Blockchain Algorithms summary

D. Papaioannou, P. Christodoulou, Prof. Ioannis Pitas
Aristotle University of Thessaloniki
pitas@csd.auth.gr
www.aiia.csd.auth.gr
Version 4.0

Artificial Intelligence & Information Analysis Lab

Blockchain Algorithms

- **Introduction to Cryptography**
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Introduction to Cryptography

- ***Four pillars of Cryptography:***
  - ***Encryption:*** Method for converting normal texts (plaintext) into a sequence of random bits (ciphertext).
  - ***Decryption:*** Defined as the inverse task of encryption, transformation of ciphertext to plaintext.
  - ***Cipher:*** An algorithm for modifying plaintext to chiphertext or the inverse.
  - ***Key:*** A set of information which is fed as input in the encryption/ decryption operation in order to produce the desired output, the ciphertext or the plaintext respectively.

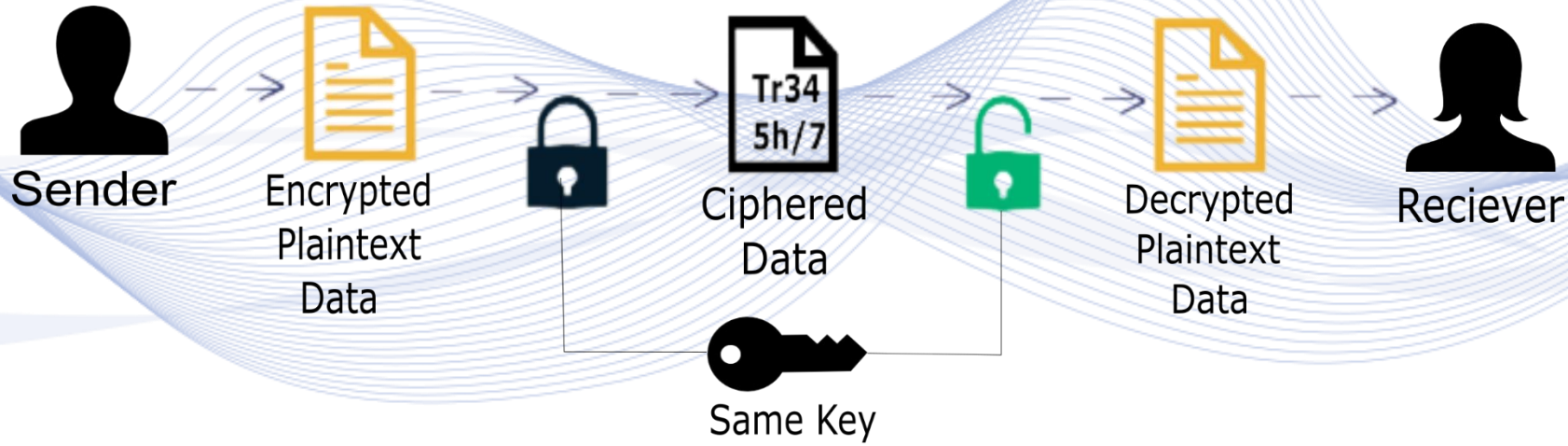
Introduction to Cryptography

- ***Types of Cryptography:***
  - ***Symmetric Key Cryptography***: Achieving encryption and decryption using a single-key, also known as Private key Cryptography.
  - ***Asymmetric Cryptography:*** Achieving encrypt and decrypt by using a public key for the first operation and a private key for the second operation, also called Public key Cryptography.
  - ***Hash Functions:*** Irreversibly "encrypt" information achieved by using mathematical transformations.

Introduction to Cryptography

Symmetric Key Cryptography

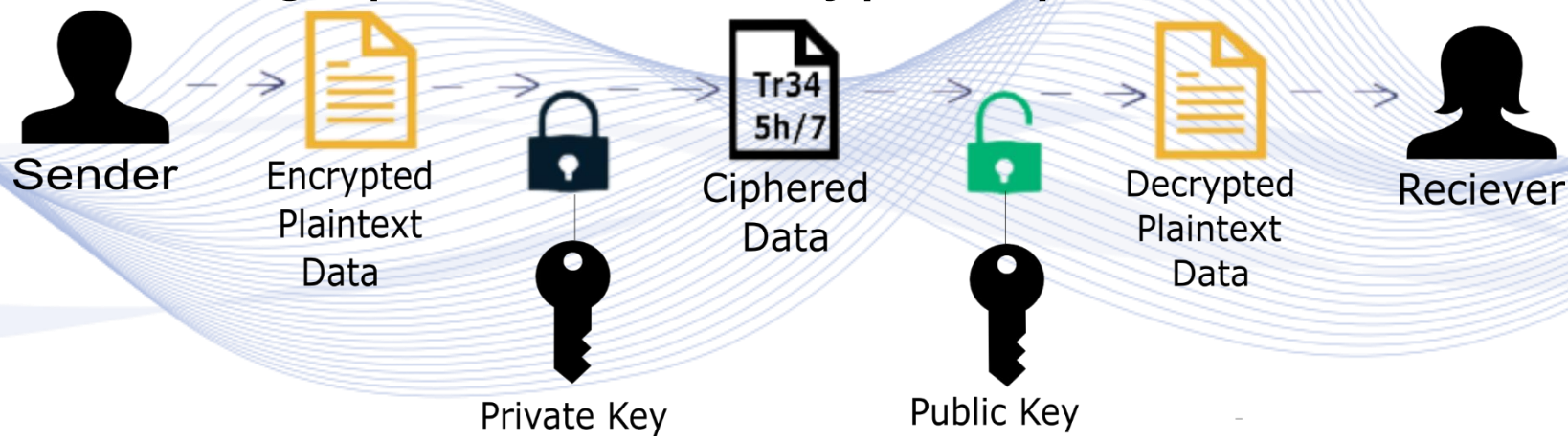
- ***Encryption*** and ***decryption*** achieved by ***one single key.***
- Both, sender and receiver, have knowledge about the ***shared key***.
- Fast Symmetric encryption achieved.
- Less secure for sensitive data since key size is smaller.



Introduction to Cryptography

Asymmetric Key Cryptography

- ***Public key*** is shared by ***everyone*** on the cryptographic scheme while ***Private key*** is known only by the ***authenticated user***.
- Private key is a result by a randomly generated number and the public key is the result of the irreversible algorithm
- Less processing speed and encryption power.



Introduction to Cryptography

- ***Hash Function:*** Can be defined as a " ***digital fingerprint"*** (unique identifier) for every given piece of data.

- A process that receives as ***input*** a plaintext data $\boldsymbol{X}$ of any size and maps it into a ***unique output*** (chipertext) of a fixed size.

- The ***Secure Hash Algorithm SHA***, developed by NIST and NSA, is one of the most efficient and well-known hash function families.

Introduction to Cryptography

- ***Four pillars of Cryptography:***
  - ***Encryption:*** Method for converting normal texts (plaintext) into a sequence of random bits (ciphertext).
  - ***Decryption:*** Defined as the inverse task of encryption, transformation of ciphertext to plaintext.
  - ***Cipher:*** An algorithm for modifying plaintext to chiphertext or the inverse.
  - ***Key:*** A set of information which is fed as input in the encryption/ decryption operation in order to produce the desired output, the ciphertext or the plaintext respectively.

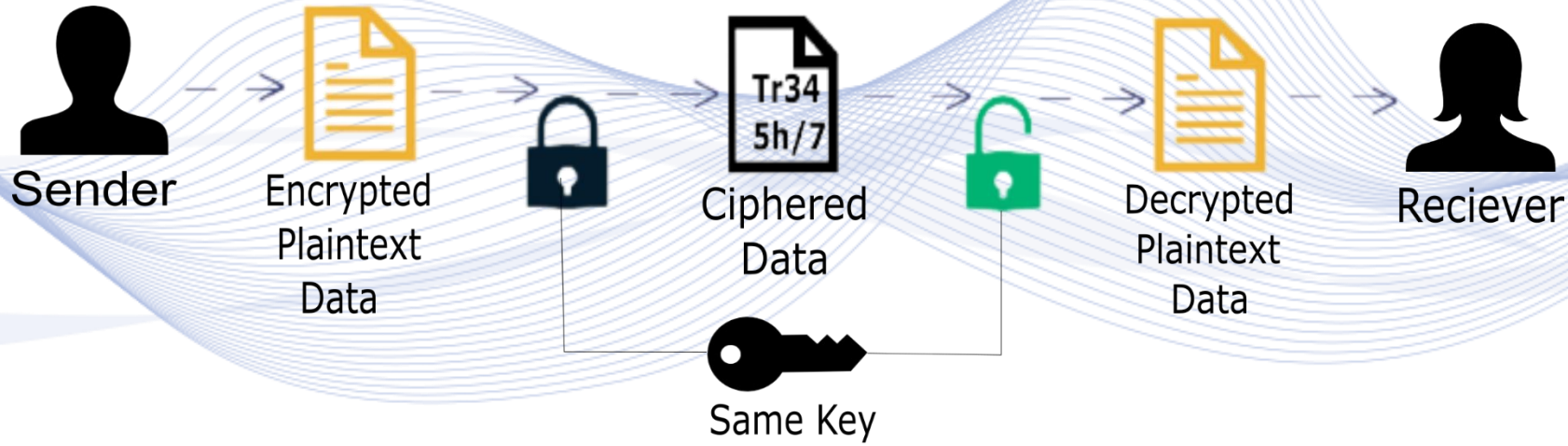
Introduction to Cryptography

- ***Types of Cryptography:***
  - ***Symmetric Key Cryptography***: Achieving encryption and decryption using a single-key, also known as Private key Cryptography.
  - ***Asymmetric Cryptography:*** Achieving encrypt and decrypt by using a public key for the first operation and a private key for the second operation, also called Public key Cryptography.
  - ***Hash Functions:*** Irreversibly "encrypt" information achieved by using mathematical transformations.

Introduction to Cryptography

Symmetric Key Cryptography

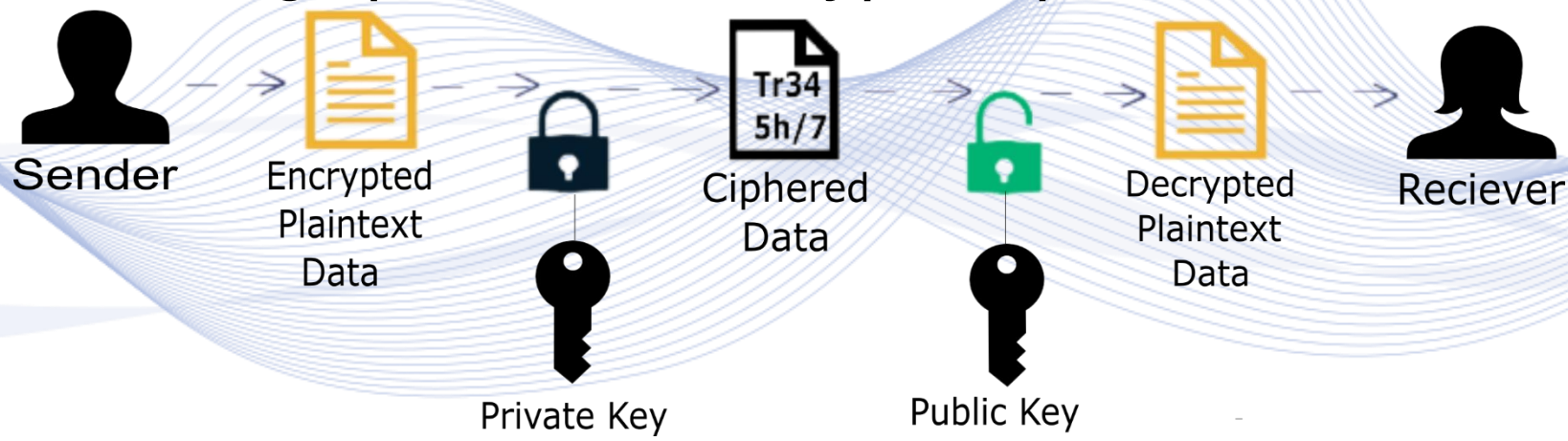
- ***Encryption*** and ***decryption*** achieved by ***one single key.***
- Both, sender and receiver, have knowledge about the ***shared key***.
- Fast Symmetric encryption achieved.
- Less secure for sensitive data since key size is smaller.



Introduction to Cryptography

Asymmetric Key Cryptography

- ***Public key*** is shared by ***everyone*** on the cryptographic scheme while ***Private key*** is known only by the ***authenticated user***.
- Private key is a result by a randomly generated number and the public key is the result of the irreversible algorithm
- Less processing speed and encryption power.



Introduction to Cryptography

- ***Hash Function:*** Can be defined as a "***digital fingerprint"*** (unique identifier) for every given piece of data.

- A process that receives as ***input*** a plaintext data $\boldsymbol{X}$ of any size and maps it into a ***unique output*** (chipertext) of a fixed size.

- The ***Secure Hash Algorithm SHA***, developed by NIST and NSA, is one of the most efficient and well-known hash function families.

Blockchain Algorithms

- Introduction to Cryptography
- **Blockchain**
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Blockchain

- Blockchain essentially consists of a ***sequence of blocks*** each one of them holding a complete list of transactions records.

- Can be defined as a database system which holds data sets secured and bounded to each other in a ***chain***, using crypto-graphic principles, in form of packages (blocks) where each one of the blocks consist of various transactions (TX-n).

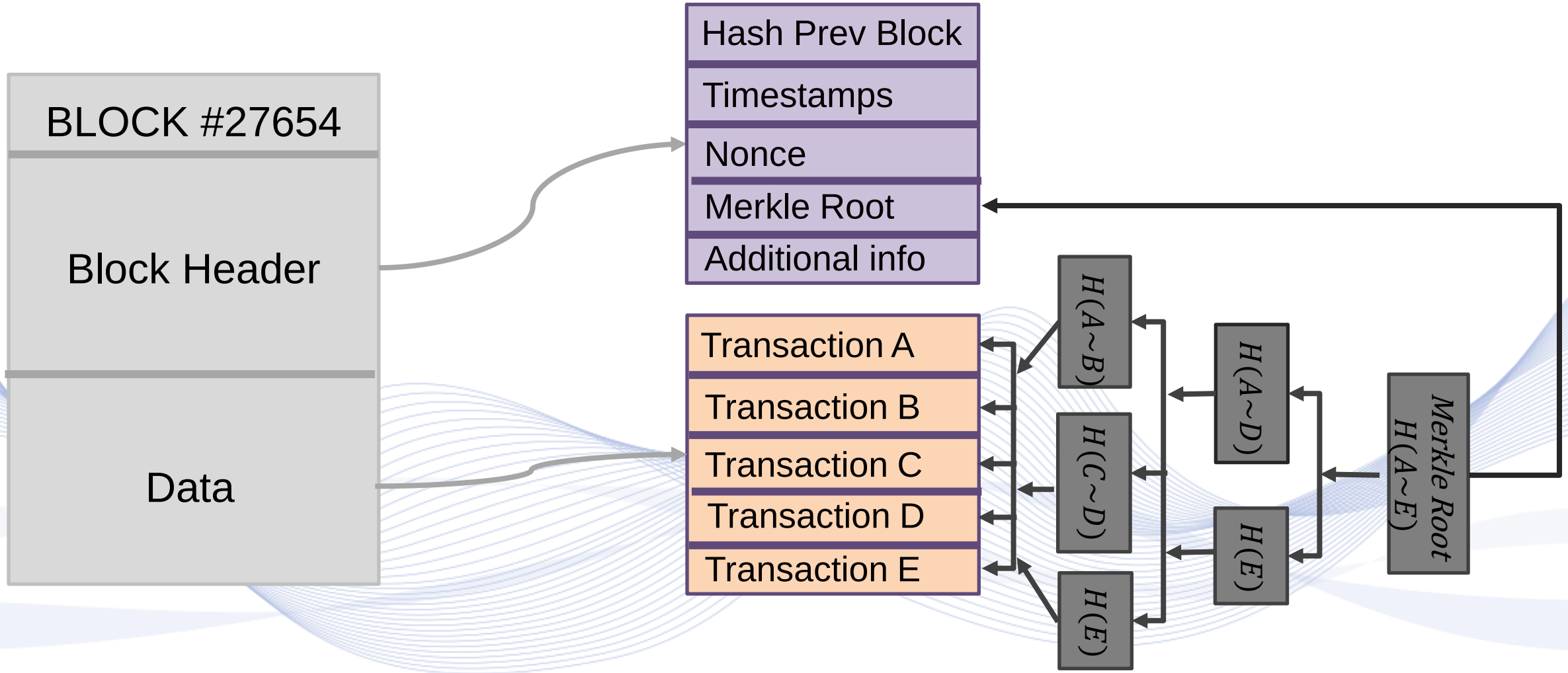
Blockchain

- Each block contains the header where ***timestamp, hash for the past block, hash for the block itself*** and a ***nonce*** are stored inside. The above structure can ensure the integrity of the chain from the first to the last block.

- First block is known as "***Genesis block***" and does not contain hash value for the previous block.

- ***Nonce*** is a $32 - bit$ random integer number used in order to validate the hash value produced for the specific block.

Blockchain


BLOCK #27654
Block Header
Data
Hash Prev Block
Timestamps
Nonce
Merkle Root
Additional info
Transaction A
Transaction B
Transaction C
Transaction D
Transaction E
H(A~B)
H(C~D)
H(E)
H(A~D)
H(E)
Merkle Root
H(A~E)

Blockchain

Blockchain technology is a ***peer-to-peer***, distributed network ledger which is secured by cryptography methods, immutable, append-only and updated only via consensus or agreement between nodes.

- ***Peer-to-peer:*** No central authority governs this network, but instead all participants (peers) must communicate directly with each other. In case of transaction cash, this feature allows exchanges to be achieved directly between peers without third-party interaction.

Blockchain

- ***Distributed Ledger:*** Ledger is spread all over the network and between all the nodes (peers), and each node keeps a copy of the entire ledger.

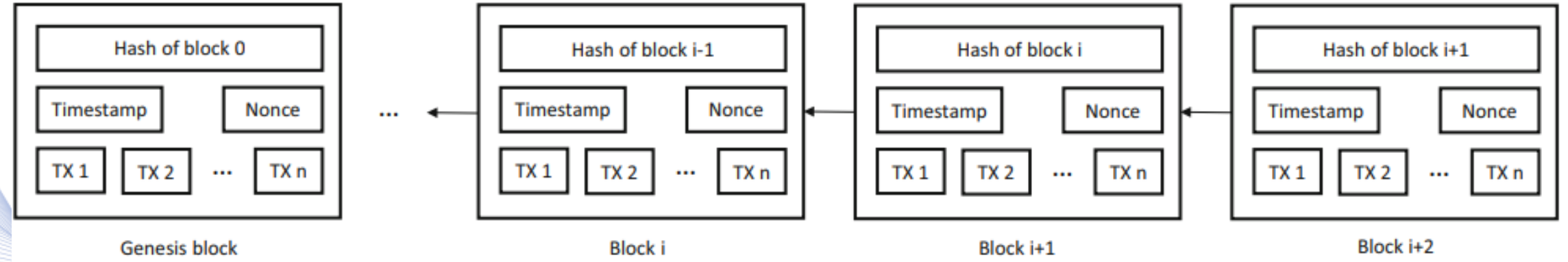
- ***Cryptographically-secure:*** Cryptography assures the security, making in that way the ledger safe in terms of misuse and tampering. Data origin authentication and integrity is necessary for that step.

Blockchain

- ***Timestamps*** used in order to store the blocks in the chain in chronological order. They consist of information about when a certain event is occurred (date, time of day).

- The output of a hash function is **unique** and **secured**, since if a tiny change occur in the block, then the hash value will be changed significantly **(Avalanche effect)** making easy in this way to detect and prevent frauds.

Blockchain



The blocks structure.

Source: [NOF2017]

Blockchain

- New blocks in the chain can be inserted only if the majority of the users (nodes) came in some kind of ***agreement*** about the validity of the transaction which is recorded inside it, thus leading to the verification of the entire block.

- Once a **new block** is verified and inserted in the chain is permanently and can not be undone.

- The information inside the block can no longer be changed.

Blockchain

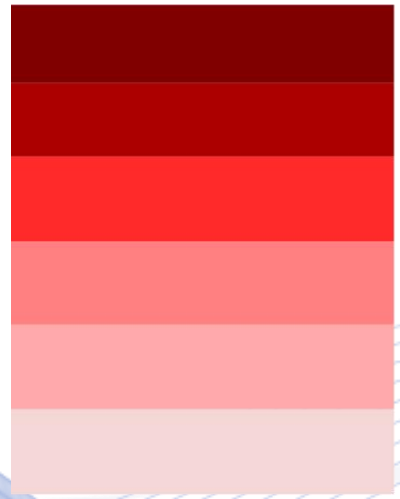
- Each blockchain node has an independent copy of the ledger, achieving in this way a complete ledger of transactions history.

- ***Cryptography*** and ***consensus*** is fundamental aspects of the blockchain since assures the consistency and the integrity of the entire chain.

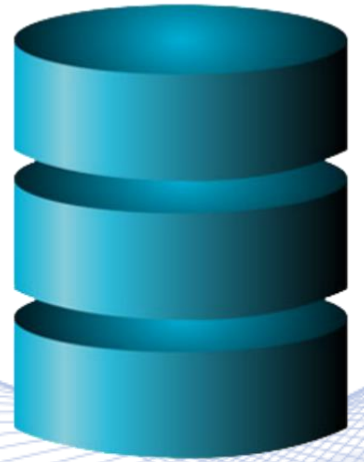
- The above architecture is actually a ***distributed ledger system***, ensuring the persistence of the chain even some nodes (users) break down.

Blockchain

Timestamp



Database



Consensus

Blockchain

- Participants in the Blockchain ecosystem:
  - ***Users:*** People who just want to transact and use the blockchain, they do not need to have a full copy of the transaction history in their devices, they can use the online wallet applications in order to communicate with the blockchain.

  - ***Nodes:*** As nodes are known the devices (laptop, computers, servers) which is used by the participants of the network. Nodes usually validate the transactions and they are responsible to store, preserve and spread (broadcasting transactions) the chain. Blockchain exists entirely on nodes.

Blockchain

- Participants in the Blockchain ecosystem:
  - ***Miners:*** A miner is in charge of creating new blocks and preserving the integrity of the chain. Must always run a full node in order to have access in all the valid transactions in the network. A node however, is not necessarily simultaneously a miner.

  - One device can run a full node, but it will only be able to validate, store and update the transactions. In order to be able to mine the blocks it will need the additional software as well as the additional hardware power.

Decentralized Ledger

- In a ***decentralized ledger***, transactions are no longer controlled by a single entity instead there are multiple entities controlling the system, each of which usually manipulate the ledger.

- In those ledgers data of transactions are maintained by multiple parties through a consensus protocol (agreement) and each party keeps a full copy of the ledger.

- Transactions are no longer need a middleman (third-party) interaction.

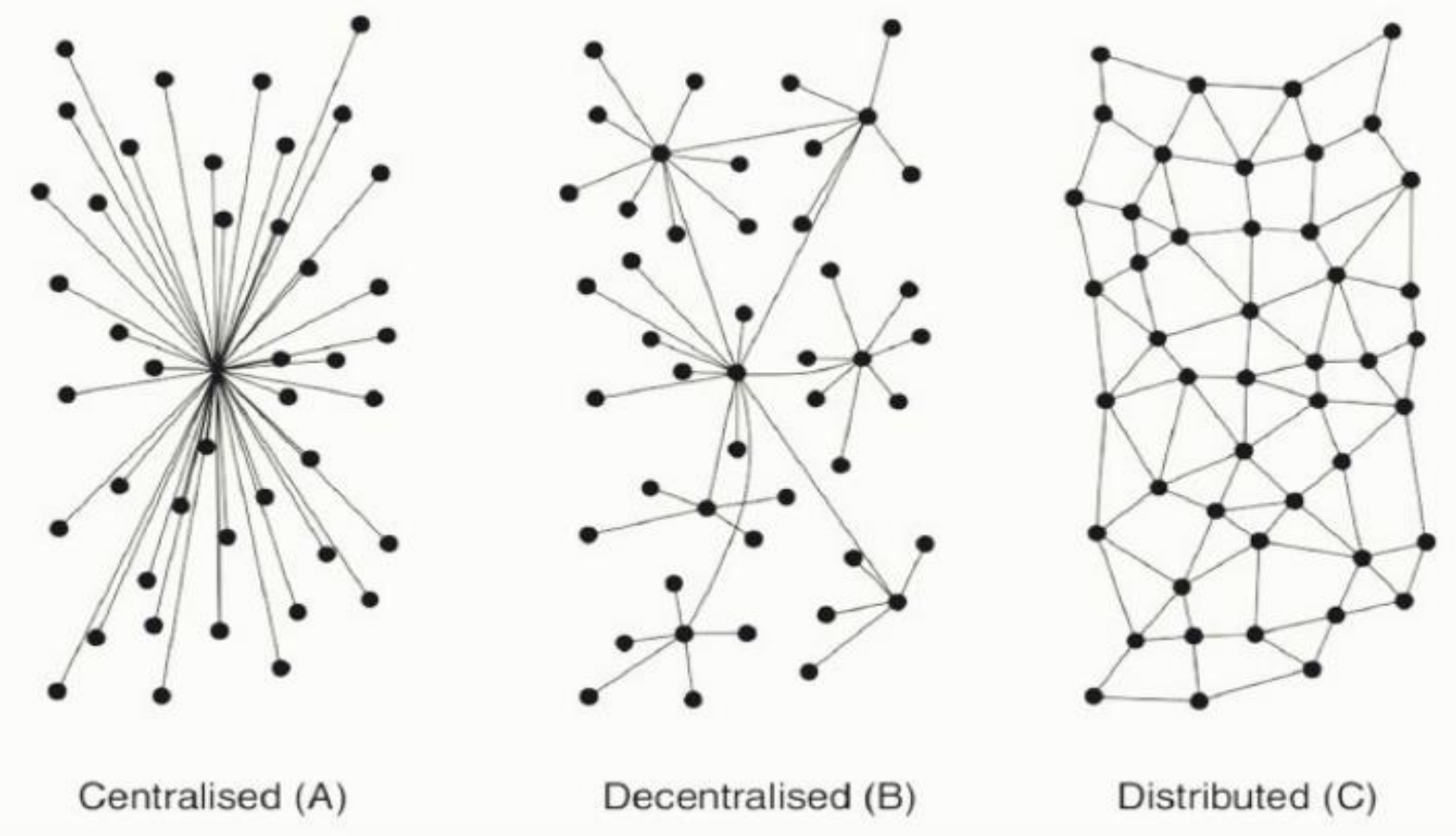
Distributed Ledger

- In a ***distributed ledger,*** all participants of the network have access to it, while the transactions are recorded between peers in a chronological order by using timestamps. Do not need a third party interaction or authorization.

- In a technical aspect ledgers/servers/databases are allocated in different locations instead in a central location.

- Everyone can own the full history of transactions (database) and have the rights to change the state of the ledger and to store it.

The Network



Source: Baran (1964)

Distributed Ledger Network

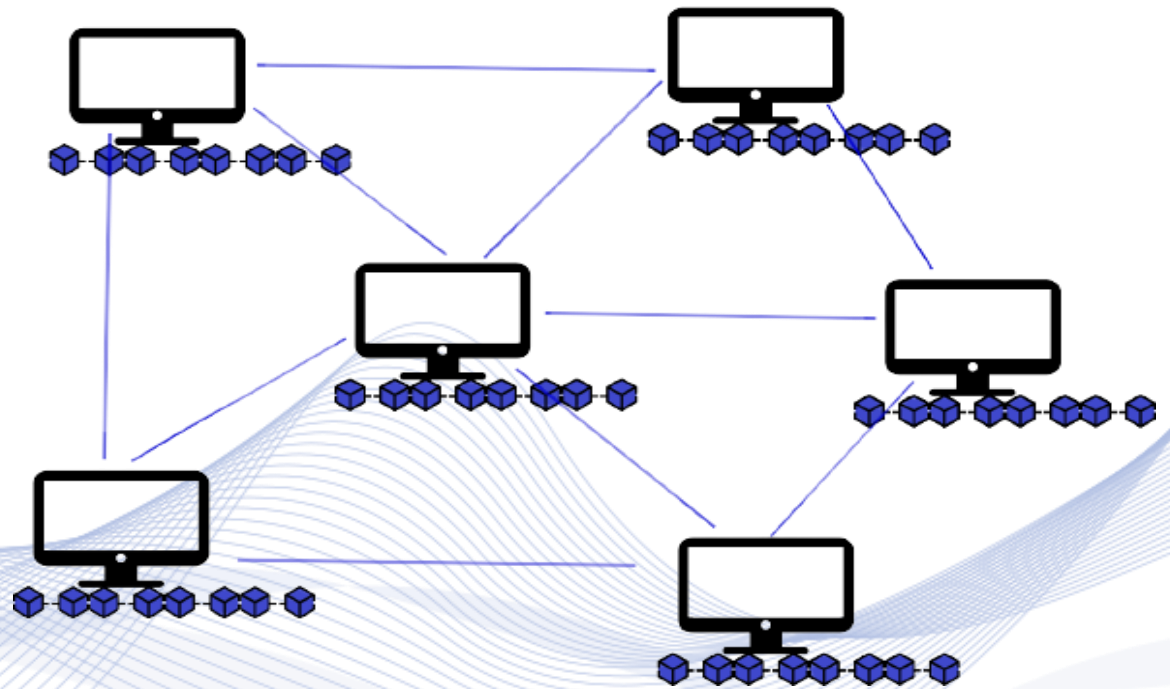
- Asymmetric cryptography ensures that every new piece of information to be written in the database can be uniquely linked to the sending participant (proof of origin) and cannot be changed or manipulated because it is encrypted.

- Transactions are verified only if a quorum achieved between the participants.

Distributed P2P Network

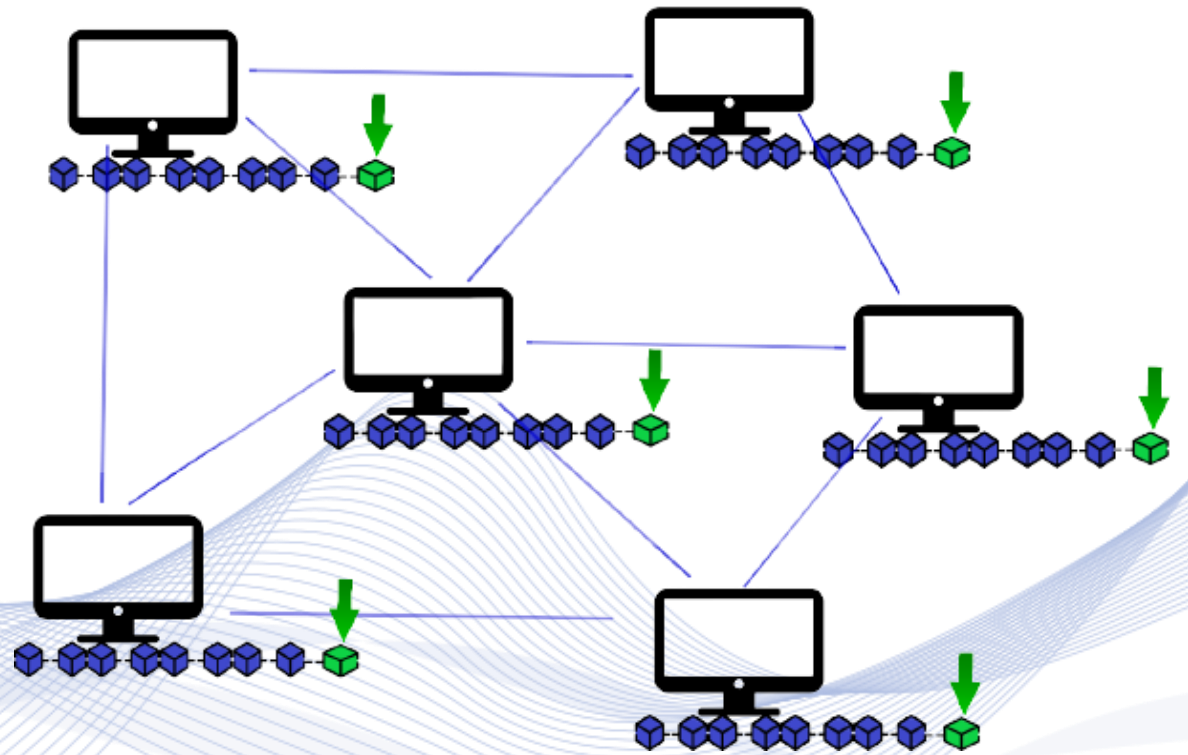
- In distributed P2P networks all computers are interconnected and communicate directly with each other.

- Blockchain is actually copied across all of those computers, instead of storing the data in one system or government server.



Distributed P2P Network

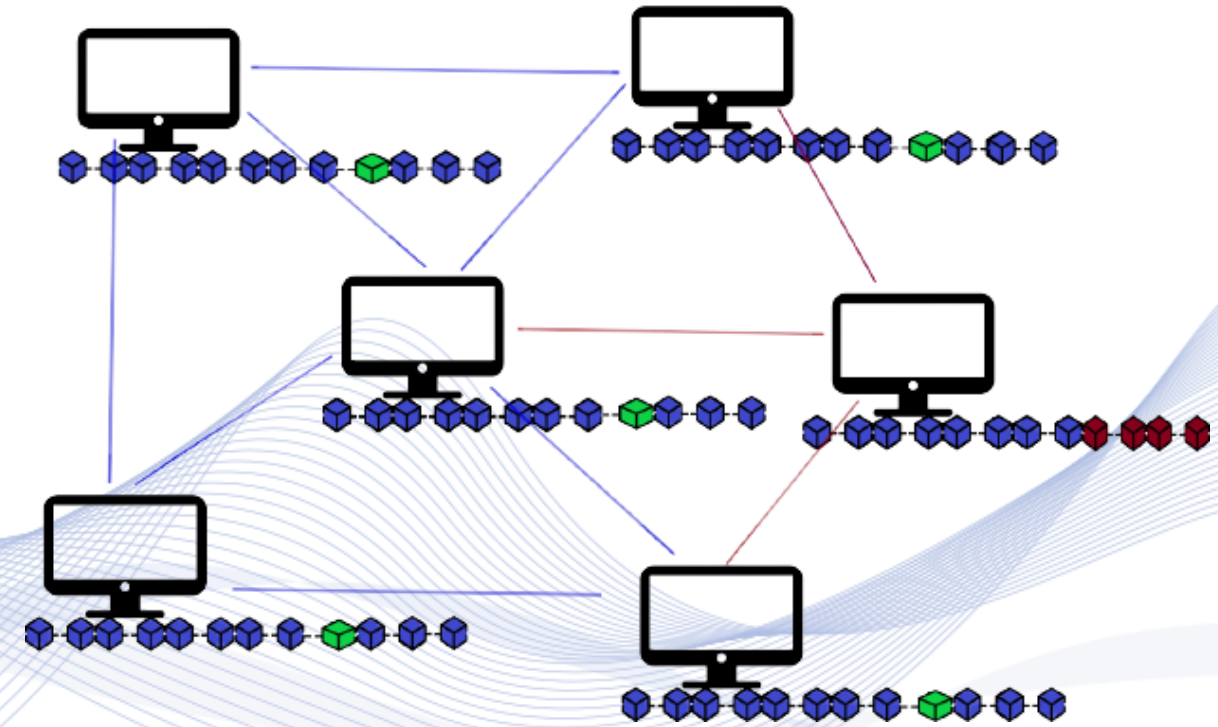
- Once a new block is added, that information is broadcasted across all the network until all the peers have access to that new block.
- Following this scheme more and more blocks added in the network.



Distributed P2P Network

- If a malicious attack occur in a single peer, then it can be detected because the network constantly checking if the blocks of the peers are matched up.

- As soon as the hacked peers is a minority then the correct blocks will be copied in the damaged peers and blockchain will be restored. So, a hacker must attack not in one computer in order to be successful, but in the majority of them.

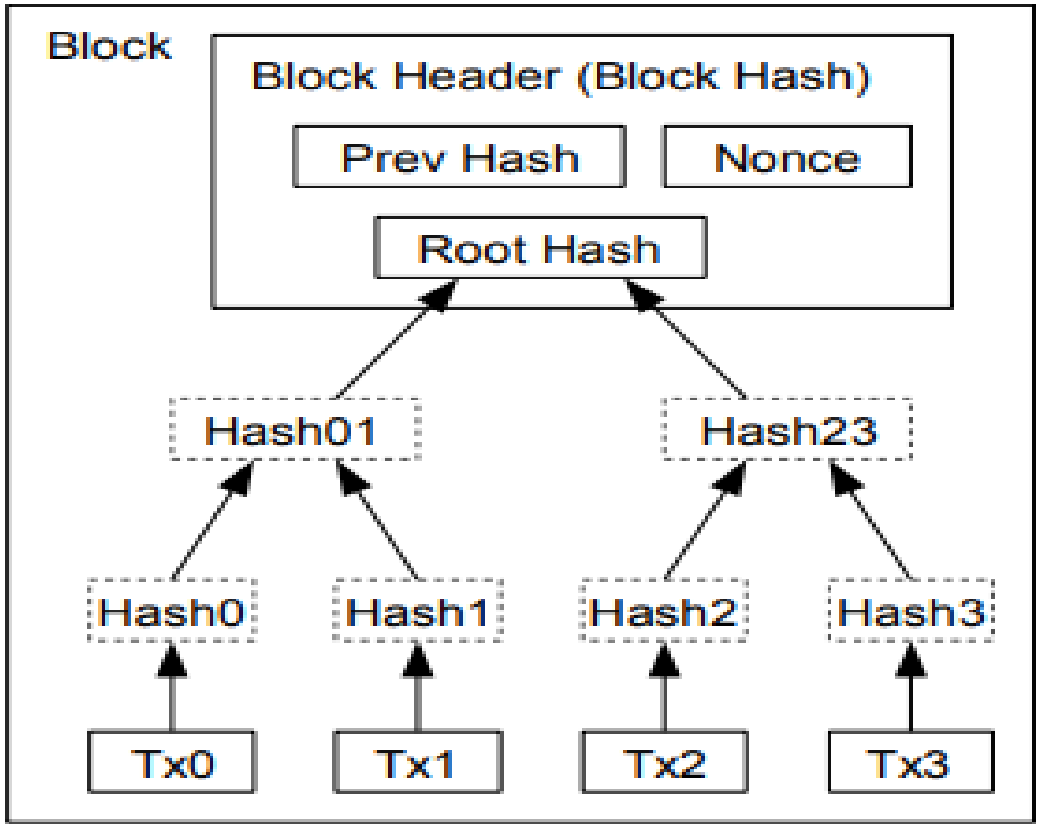


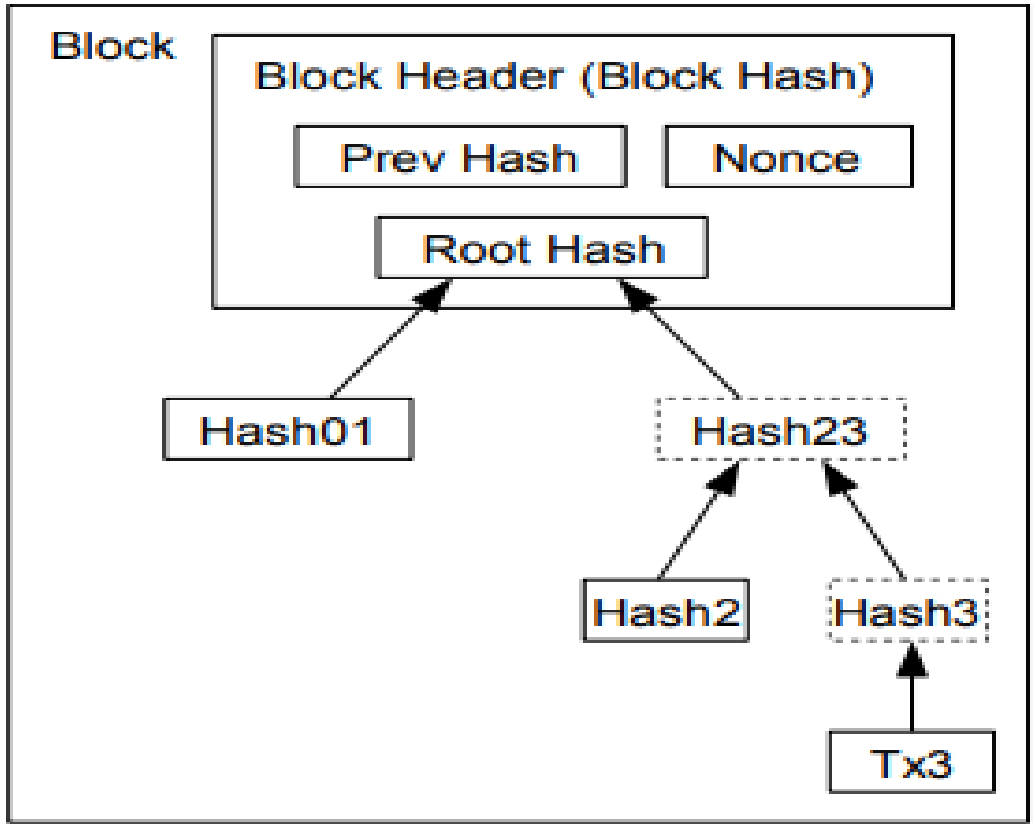
Storage Structure

- In each block we store transactions. In case of bitcoin, each block can store $1mb$ of transaction records and each new block is created every ten minutes.

- An average blockchain consist of hundred of thousand blocks where each block made up with thousands of transactions, as we see we speak for tons of data which they result memory and computing power issuses. That problem is sloved by the Merklee Binary Tree.

Storage Structure





Transaction Storage.

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- **The Byzantine Generals' problem**
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Byzantine Generals' Problem

Let's imagine that the Byzantine army gathered around an enemy city, troops are ruled by its own General, they observe and then they need to agree on a common action plan– **Attack** or **Retreat**.

- Generals can communicate with each other only through messengers.

- ***Synchronization*** between generals on a ***common plan*** must be achieved.

Byzantine Generals' Problem

Traitors may appear among the generals, aiming to prevent the ***honest generals*** from agreeing.

Generals need an algorithm which will ensure:

- **Condition A:** All honest generals will eventually agree on the common battle plan (agreement).

- **Condition B:** Honest generals cannot be forced by a small number of traitors to embrace a bad battle plan.

Byzantine Generals' Problem

The generals' decision is achieved by:

- Each one of the generals is firstly observing the enemies and then conveys his remarks to the others. Let's assume that $v(i)$ is the remarks communicated by the $i-th$ general.

- Some techniques are used by each general to concatenate the $v(1), \ldots, v(n)$ observations into a common battle plan.
  - n refers to the number of the generals.
  - The Generals must use the same technique to concatenate the observations into a single decision in order the Condition A to be achieved.

Byzantine Generals' Problem

The generals' decision is achieved by:

- If most of the votes are gathered around the general's opinion, then the final decision can be achieved.

- The final decision can be contaminated by the presence of traitors if and only if honest generals are almost equally divided between the two decisions (Attack or Retreat).

- In this scenario, neither decision can be considered bad.

Byzantine Generals' Problem

Communication among the generals:

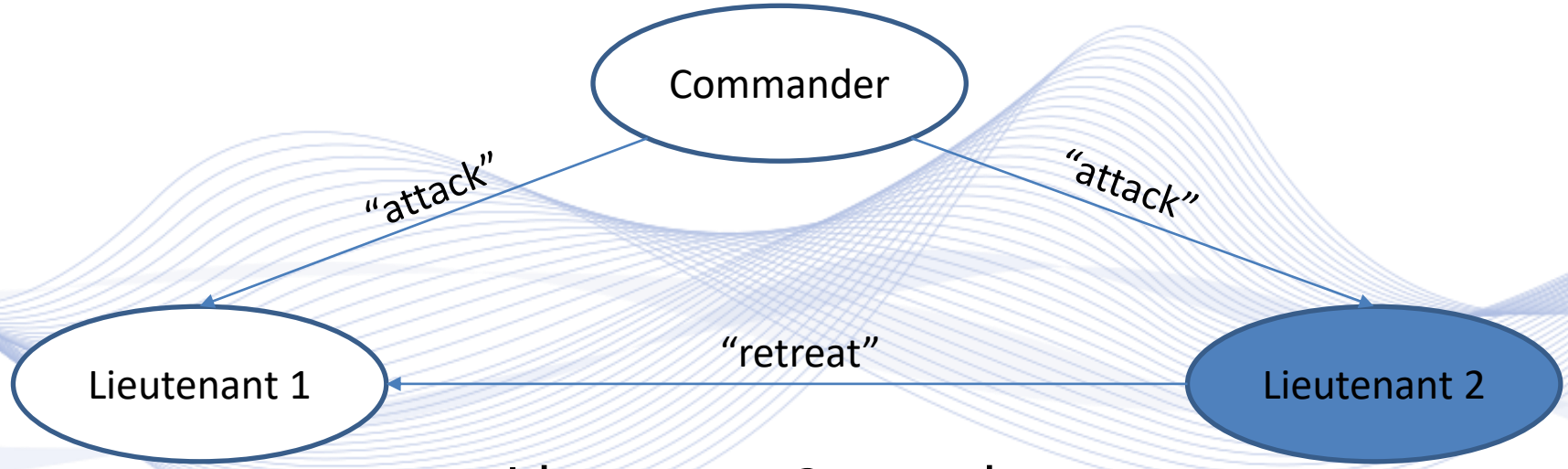
- The clearest technique is the $i-th$ general to send $v(i)$ information, through messengers to each other general.

- But to achieve term A, this strategy is ambiguous, as each general must receive the same values of messages $v(1), \dots, v(n)$ but if a single traitor exists, then he can send different values to different generals.

Oral Messages Solution

If generals are three then no solution can be applied:

- In a three-generals scenario, if a single traitor exists, then we cannot have a solution.
  - ***Scenario 1:*** IC2 is achieved if the lieutenant 1 obey the commander's order to attack.

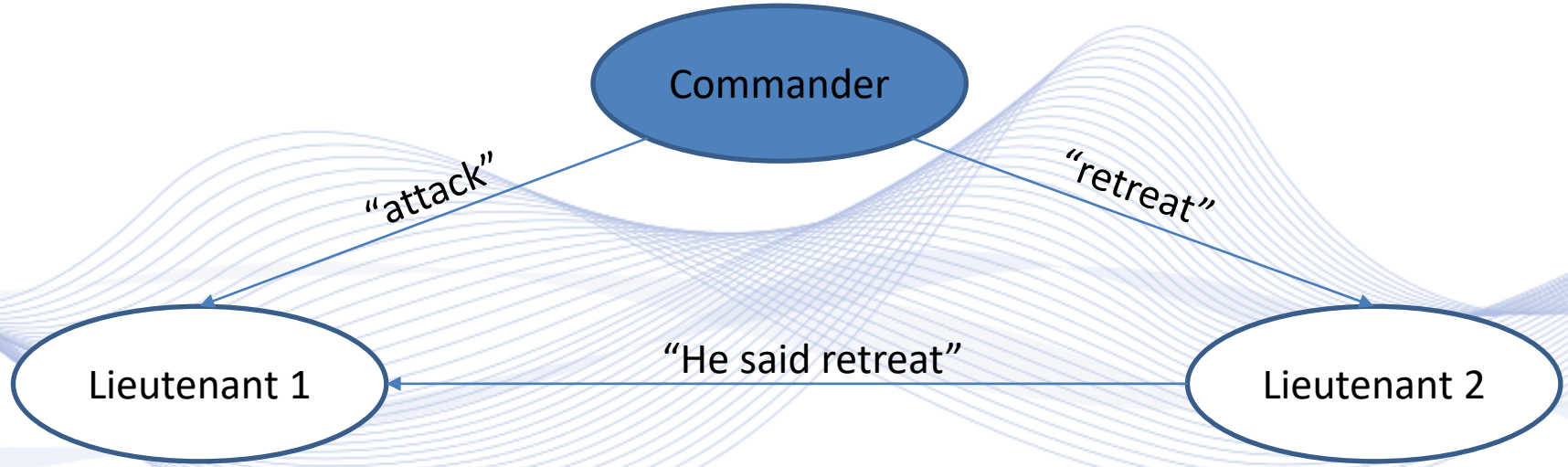


Lieutenant 2 as traitor.

Oral Messages Solution

If generals are three then no solution can be applied:

- ***Scenario 2:*** When the traitor is the commander, he can send ambiguous messages to all the lieutenants. In this scenario lieutenants can not detect malicious orders, so they must obey in both.



Commander as traitor.

Oral Messages Solution

- If m traitors exist then there is no solution, as we see before, for less than $3m + 1$ generals.

- ***Oral messages(OM)*** can provide a solution for more than $3m + 1$ generals.

- In OM, every general must solve an algorithm which sends messages to other generals, and we assume that every loyal general can successfully complete his algorithm.

Oral Messages Solution

- An ***oral message*** involves the follow properties:
  - ***A1:*** Each of the sent messages are hand over accurately
  - ***A2:*** The recipient knows who the message came from.
  - ***A3:*** The non-sending of a message can be recognized.

Oral Messages Solution

- If a traitor commander exist it may decide not to send any value to his lieutenants, and since lieutenants must obey an order, we must define a default one, let's say "RETREAT".

Algorithm $OM(0)$:

- The commander sends his value to every lieutenant.

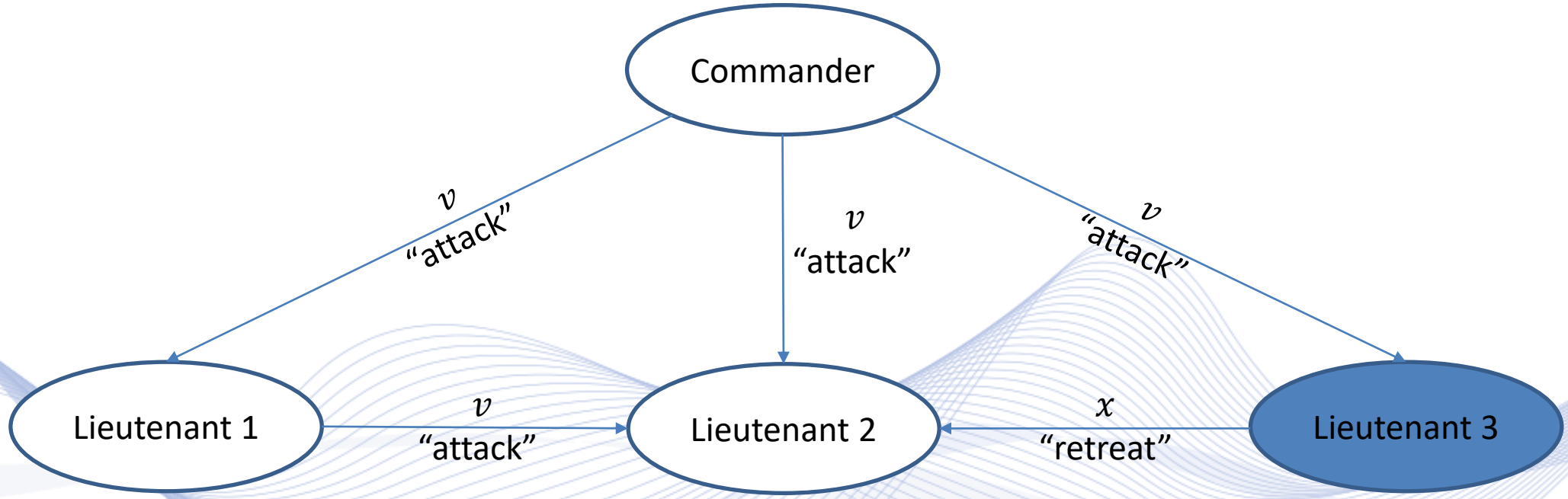
- Each lieutenant uses the commander's given value, or the default value "RETREAT" when the commander does not provide any order.

Oral Messages Solution

- This algorithm assumes a function named as "$majority$" with the property that if a majority of the values v_i equal v, then $majority(v_1, v_2, \dots, v_n) = v$.

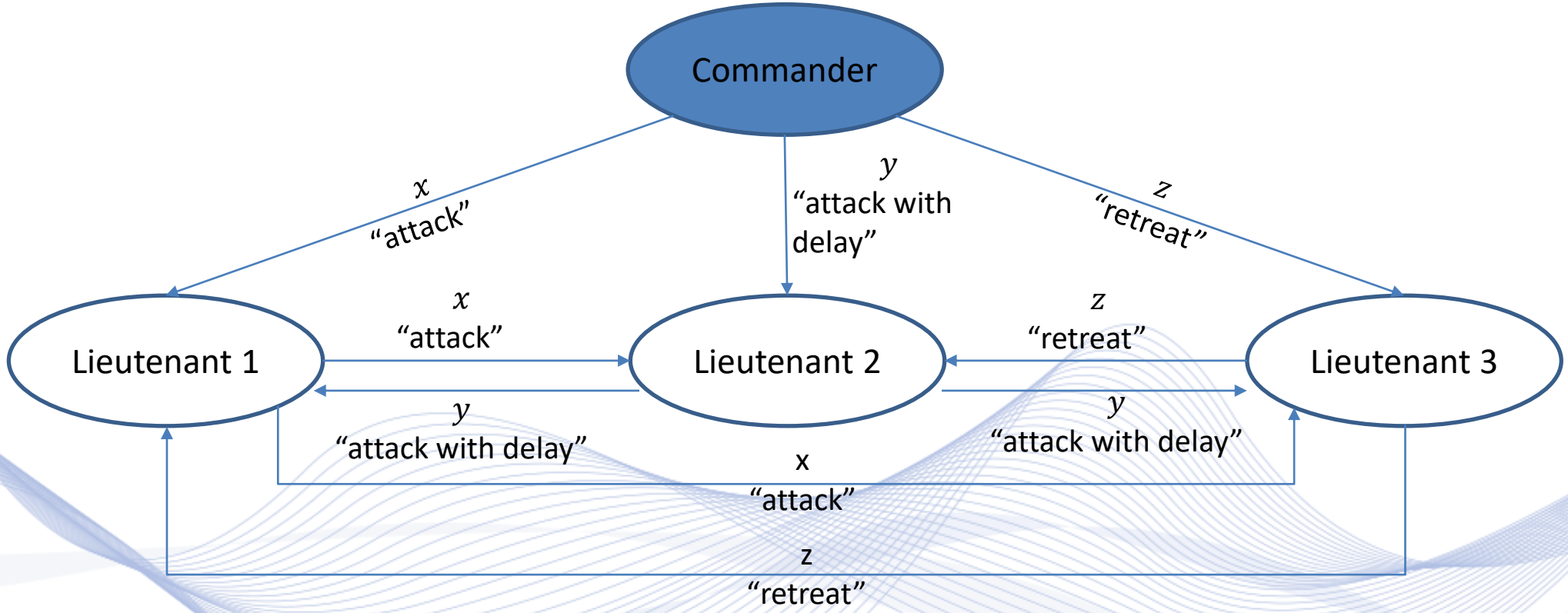
- Precisely it assumes a sequence of those functions one for each general n. There exist two natural choices for the value of $majority(v_1, v_2, \dots, v_n)$:
  - The majority value among the v_i if it exists, else the value RETREAT.
  - The median of the v_i, if they come from an ordered set.

Oral Messages Solution



Example of Algorithm $OM(1)$

Oral Messages Solution



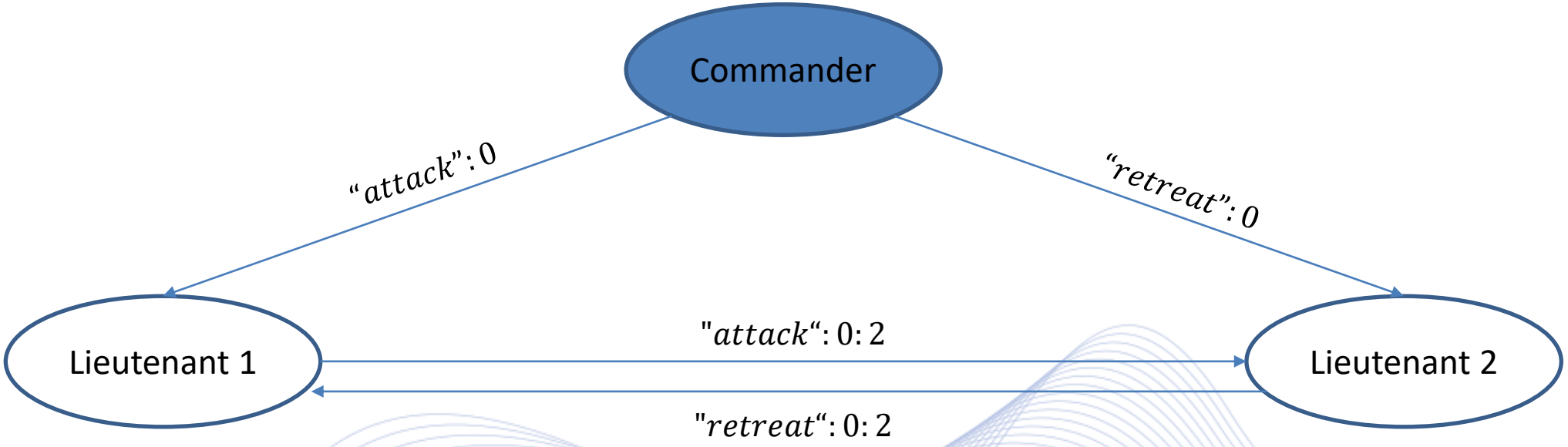
Commander as traitor.

Signed Messages Solution

- In this scheme, a commander sends a signed command to each of his lieutenants. Each lieutenant adds his own signature to that command and send it to the other lieutenants.

- A lieutenant must efficiently receive one signed message, copy it enough times, sign it and send those copies. In the end, each message may consist of a stack of identical messages which are signed and distributed as required.

Signed Messages Solution



Algorithm SM- Traitor Commander.

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- **Distributed System Consensus**
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Distributed Consensus

Consensus in distributed systems represent a state that all participants agree on the same data values. Depending on the medium for message transmission, distributed systems are classified as follows:

- **Message Passing Systems:** Distributed consensus on a single network history is achieved through peer-to-peer communication.

- **Shared Memory Systems:** Consist of multiple independent processing nodes with local memory modules which are connected by a general interconnection network.

Distributed Consensus

- Let a distributed system consisting of N independent processes. Each process p_i begins with an individual initial value x_i and communicates with the others to update this value which can be used for a certain task.

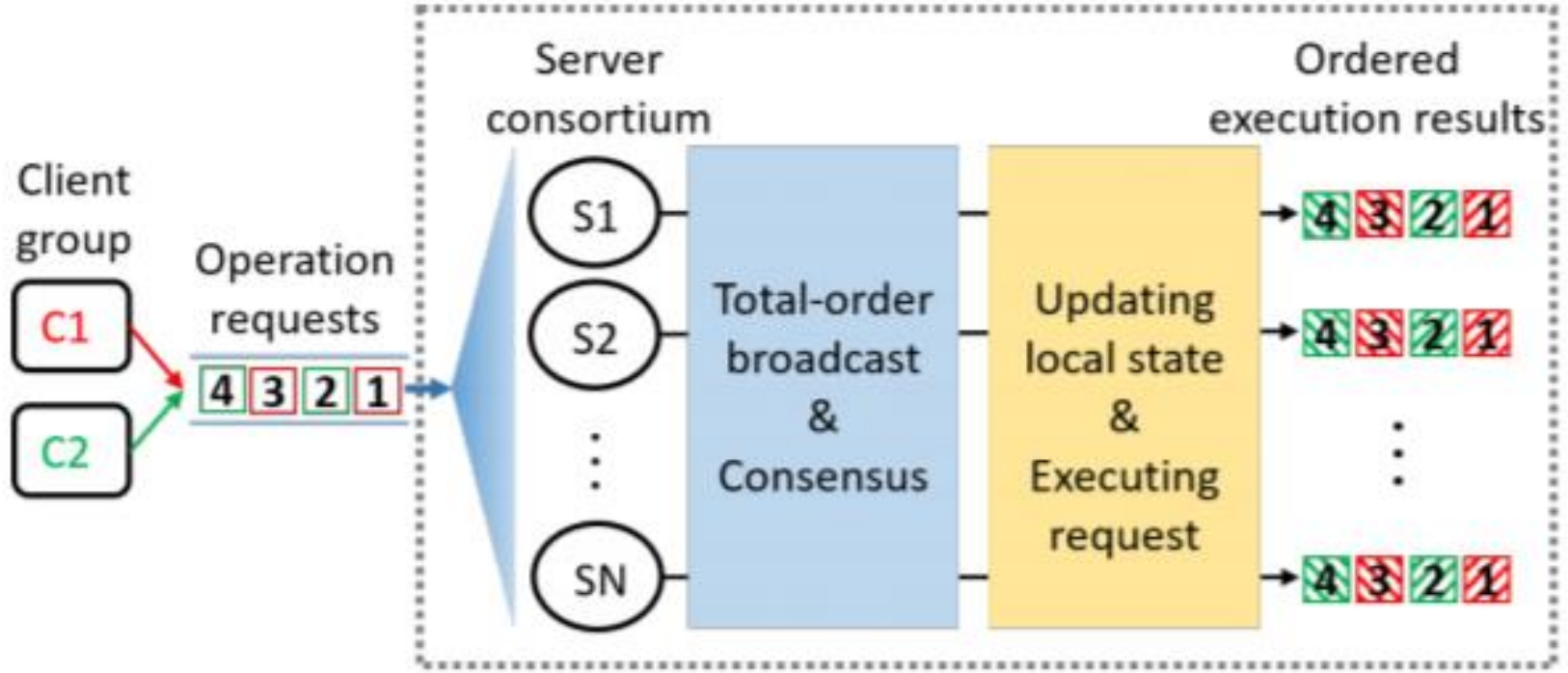
- If the processes are required to perform the same task, consensus on a single value is required before they proceed to the task.

Consensus Correctness

- In a distributed computing framework, one or more clients issue operations requests to the server consortium, which provide timely and correct computing service in response to the request despite of some server failures.

- Correctness of Consensus need the follow requirements:
  - ***Safety:*** Every server correctly executes the same requests.
  - ***Liveness:*** All requests should be served.

Consensus Correctness



Source: [XIA2020]

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - **Byzantine Fault Tolerance Consensus**
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Byzantine Fault Tolerance Consensus

A consensus protocol is named as ***Byzantine Fault Tolerance (BFT)*** if it can tolerate one or more crash or byzantine failures while keeping normal functioning.

BFT is defined with the follow requirements:

- ***Termination:*** Every nonfaulty process decides an output.

- ***Agreement:*** Every nonfaulty process eventually decides the same output $\hat{y}$.

Byzantine Fault Tolerance Consensus

BFT is defined with the follow requirements:

- ***Validity:*** If every process begins with the same input $\hat{x}$, then $\hat{x} = \hat{y}$.

- ***Integrity:*** Every nonfaulty process' decision and the consensus value $\hat{y}$ must have been proposed by some nonfaulty process.

Byzantine Fault Tolerance Consensus

The distributed network should satisfy the follow condition in order a BFT consensus protocol to operate:

$$N \geq 2f + 1,$$

where f is the number of Byzantine processes (faulty) and N the total number.

Byzantine Fault Tolerance Consensus

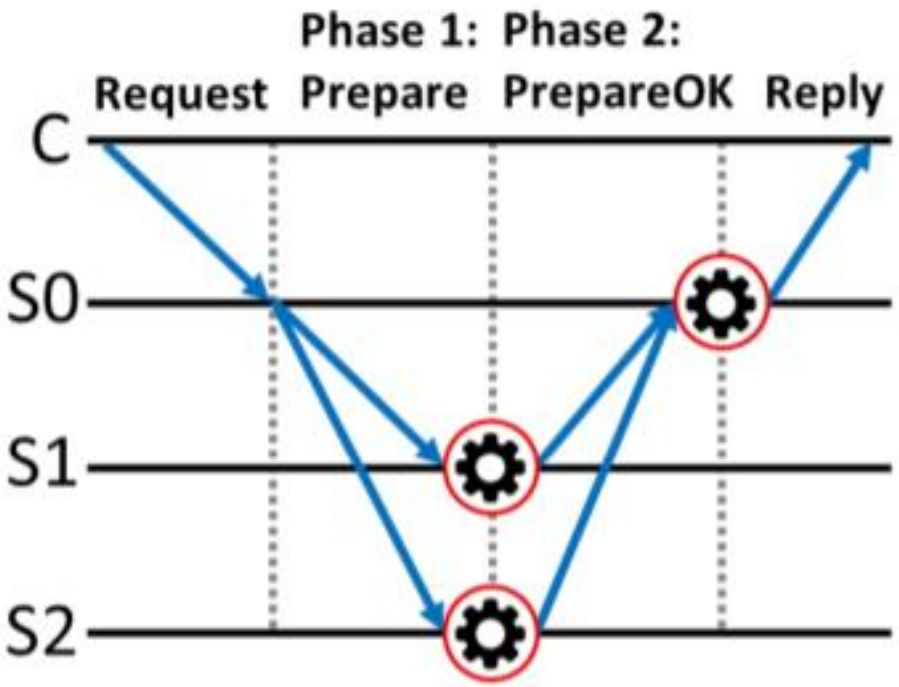
BFT-based Consensus Protocol idea:

- In most BFT algorithms we have a client-server scheme, where a client send requests to the server.

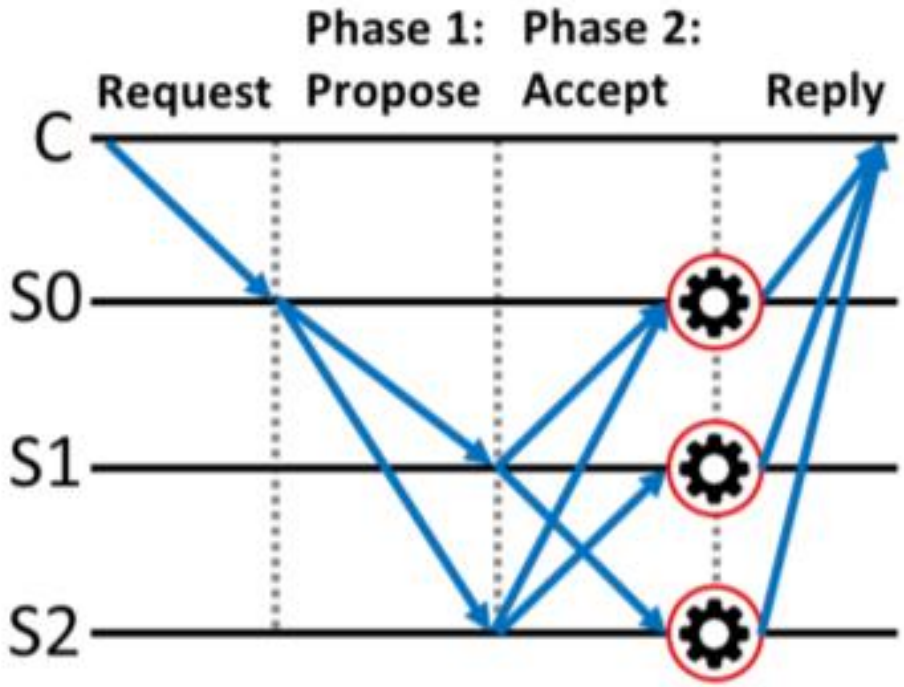
- The N-server consortium accepts client requests and confirms each other's state before reaching consensus and execute requests.

- The BFT SMR protocols are selecting a server as primary via a leader-electing process. This process is assumed to be randomly in most cases.

Byzantine Fault Tolerance Consensus



VR



Paxos

Source:[XIA2020]

Blockchain Algorithms

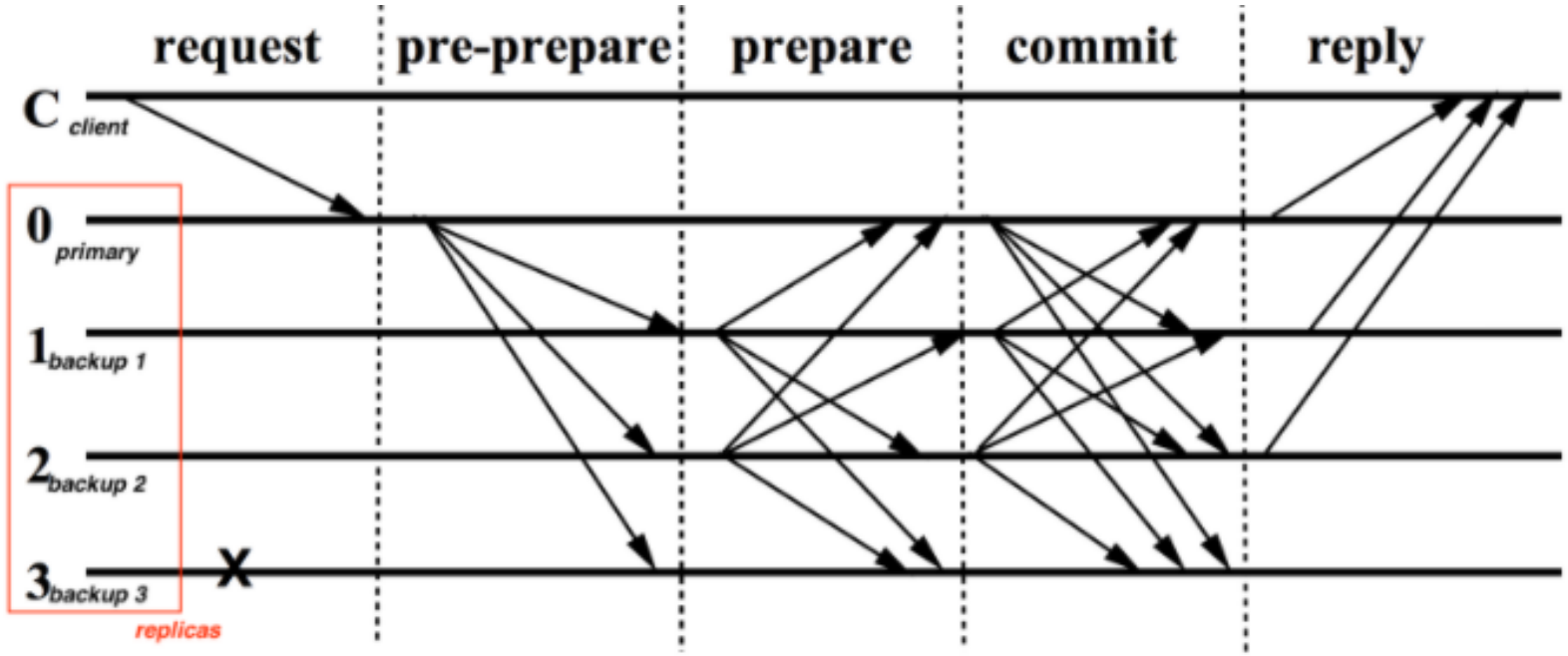
- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - **Practical BFT**
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Practical BFT

- As we see, **BFT protocols** and general the **BGP** allegory is strongly rely on **synchronous networks**, ***Practical Byzantine Fault Tolerance (pBFT)*** was the first protocol that tolerated Byzantine Faults under an ***asynchronous network***.

- The algorithm can be used to implement any deterministic replicated ***service*** with a ***state*** and some ***operations***.

Practical BFT



Algorithm implementation.

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - **Blockchain Consensus**
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Blockchain Consensus

- In a blockchain network ***every participant*** can be both a ***client*** (to issue transactions) and a ***server*** (to validate and finalize transactions).

- The foundational infrastructure of blockchain is a ***peer-to-peer*** overlay network on top of the internet.

- Every peer (or node) in the network operates autonomously with respect to the set of rules that covering peer protocol, consensus protocol, transaction processing, ledger management.

Blockchain Consensus

- Blockchain Network is considered as ***semi-synchronous*** where:
  - ***Weak synchronous:*** a message delivery is considered guaranteed, while the message delay vary but most likely will not grow longer as time elapse or,
  - ***Δ-Synchronous:*** remains within a certain bound.
- Those assumptions allow the consensus protocol to take advantage of the timing services of the Internet (e.g., timestamps).

Blockchain Consensus

- In blockchain networks, the message transmission suffers from network delays which gives rise to asynchronous consensus protocols.

- The ***goal*** of a blockchain consensus Protocol is to ensure that all participating ***nodes agree on a common network transaction history*** which is serialized in the form of blockchain.

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - **Nakamoto Consensus**
  - Proof of Work Consensus
  - Proof of Stake Consensus
- The 51% attack

Nakamoto Consensus Protocol

- The block or transaction messages are propagated across the P2P network through gossiping.

- Nakamoto consensus specifies the termination requirement into probabilistic finality:
  - ***Probabilistic finality:*** For any honest node, every new block is either discarded or accepted into its local blockchain. An ***accepted block*** may still be discarded but with an ***exponentially diminishing probability*** as the blockchain continues to grow.

Nakamoto Consensus Protocol

- Components of Nakamoto Consensus Protocol:
  - ***Proof of Work (PoW):*** Block generation requires finding a preimage to the hash function, so the hash result satisfies a difficulty target, which is dynamically adjusted to maintain an average block generation time.
  - ***Gossiping rule:*** Any newly received or locally generated transaction or block should be immediately advertised and broadcast to peers.

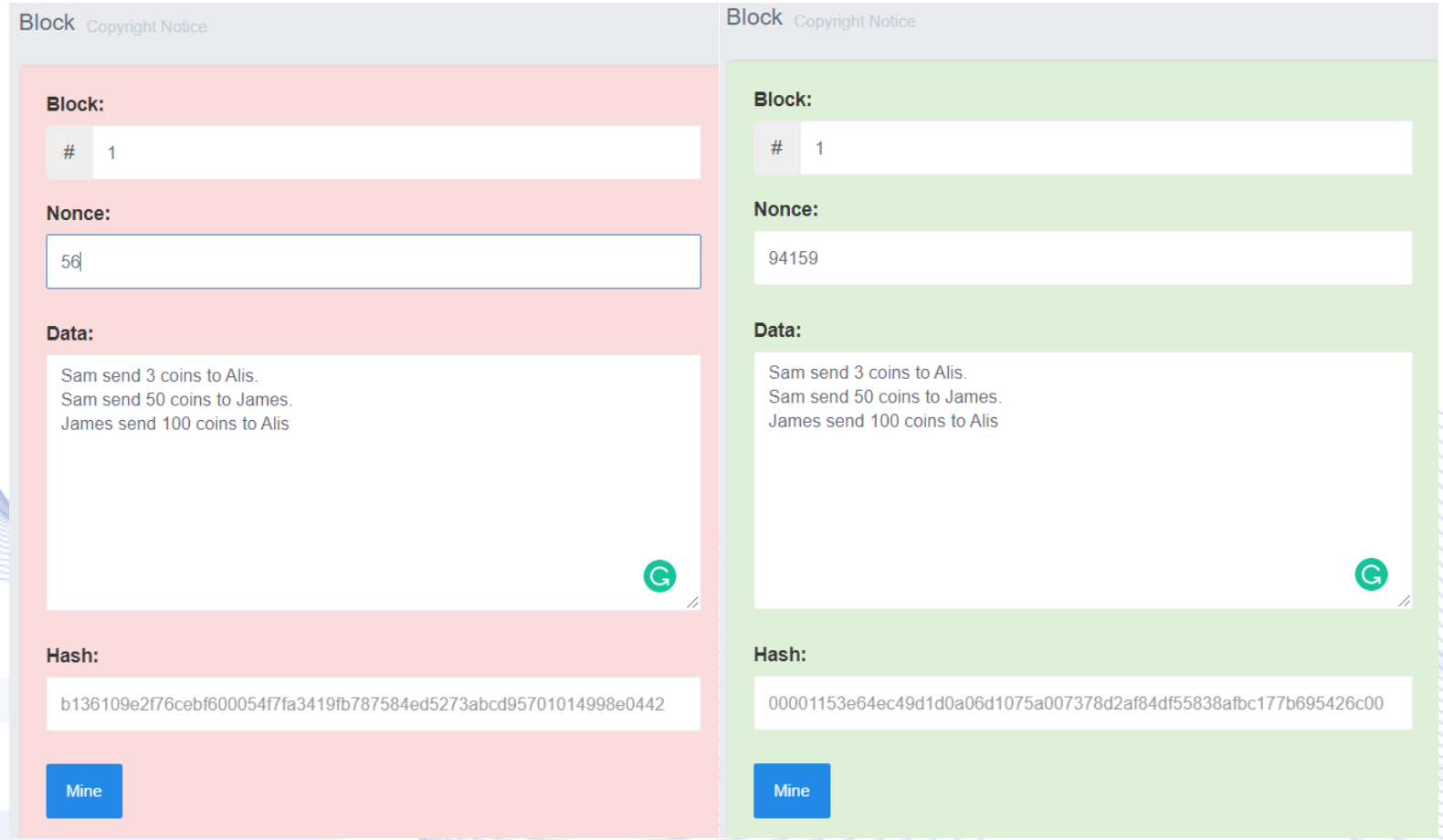
Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - **Proof of Work Consensus**
  - Proof of Stake Consensus
- The 51% attack

Proof of Work Consensus

- Proof of Work consensus protocol define the follow procedures:
  - ***The procedure of chain validation:*** Provides a Boolean judgment on whether a given chain has the valid structural properties. In more details, it checks if each block in the chain provides valid PoW solution (nonce) and no conflict between transactions as well as the historical records exists.
  - ***The procedure of chain comparison and extension:*** Compares the length of a set of chains which may be either received from peer nodes or locally proposed.
  - **The procedure of PoW solution search:** Defines a cryptographical puzzle-solving procedure in a computational-intensive manner.

Cryptographic Puzzle



Possible Hashes.

56

98

5664

8765

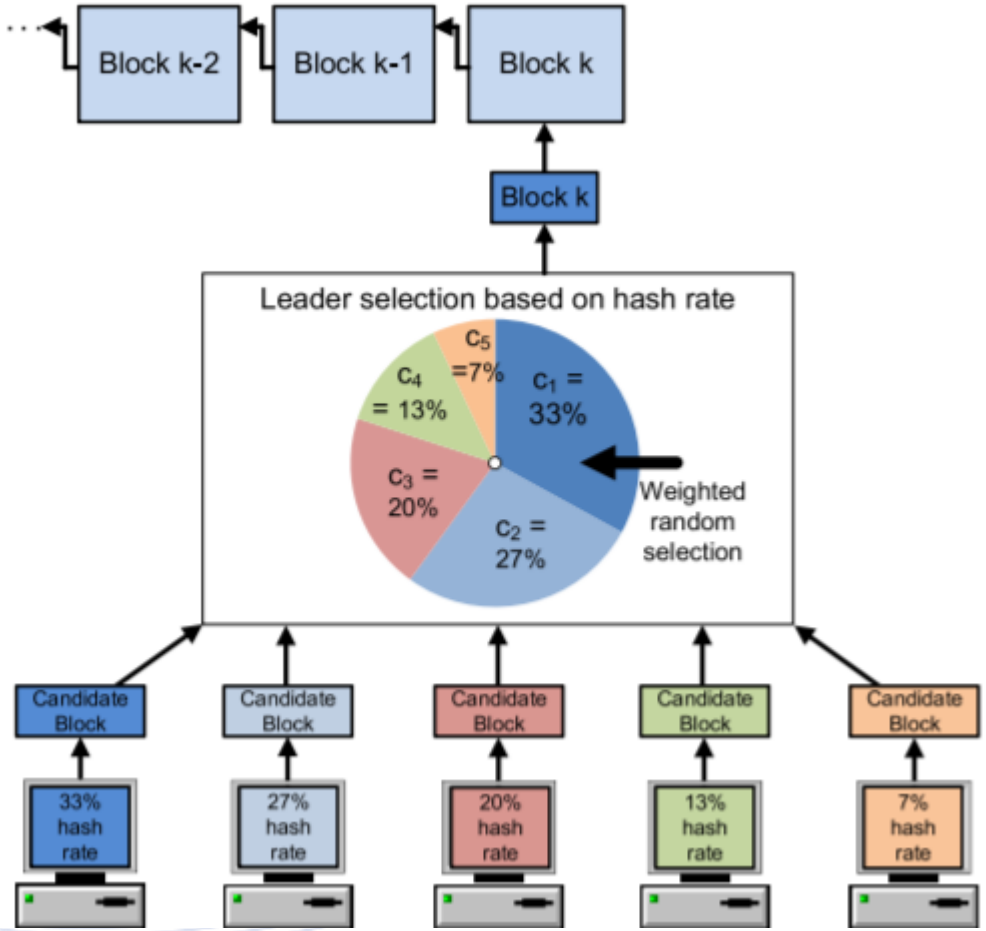
Target Value (“0000”)

94159

Mining Difficulty

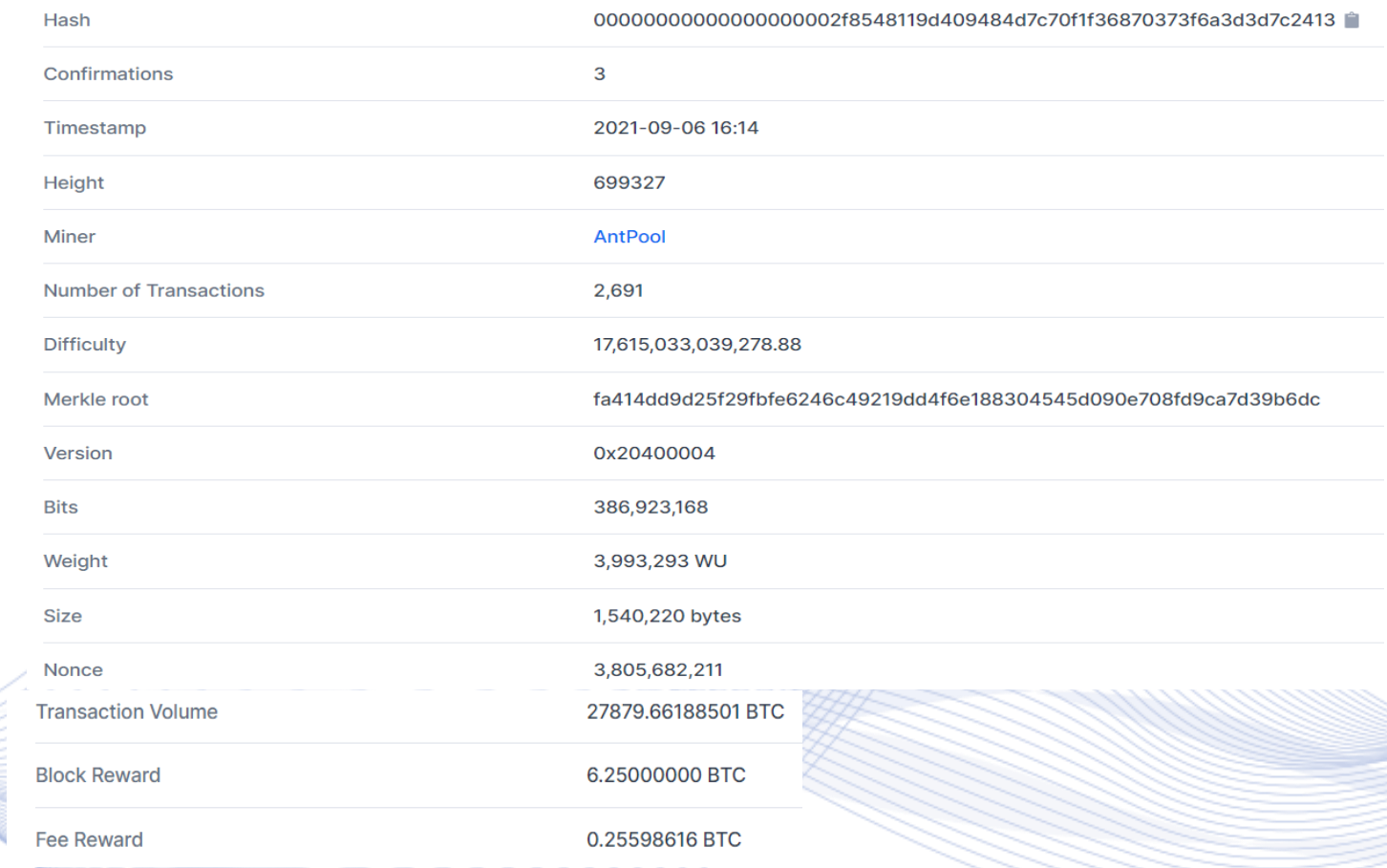
- The adjustable hardness condition (Difficulty):
  - Works as ***insurance*** that the candidate block will propagate to the whole P2P network before a new block is mined. Hence, it must adjust constantly so the block intervals have a constant value.
  - In case of bitcoin, it adjusted every 2016 block (e.g., two weeks) keeping in that way a constant 10 mins interval between new blocks. The adjustment process is regulated by the nodes themselves and the blockchain software.
  - A higher mining difficulty requires more brute force trials in order to find a winning nonce. Preventing in that way sybil or DDoS attacks.

Proof of Work Consensus



Source: [NYG2019]

Proof of Work Consensus

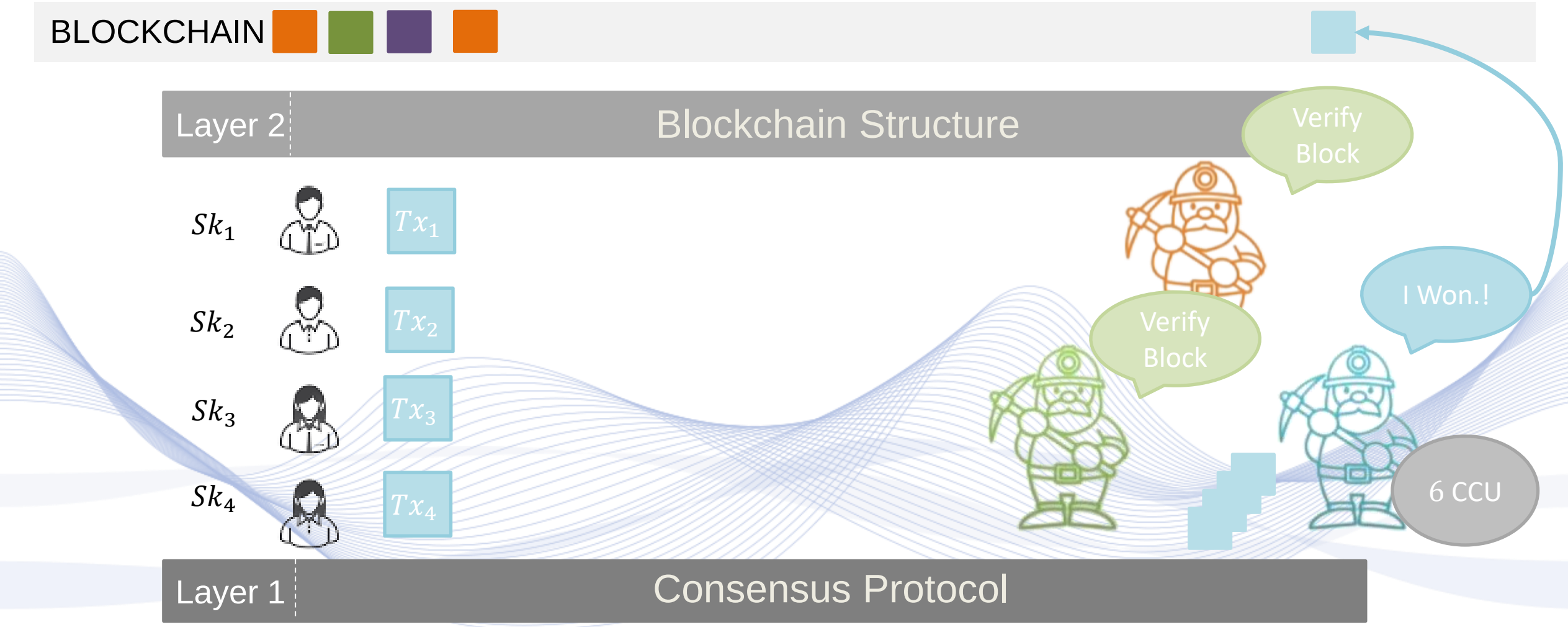


Hash	00000000000000000002f8548119d409484d7c70f1f36870373f6a3d3d7c2413
Confirmations	3
Timestamp	2021-09-06 16:14
Height	699327
Miner	AntPool
Number of Transactions	2,691
Difficulty	17,615,033,039,278.88
Merkle root	fa414dd9d25f29fbfe6246c49219dd4f6e188304545d090e708fd9ca7d39b6dc
Version	0x20400004
Bits	386,923,168
Weight	3,993,293 WU
Size	1,540,220 bytes
Nonce	3,805,682,211
Transaction Volume	27879.66188501 BTC
Block Reward	6.25000000 BTC
Fee Reward	0.25598616 BTC

Example of a block header in Bitcoin. Block #699327

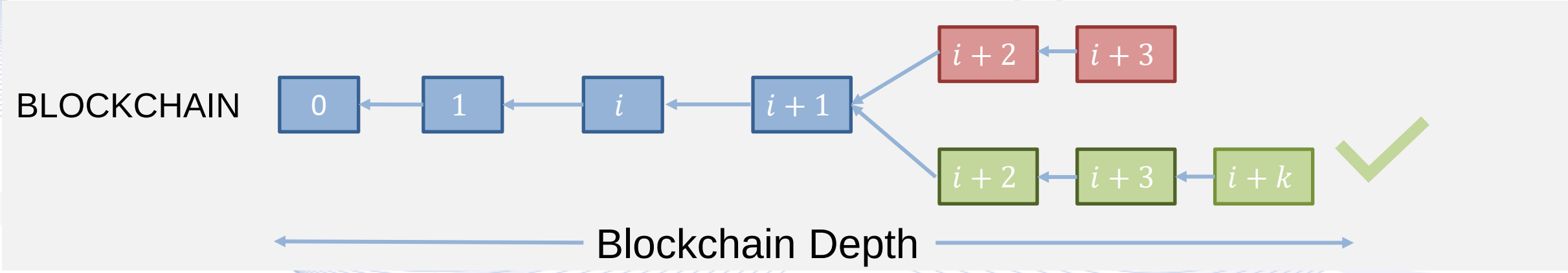
Source: blockchain.com

Proof of Work Consensus



Proof of Work consensus

- The blockchain structure starts with a genesis block at index 0 and links successive blocks in reverse order of their index. A new block is decided at index $i > 0$ when the blockchain depth reaches $i + k$.



Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - **Proof of Stake Consensus**
- The 51% attack

Proof of Stake Consensus

Proof of Stake is presented in order to reduce the limits of Proof of Work.

The basic idea in the PoS consensus algorithm:

- The nodes who want to be part in the process of creating new blocks must be able to ***prove*** first that they already have (own) a ***certain amount of assets*** (e.g., coins).
- And then to save a certain amount of them in escrow account called ***"stake"*** and from now on they can get involved in the consensus.

Proof of Stake Consensus

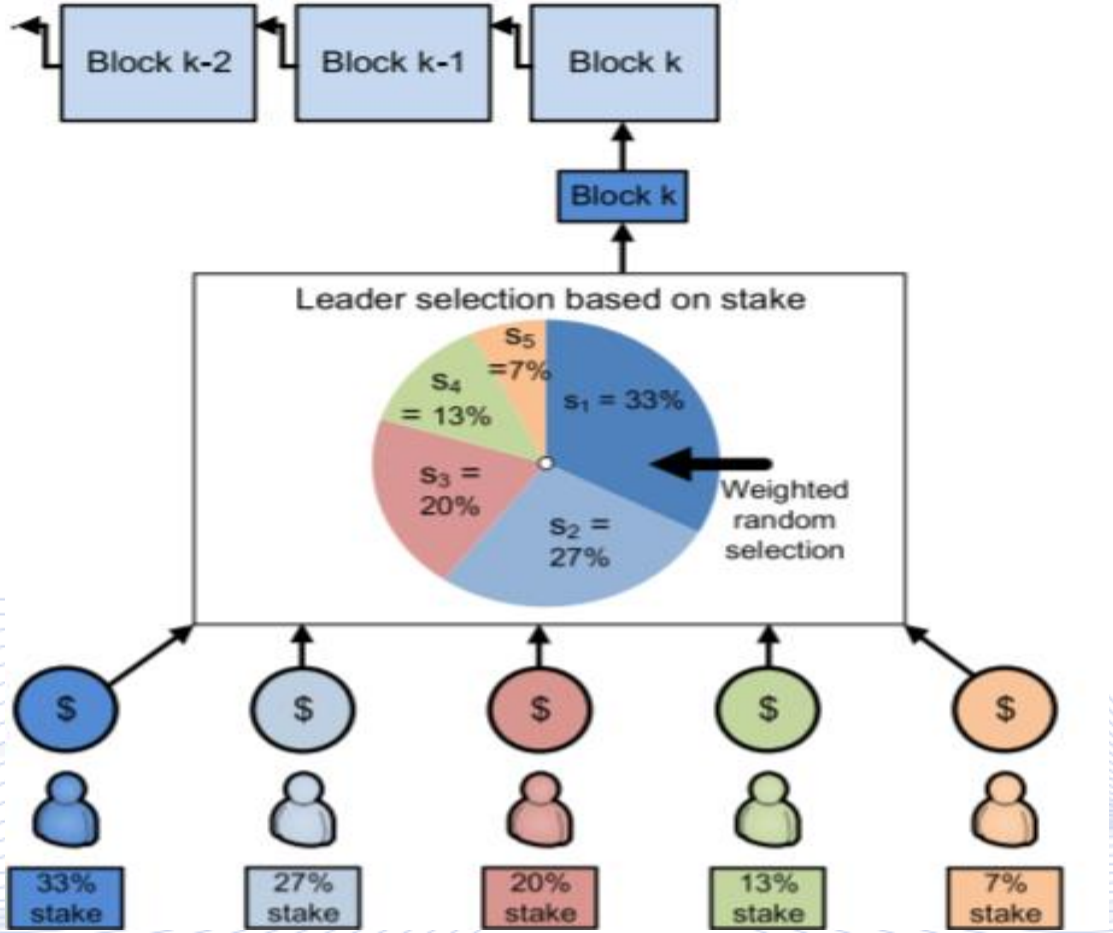
Penalty Rule:

- "Stake" works as insurance that the node, also called ***stakeholder***, will be corporate with the protocol rules. If the node misbehaves then it will lose his stake.

Once a new block is created by the stakeholder:

- Can obtain the transaction taxes within the block,
- Or may receive a number of coins that acts as a kind of expression of interest for its stake.

Proof of Stake Consensus



Source: [NYG2019]

Proof of Stake

- ***Advantages of PoS:***
  - ***Energy efficiency:*** PoS protocol does not demand from stakeholders to solve a computational hard problem.
  - ***Mitigation of Centralization:*** Anyone who is willing to enter in the decision-making process can enter without investing in ASIC/GPU mining rings in order to maximize the benefits.
  - ***Explicit Economy Security:*** Security is significantly enhanced through the penalty mechanism within the PoS protocols which makes it possible to detect anyone trying to perform a misbehaving attack. Once it is detected, the stakeholder loses his stakes and is likely to be excluded from any future block creation process.

Proof of Stake

- ***Limitations of PoS:***
  - ***Collusion:*** If there are not enough validators, it is easy to perform a 51% attack on the protocol by colliding with other validators.
  - ***Wealth Effect:*** The “token” model of the protocol among with the selections of validators can create a friendly environment for users with large number of coins, to obtain large influence over the system.

Blockchain Algorithms

- Introduction to Cryptography
- Blockchain
- The Byzantine Generals' problem
- Distributed System Consensus
  - Byzantine Fault Tolerance Consensus
  - Practical BFT
  - Blockchain Consensus
  - Nakamoto Consensus
  - Proof of Work Consensus
  - Proof of Stake Consensus
- **The 51% attack**

The 51% Attack

- The 51% attack is a theoretical technique which intends to fork a blockchain in order to achieve a double-spending. The idea is to create a malicious chain that can outgrow the honest one.

- Can be performed if a group of malicious nodes controls more than half of the total hashing power of the network.

- Often, blockchain networks with a low hashing power are vulnerable in that type of attack.

Bibliography

[LAM1982] L. Lamport, R. Shostak, and M. Pease. 1982. The Byzantine Generals Problem. ACM Trans. Program. Lang. Syst. 4, 3 (July 1982), 382–401. DOI:https://doi.org/10.1145/357172.357176

[ZHE2018] Zheng, Zibin & Xie, Shaoan & Dai, Hong-Ning & Chen, Xiangping & Wang, Huaimin. (2018). Blockchain challenges and opportunities: A survey. International Journal of Web and Grid Services. 14. 352. 10.1504/IJWGS.2018.095647.

[NOF2017] Nofer, M., Gomber, P., Hinz, O. et al. Blockchain. Bus Inf Syst Eng 59, 183–187 (2017).

[ACH2019] V. Acharya, A.Eswararao Yerrapati, N. Prakash, “Oracle Blockchain Quick Start Guide”, September 2019.

[NYG2019] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen and E. Dutkiewicz, "Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities," in *IEEE Access*, vol. 7, pp. 85727-85745, 2019.

[CAS1999] M.Castro, B.Liskov, 1999, “Practical Byzantine fault tolerance”. In Proceedings of the third symposium on Operating systems design and implementation (OSDI '99). USENIX Association, USA.

[XIA2020] Y. Xiao, N. Zhang, W. Lou and Y. T. Hou, "A Survey of Distributed Consensus Protocols for Blockchain Networks," in *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432-1465, Secondquarter 2020, doi: 10.1109/COMST.2020.2969706.

Q & A

Thank you very much for your attention!

More material in
http://icarus.csd.auth.gr/cvml-web-lecture-series/

Contact: Prof. I. Pitas
pitas@csd.auth.gr