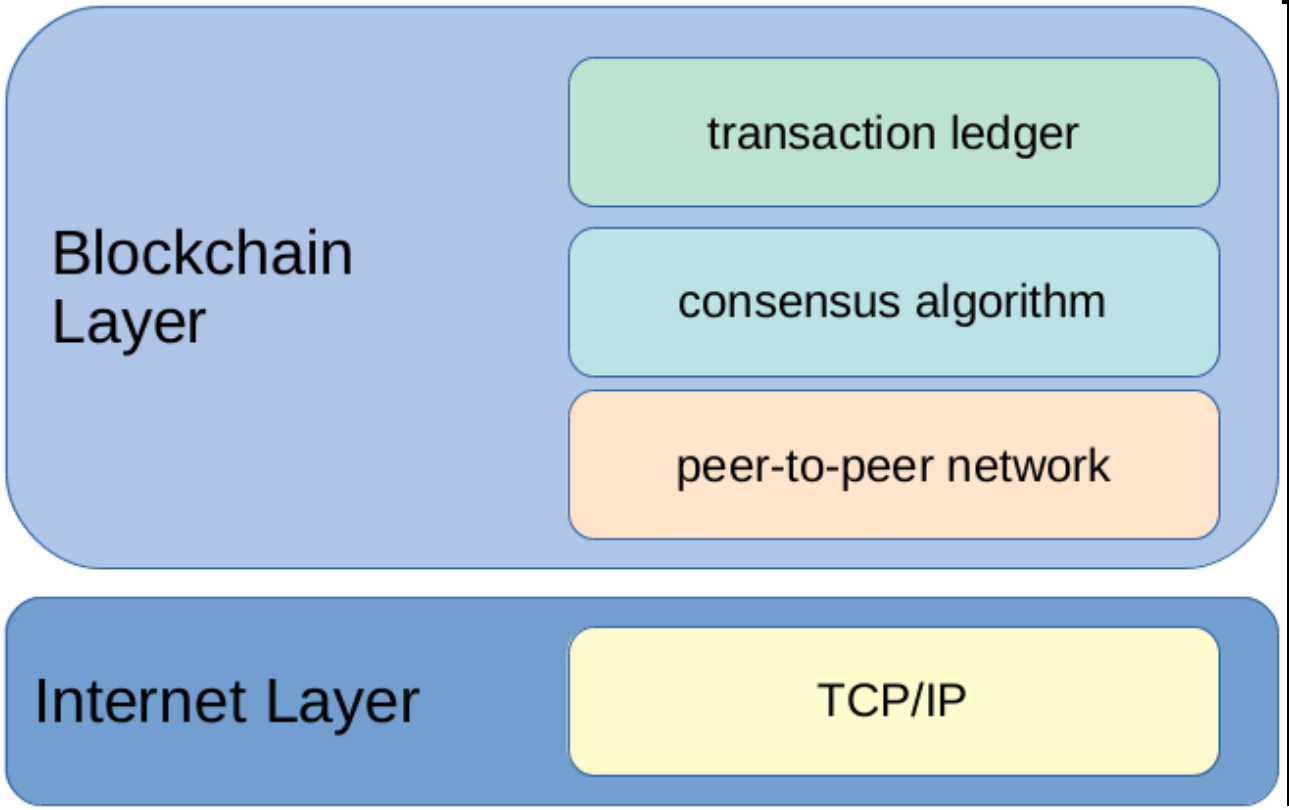
- Hardware Level:
- Software Level:
- Application Level:

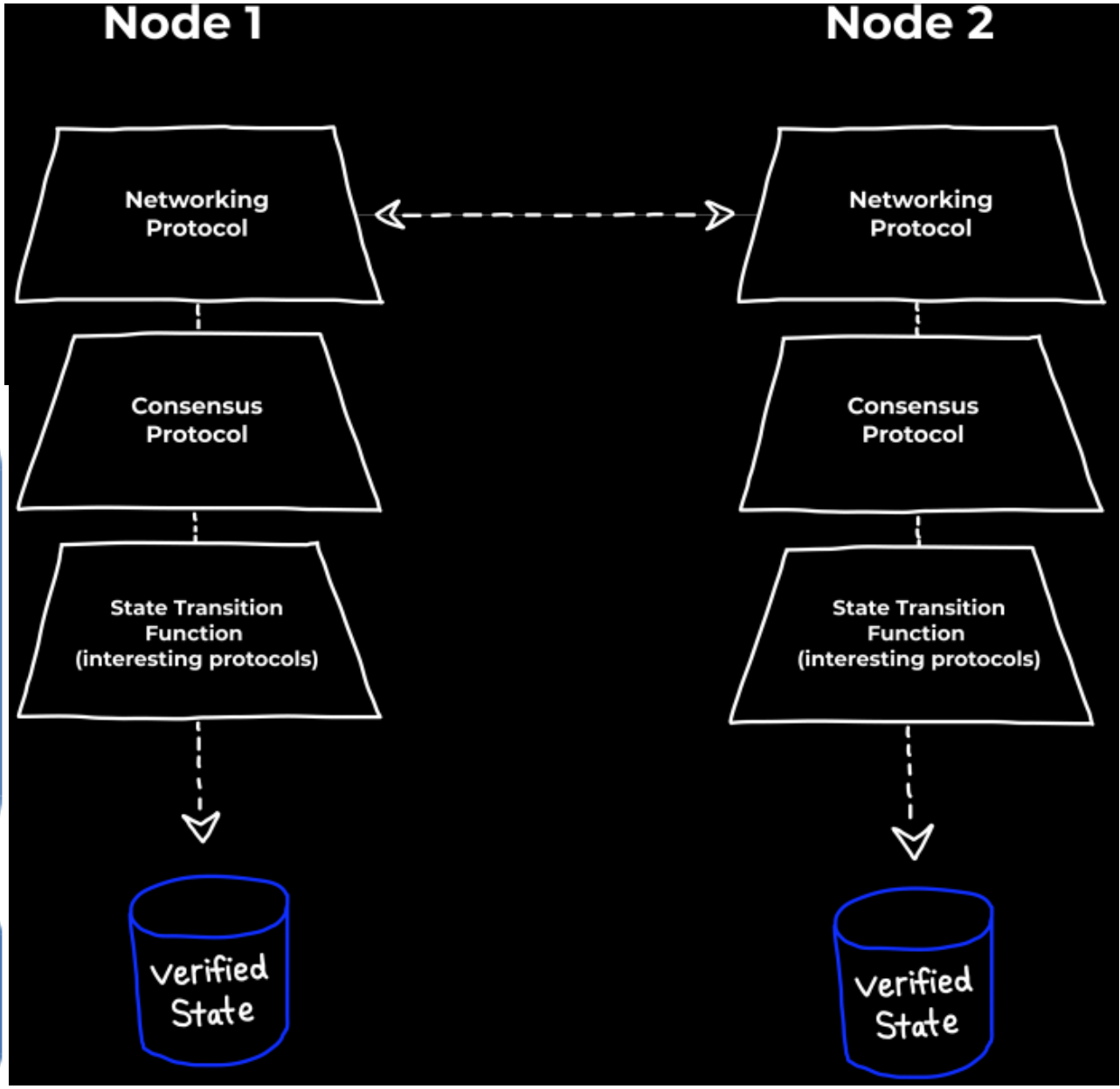


# 4.2. BLOCKCHAIN PROTOCOLS

- Components
- Fork
- Governance



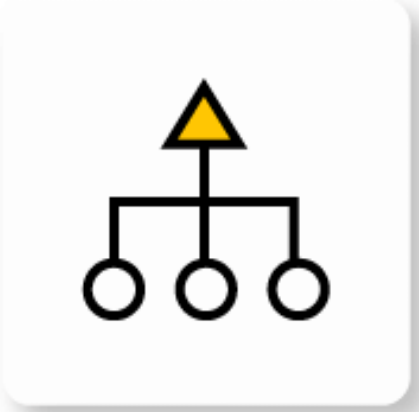
# 4.3. BLOCKCHAIN CONCENSUS

Concensus:

- Multiple processes maintaining common state.
- Classical problem in distributed computer systems.

Blockchain concensus:

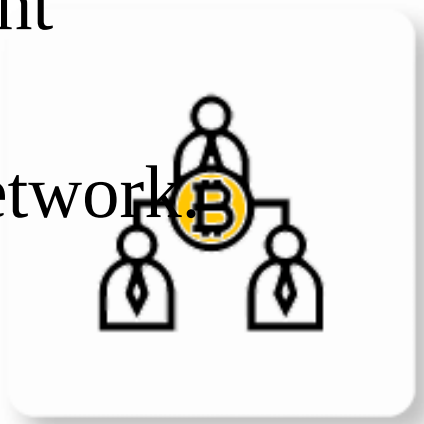
- A procedure, peers reach agreement present state of data in network.
- Establish reliability and trust in network.
- Not concensus: Fork



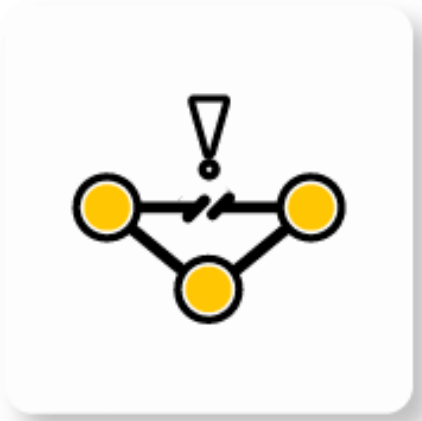
Unified Agreement

Align Economic Incentive

Fair and Equitable



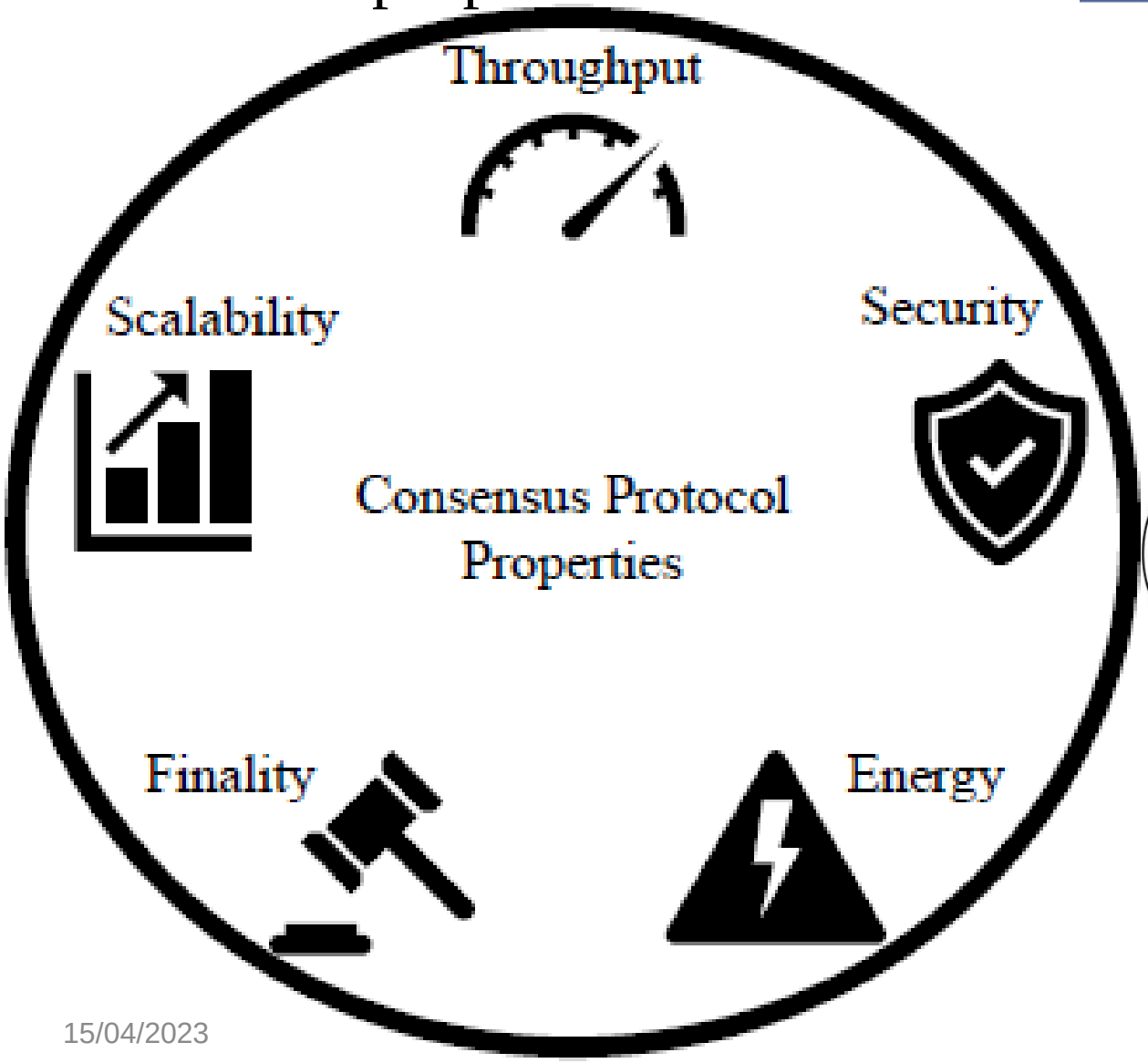
Prevent Double-Spending



Fault-Tolerant

# 4.3. BLOCKCHAIN CONCENSUS

Concensus properties:




## LEVELS OF BLOCKCHAIN CONSENSUS

*Full Decentralization*

CONSENSUS LEVEL 3

*Public ledger (trust & transparency)*

*Best for: public transactions*

*Pre-selected cluster*

CONSENSUS LEVEL 2

*- Private ledger (mutual trust)*

*Best for: cross-organizational transactions*

*Private*

CONSENSUS LEVEL 1

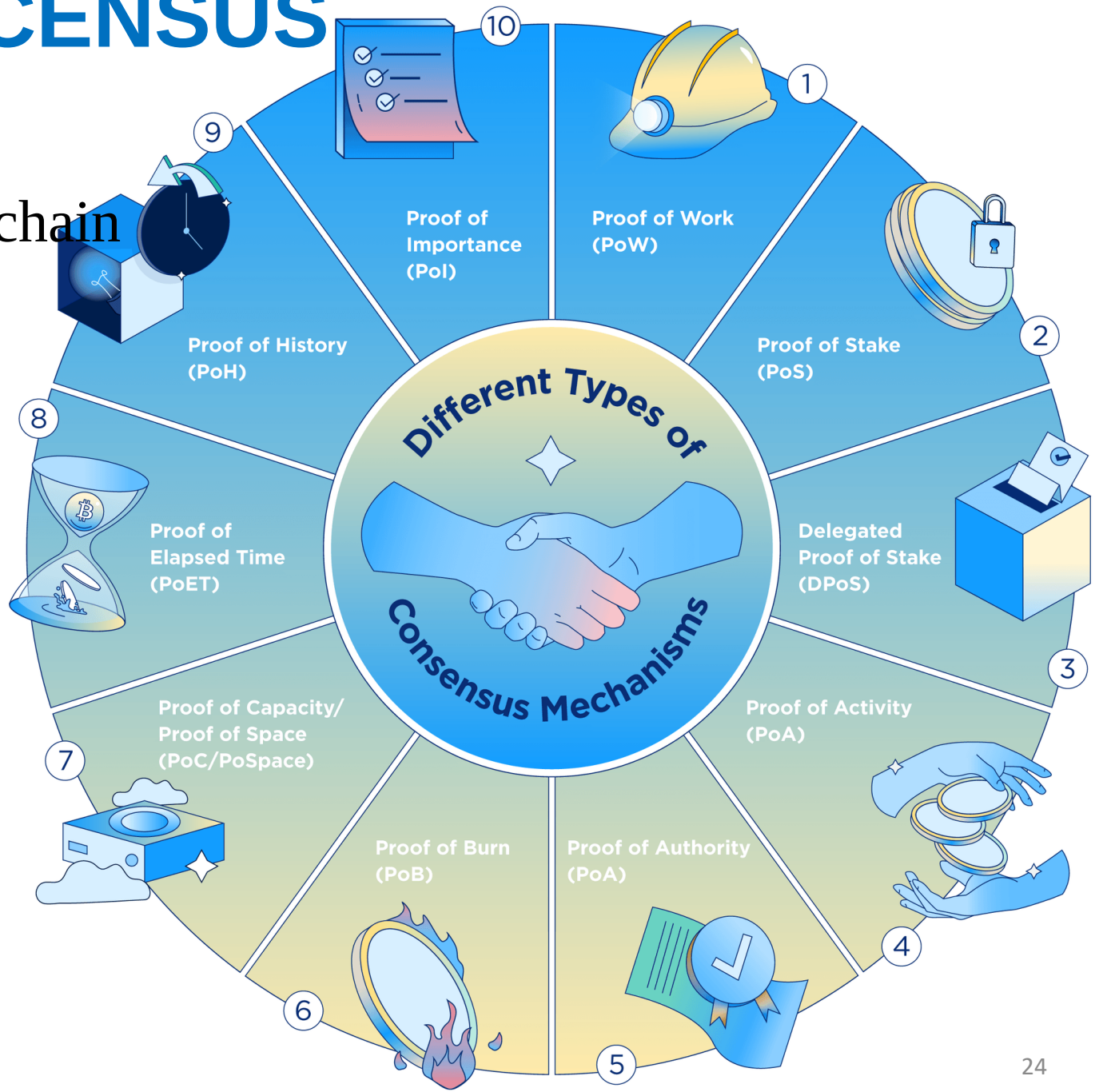
*Permissioned database (privacy)*

*Best for: internal information*

# 4.4. TYPE OF CONCENSUS

Concensus puposes:

- which node add new block to blockchain



# 4.5. PROOF OF WORK

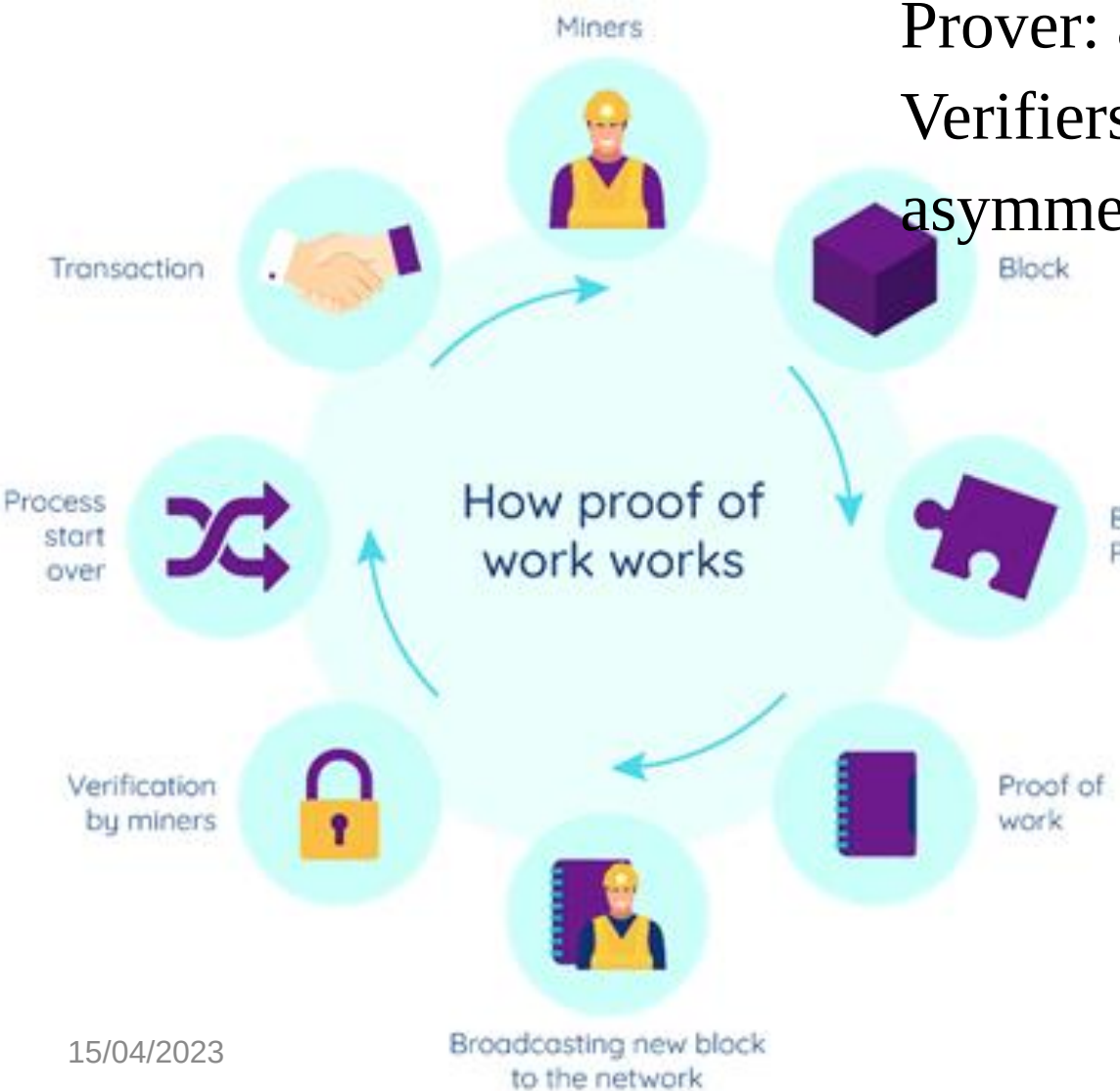
PoW: form of cryptographic proof

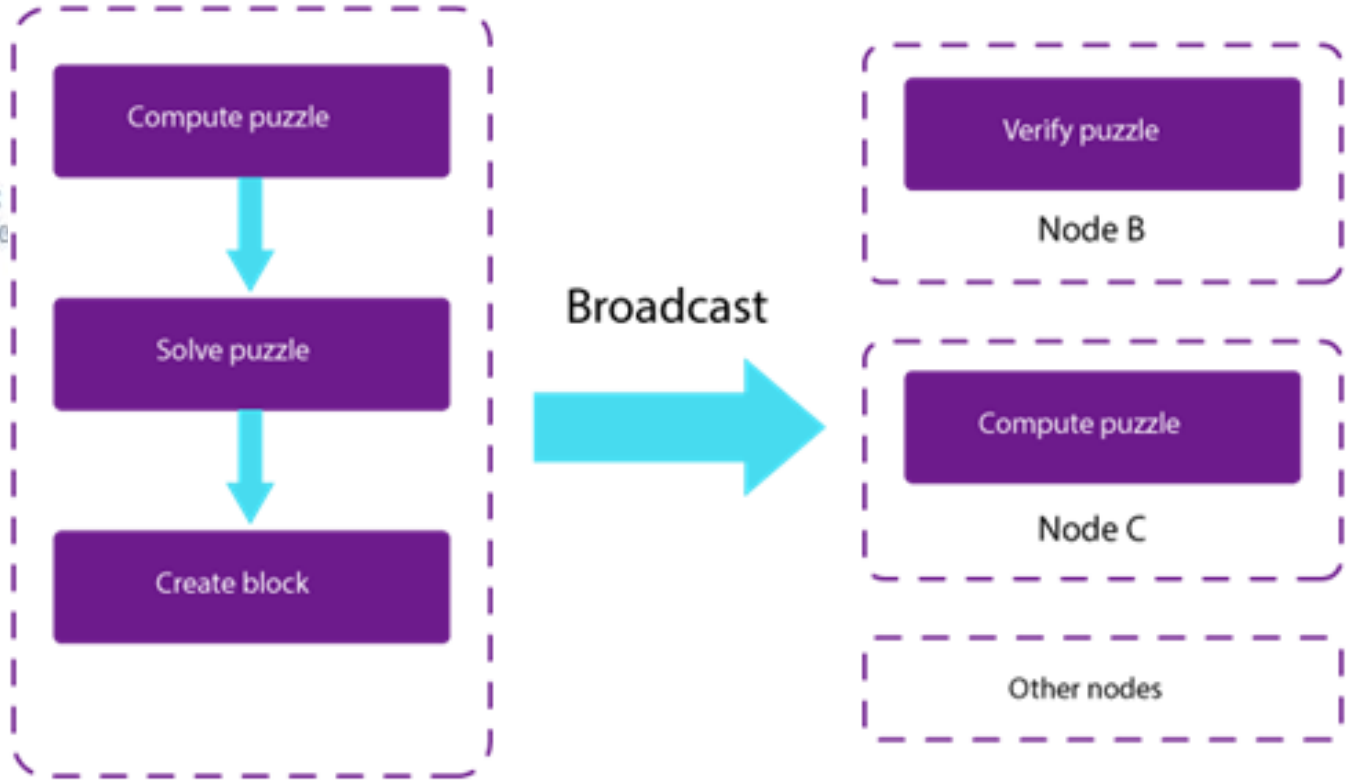
one party (prover) proves to others (verifiers)

Prover: amount of computational effort has been expended

Verifiers: can confirm this expenditure with minimal effort

asymmetry: prover work hard but the verifier check easily



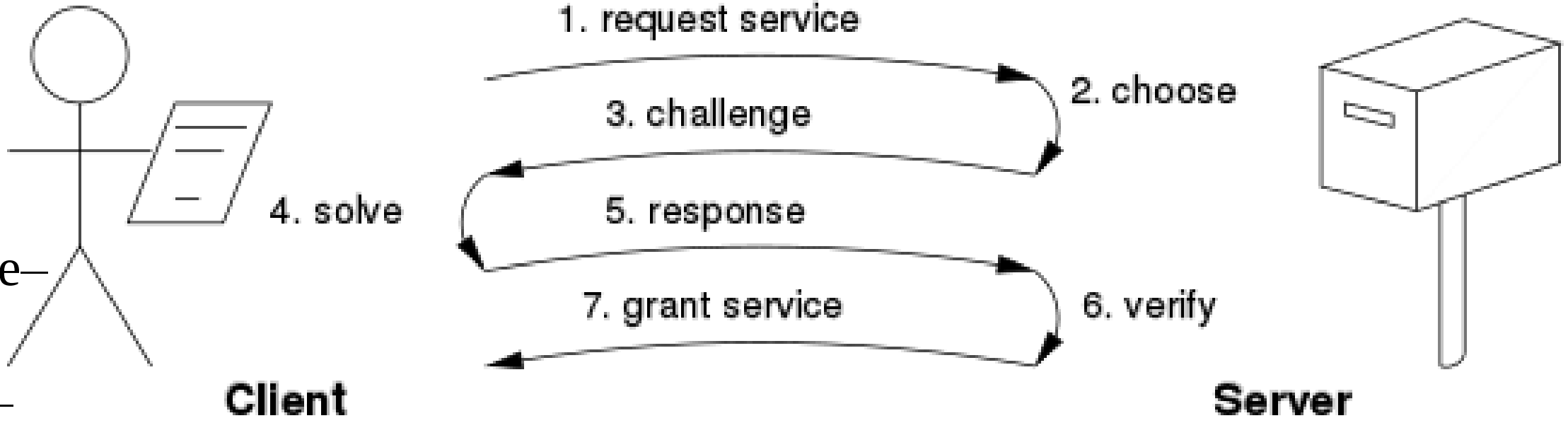



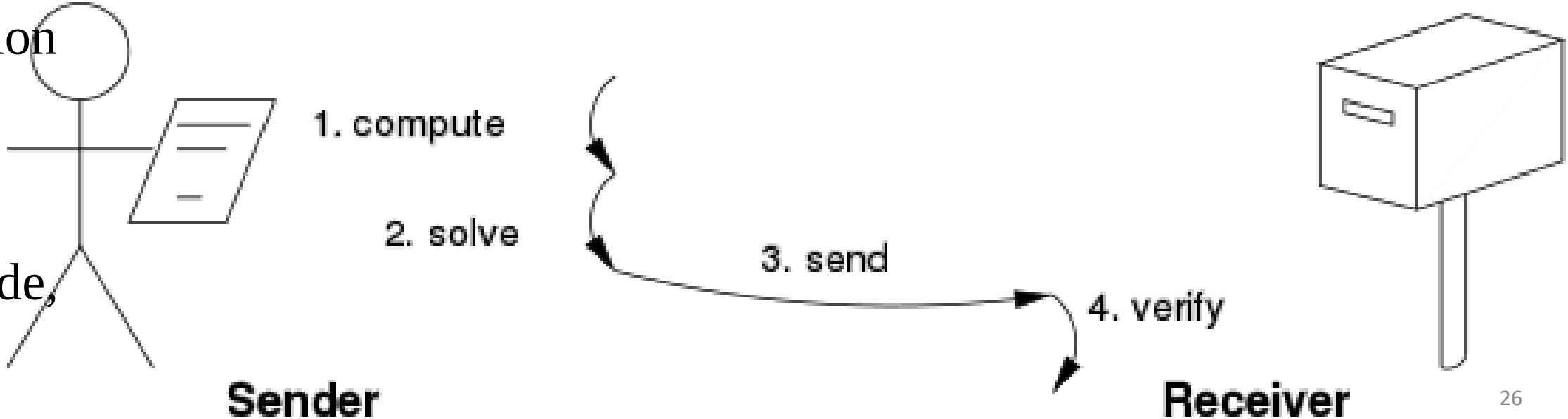
# 4.5. PROOF OF WORK

Classes of proof-of-work protocols:

- Challenge–response
- Solution–verification

Block creation nodes: mining node, miner.

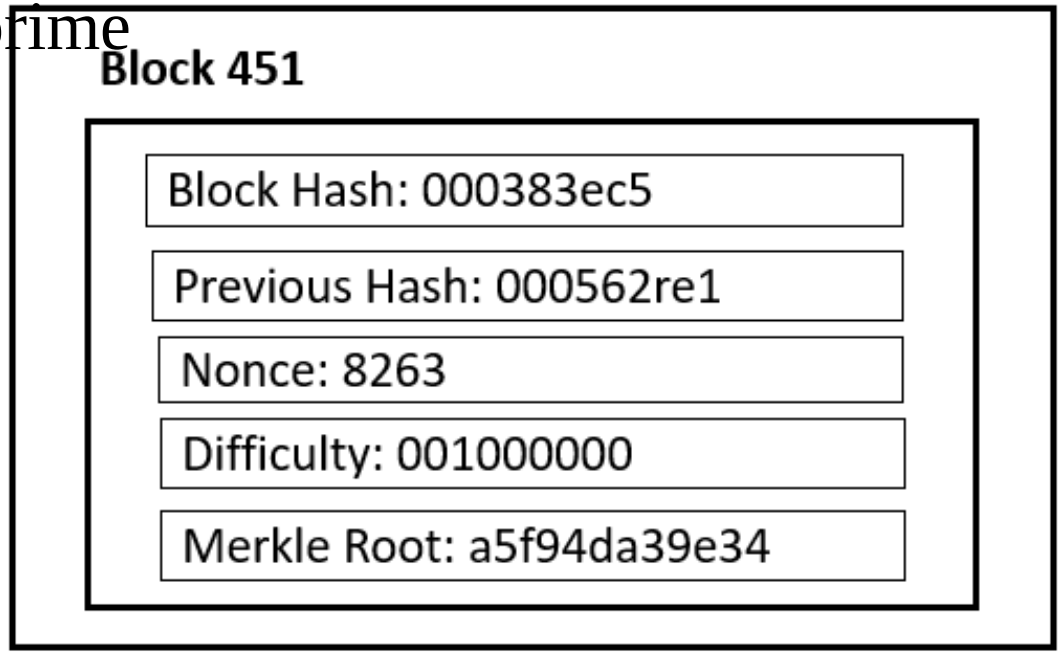


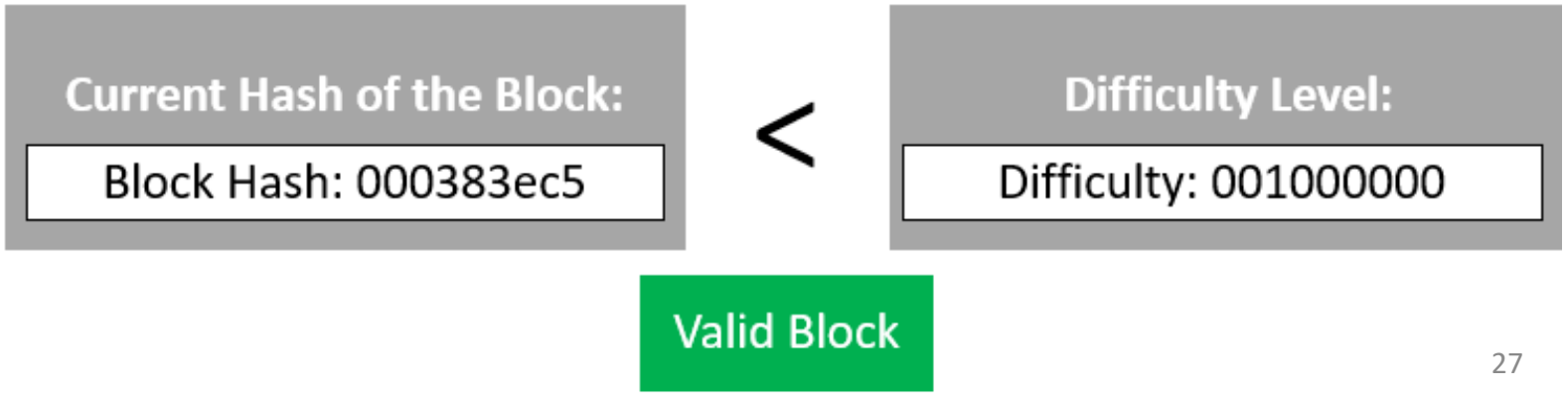



# 4.5. PROOF OF WORK

Known proof-of-work functions:

- Integer square root modulo a large prime
- Weaken Fiat–Shamir signatures
- Ong–Schnorr–Shamir signature broken by Pollard
- Partial hash inversion.
- Hash sequences; Puzzles
- Diffie-Hellman–based puzzle
- Moderate; Mbound
- Hokkaido; Cuckoo Cycle
- Merkle tree–based
- Guided tour puzzle protocol



# 4.6. PROOF OF STAKE

PoS: select validators in proportion to their quantity of holdings cryptocurrency.

## Proof of stake

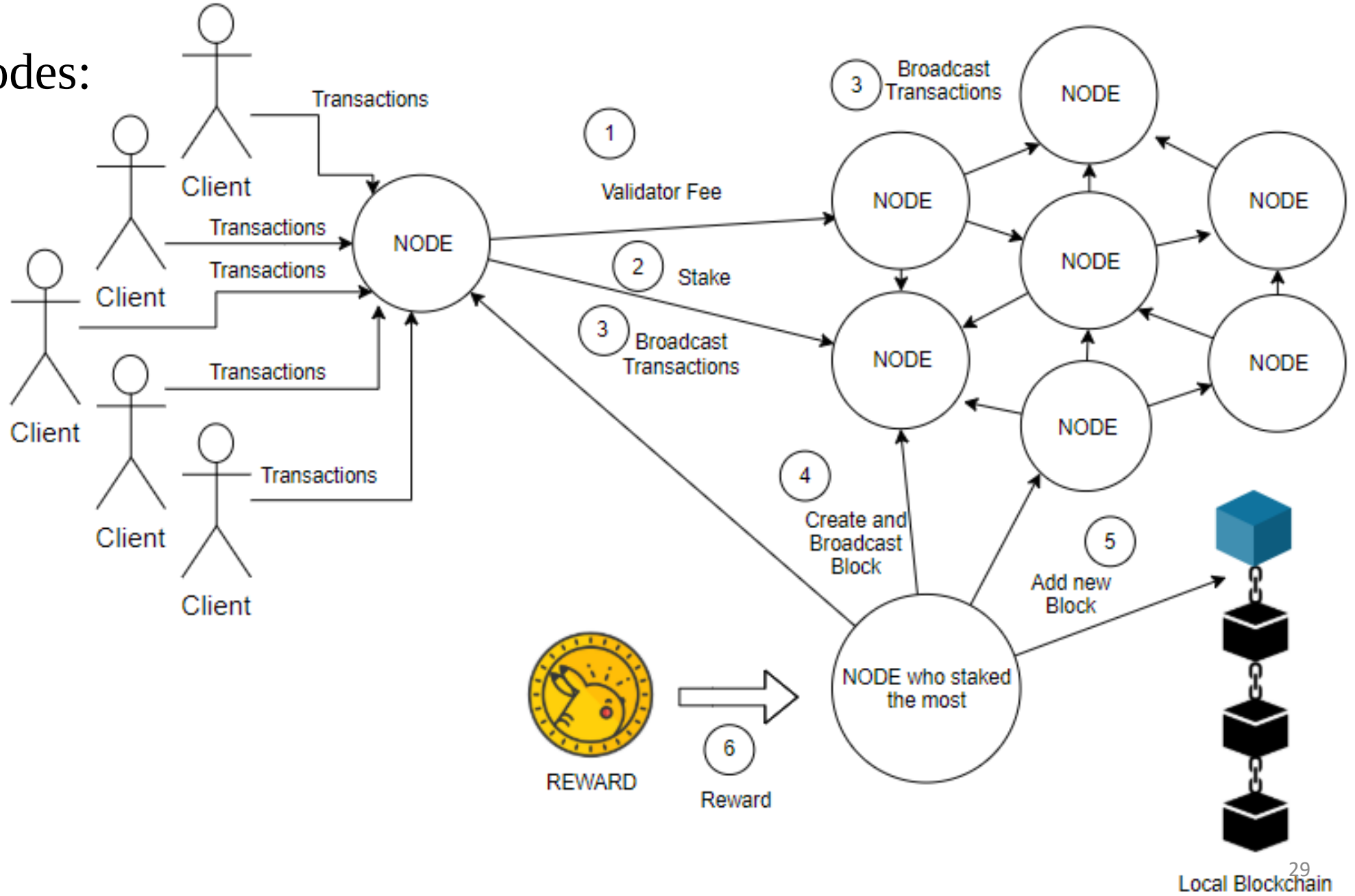
The probability of validating a new block is determined by how large of a stake a person hold.

The validators do not receive a block reward, instead they collect network fees as their reward.

Proof of stake systems can be much more cost and energy efficient than proof of work, but are less proven.

# 4.6. PROOF OF STAKE

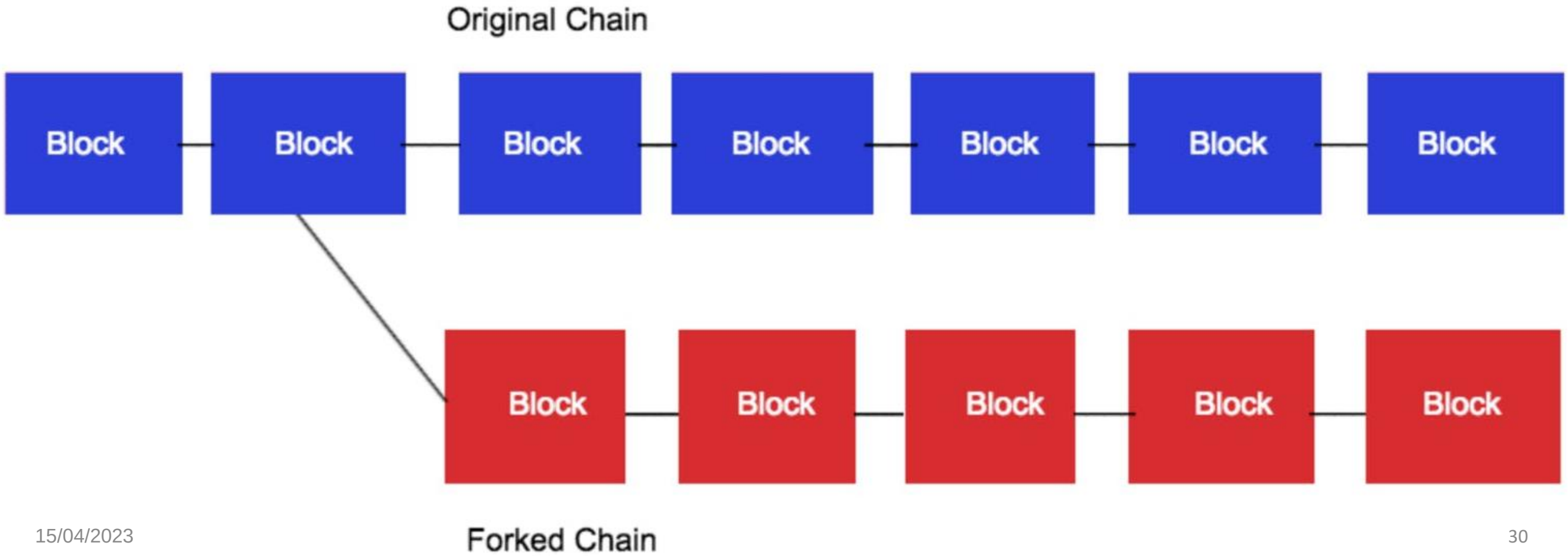
Block creation nodes: validator node



# 4.7. BLOCKCHAIN FORK

Fork:

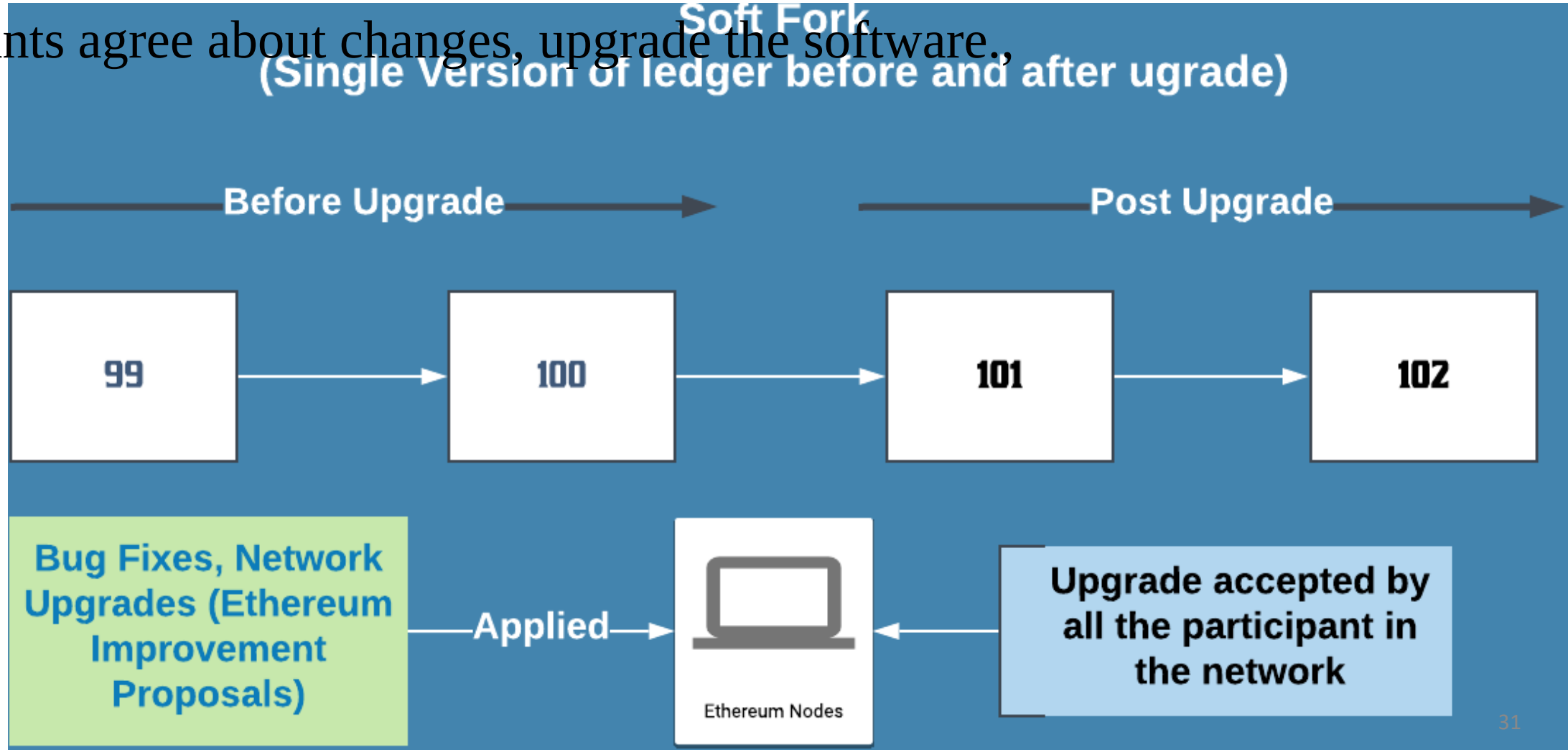
- Any changes done to rules, protocol and governing principle
- Bug fixes and updates (major/minor) applied



# 4.7. BLOCKCHAIN FORK

Soft Fork (network upgrade):

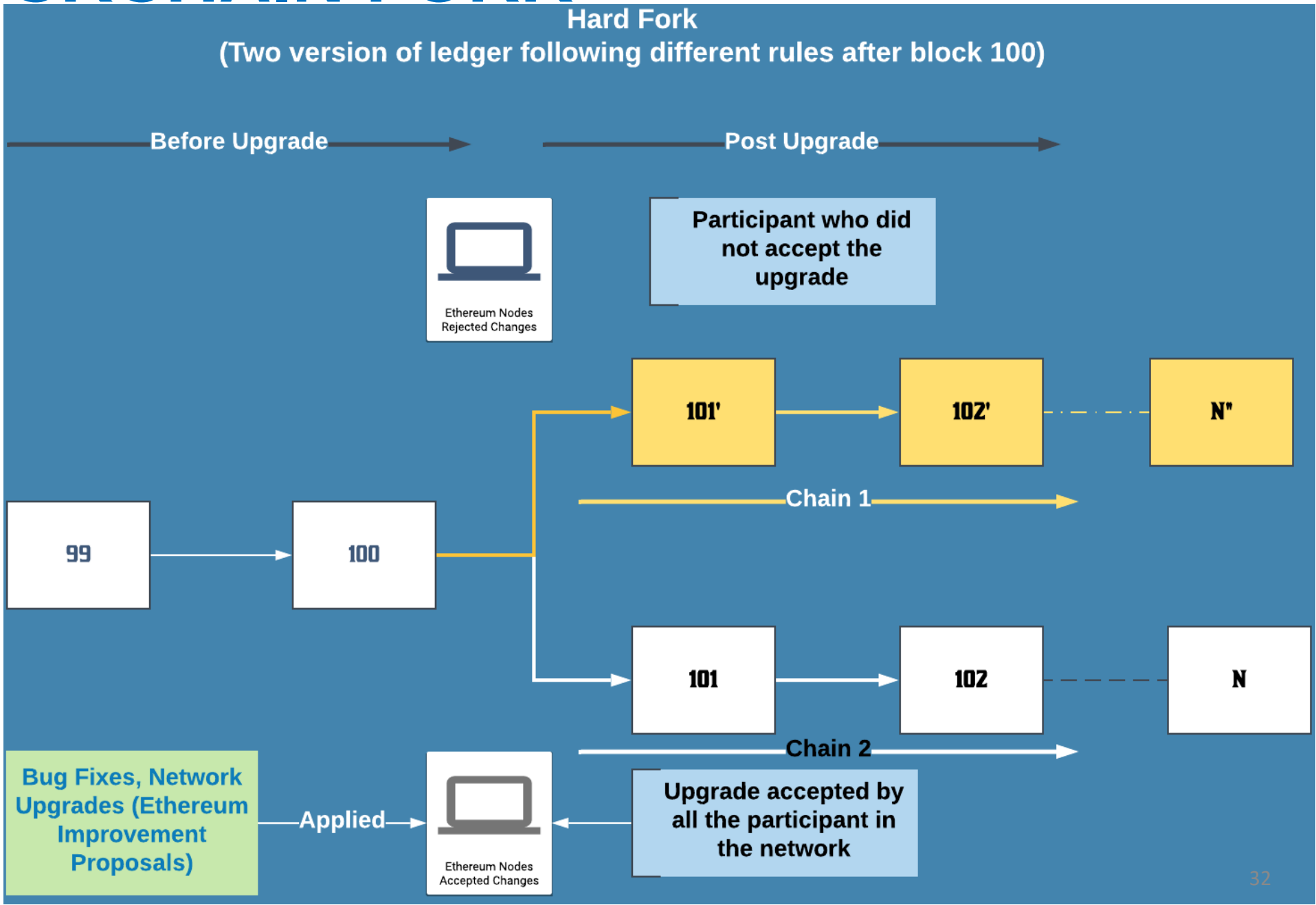
- Changes/upgrade are backwards compatible, support old, new rules.
- Participants agree about changes, upgrade the software.,



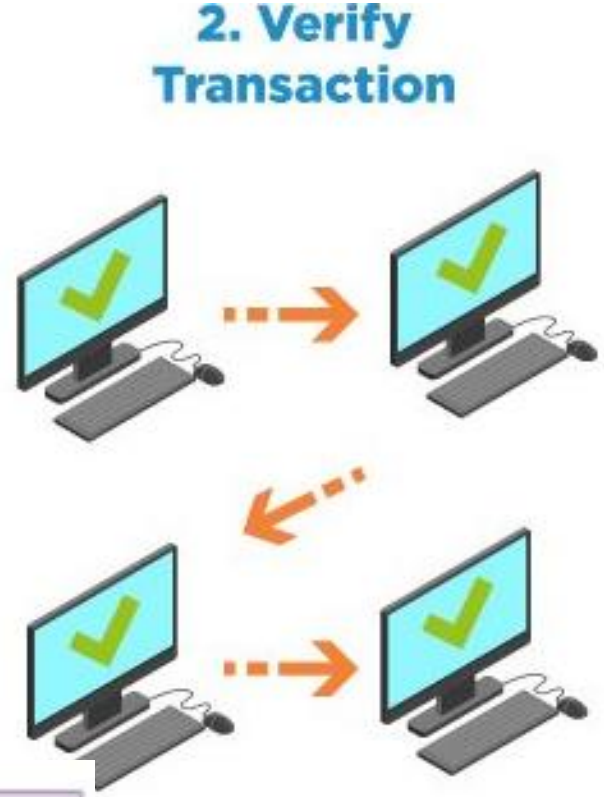
# 4.7. BLOCKCHAIN FORK

Hard Fork (split into two versions):

- Changes/upgrade are not agreed by all the participants
- Changes/upgrades are not backwards compatible.



# 5. BLOCKCHAIN WORKING PRINCIPLES



| | |
|---|---|
| Initiation and Broadcasting of Transaction | • Digital Signature<br>• Private/Public Keys |
| Validation of Transaction | • Proof of Work or<br>• Proof of Stake |
| Chaining of Blocks | • Hash Function |

# 5. BLOCKCHAIN WORKING PRINCIPLES

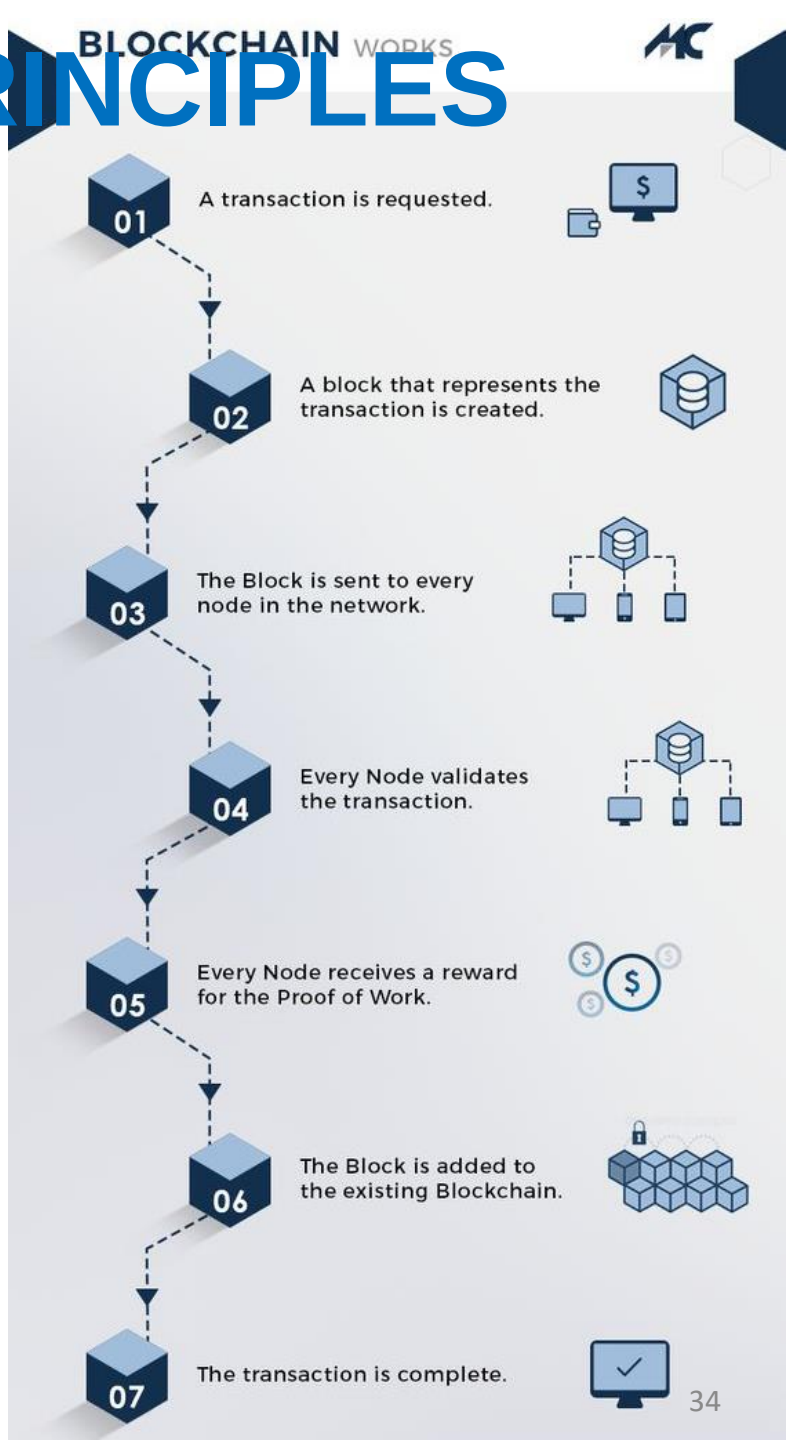
## The Blockchain Flow

**1 Party initiates transaction**
Requests verification on a blockchain with identity and transaction details

**2 Transaction sent to P2P network**
Uploaded and visible to a peer-to-peer network of individual computers (nodes)

**3 Validation by nodes**
Individual nodes work to validate the transactions legitimacy via algorithm

**4 Transaction approved**
Once approved, transactions (blocks) are added to the distributed public ledger

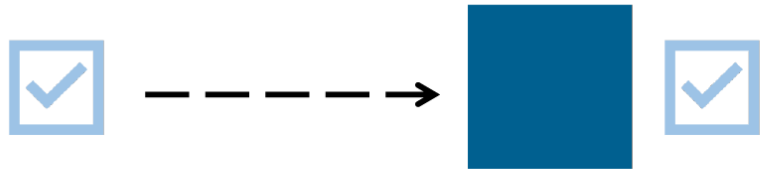


# 5. BLOCKCHAIN WORKING PRINCIPLES

## How Blockchain Works

1. A transaction is requested. The transaction can involve contracts, records, or other information (including cryptocurrency).

2. The requested transaction is broadcast to a network of nodes, comprised of computers or servers.

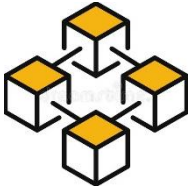


3. The network of nodes validates the transaction and the user's status using algorithms that the network has agreed upon.

4. Once verified, the transaction is combined with other transactions to create a new block of data for the ledger.

5. The new block is added to the existing blockchain. It is distributed through the network, with the other blocks ensuring that the information is not altered.

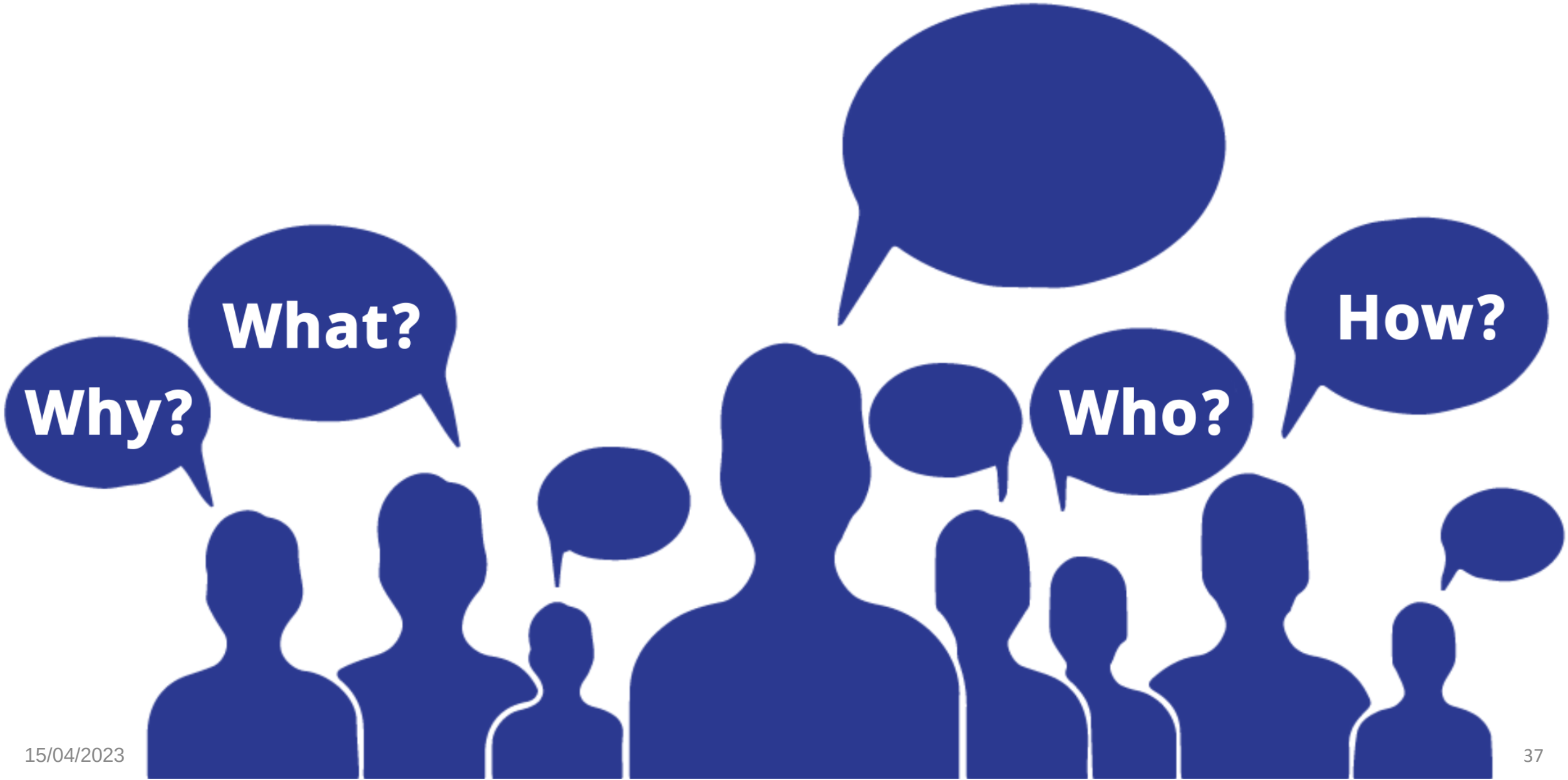
6. The transaction is completed.



# 6. SUMMARY

- Network architecture: Decentralized (hybrid P2P + Client Server)
- Nodes: devices running protocols
- Protocols: Reach the network concensus
- Concensus mechanism: PoW, PoS, PoA, …
- Working principle: create & broadcast, verify, enforce  transactions

# 7. DISCUSSION



# FINISH