Blockchain 101

Lecture Outline

- Distributed Systems Design
- Blockchain defined
- Consensus
- CAP theorem and Blockchain
- Decentralization

Lecture Outline

- **Distributed Systems Design**
- Blockchain defined
- Consensus
- CAP theorem and Blockchain
- Decentralization

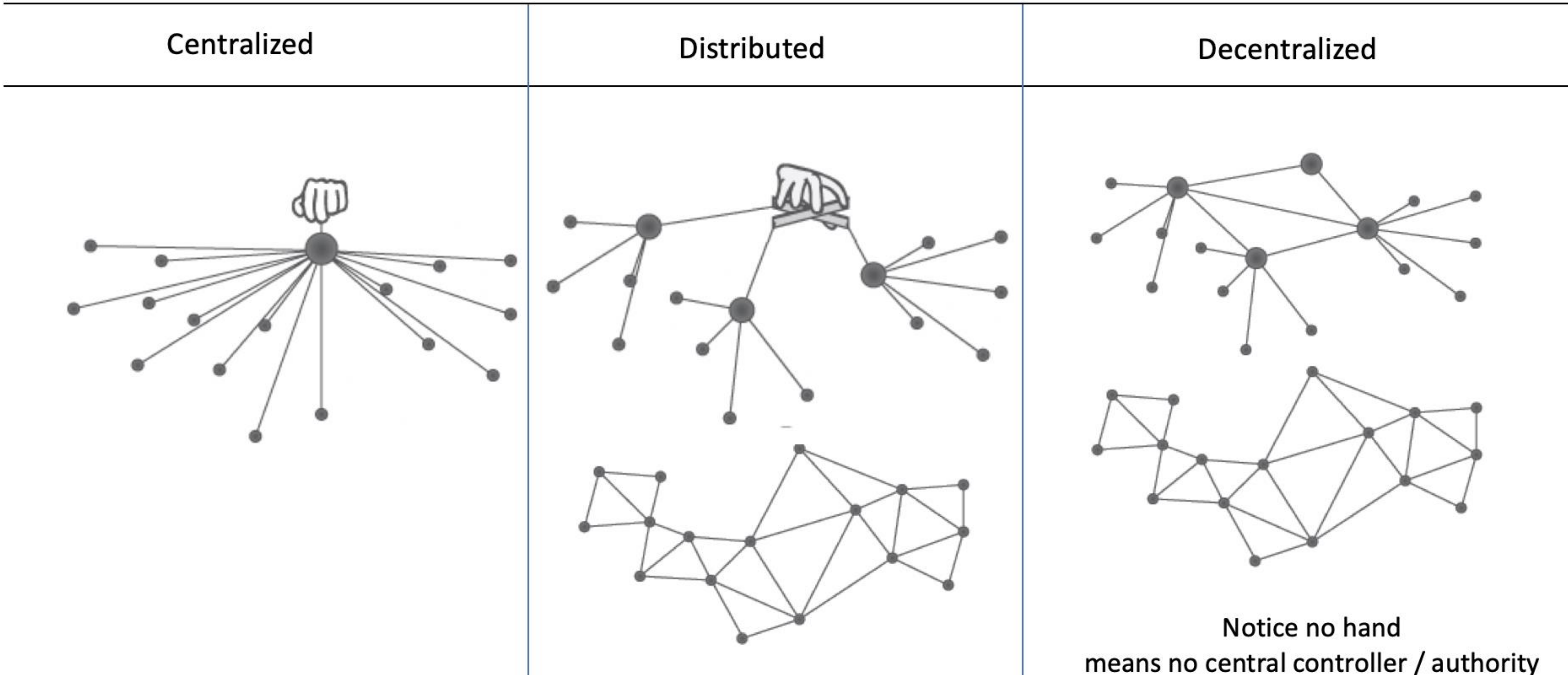
Distributed Systems

- Understanding distributed systems is essential to understand blockchain, as blockchain was a distributed system at its core.
- It is a distributed ledger that can be centralized or decentralized.
- It can be thought of as a system that has properties of the both **decentralized** and **distributed** paradigms. It is a decentralized-distributed system (decentralized vs. distributed?)

Distributed Systems

- **Centralized systems** are conventional (client-server) IT systems in which there is a single authority that controls the system, and who is solely in charge of all operations on the system.
- All users of a centralized system are dependent on a single source of service.
- In a **distributed system**, data and computation are spread across multiple nodes in the network.
- A **decentralized system** is a type of network where nodes are not dependent on a single master node; instead, control is distributed among many nodes.

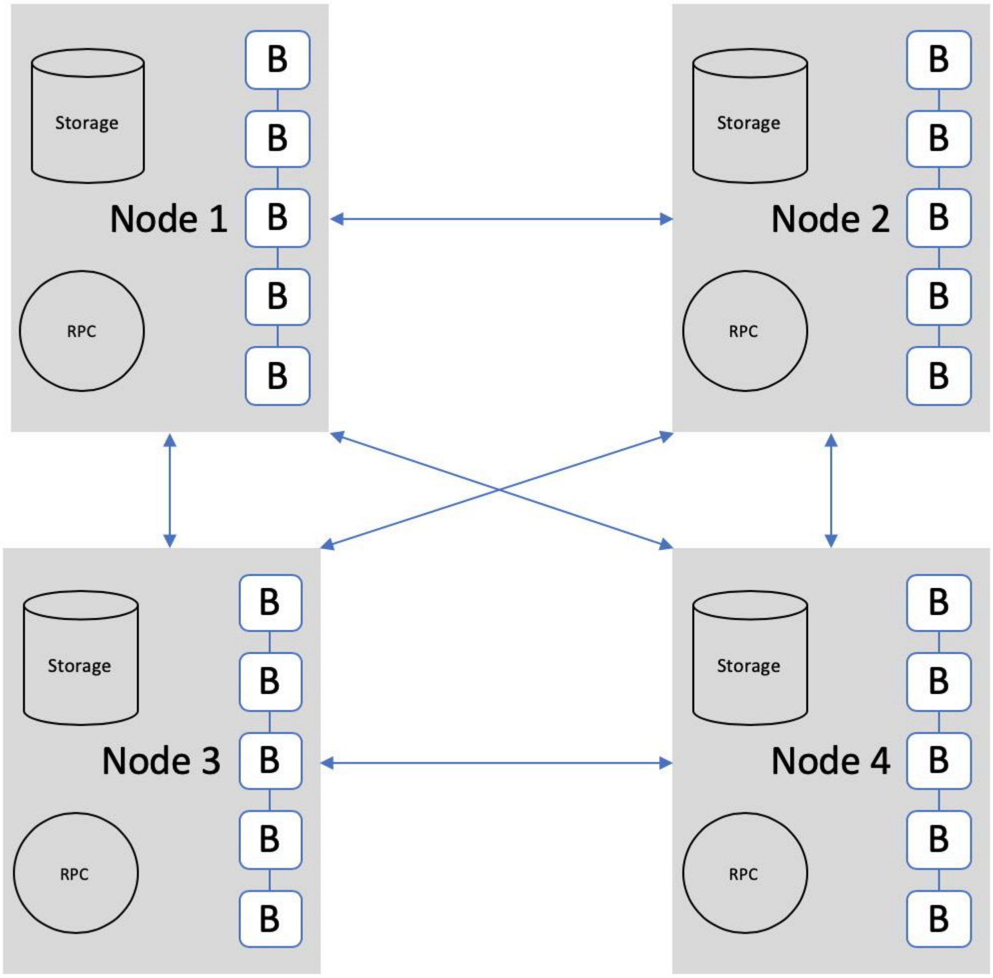
Decentralization using blockchain



In a distributed system, there is still a central authority that governs the entire system, whereas in a decentralized system, no such authority exists

Differences between distributed and decentralized systems

Decentralized



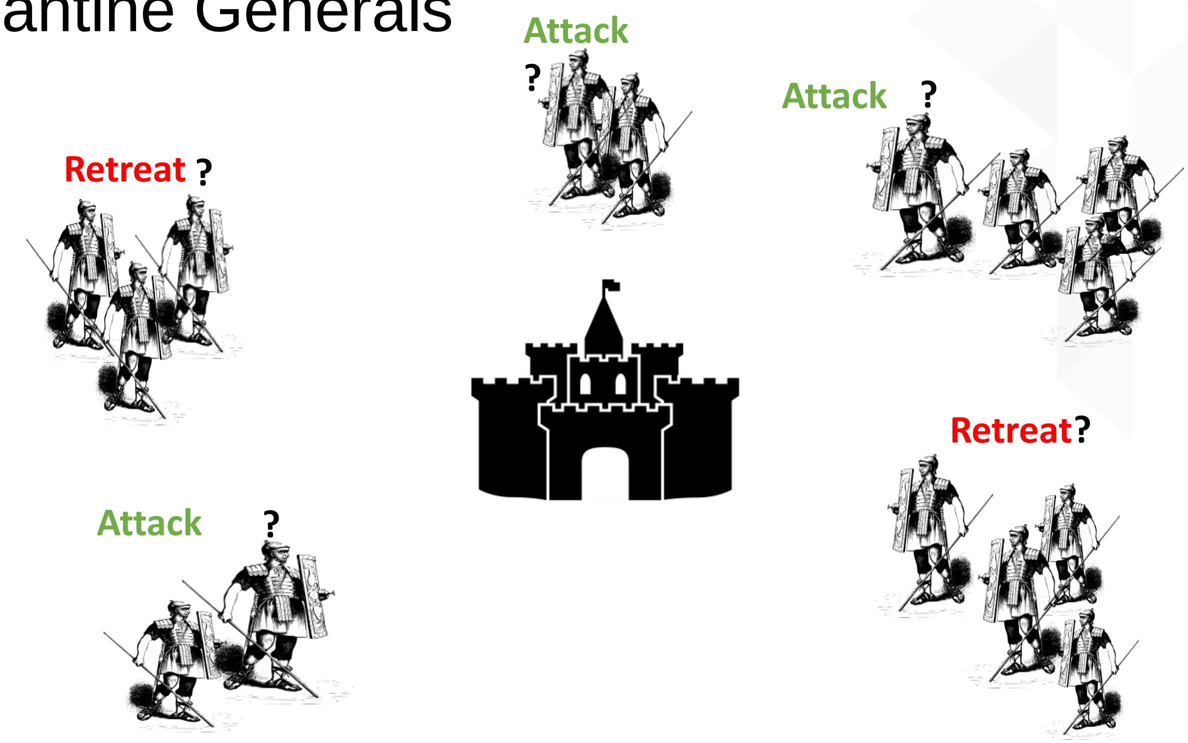
Distributed Systems (Cont.)

- Understanding distributed systems is essential to understand blockchain, as blockchain was a distributed system at its core.
- **Distributed systems** are a computing paradigm whereby two or more nodes work with each other in a coordinated fashion to achieve a common outcome.
- It is modeled in such a way that end users see it as a single logical platform.
- A **node** can be defined as an individual player in a distributed system.

Distributed Systems (Cont.)

- All nodes are capable of sending and receiving messages to and from each other. Nodes can be honest, faulty, or malicious, and they have memory and a processor
- A node that exhibits irrational behavior is also known as a *Byzantine node* after the **Byzantine Generals** problem

The Byzantine Generals problem



Attack or retreat?
Consensus required to win

The Byzantine Generals Problem

- A group of army generals who lead different parts of the Byzantine army is planning to attack or retreat from a city, where they only communicate among them is via a messenger.
- They need to agree to strike at the same time in order to win.
- The issue is that one or more generals might be traitors who could send a misleading message.
- There is a need for a viable mechanism that allows for agreement among the generals, even in the presence of the treacherous ones, so that the attack can take place at the same time.

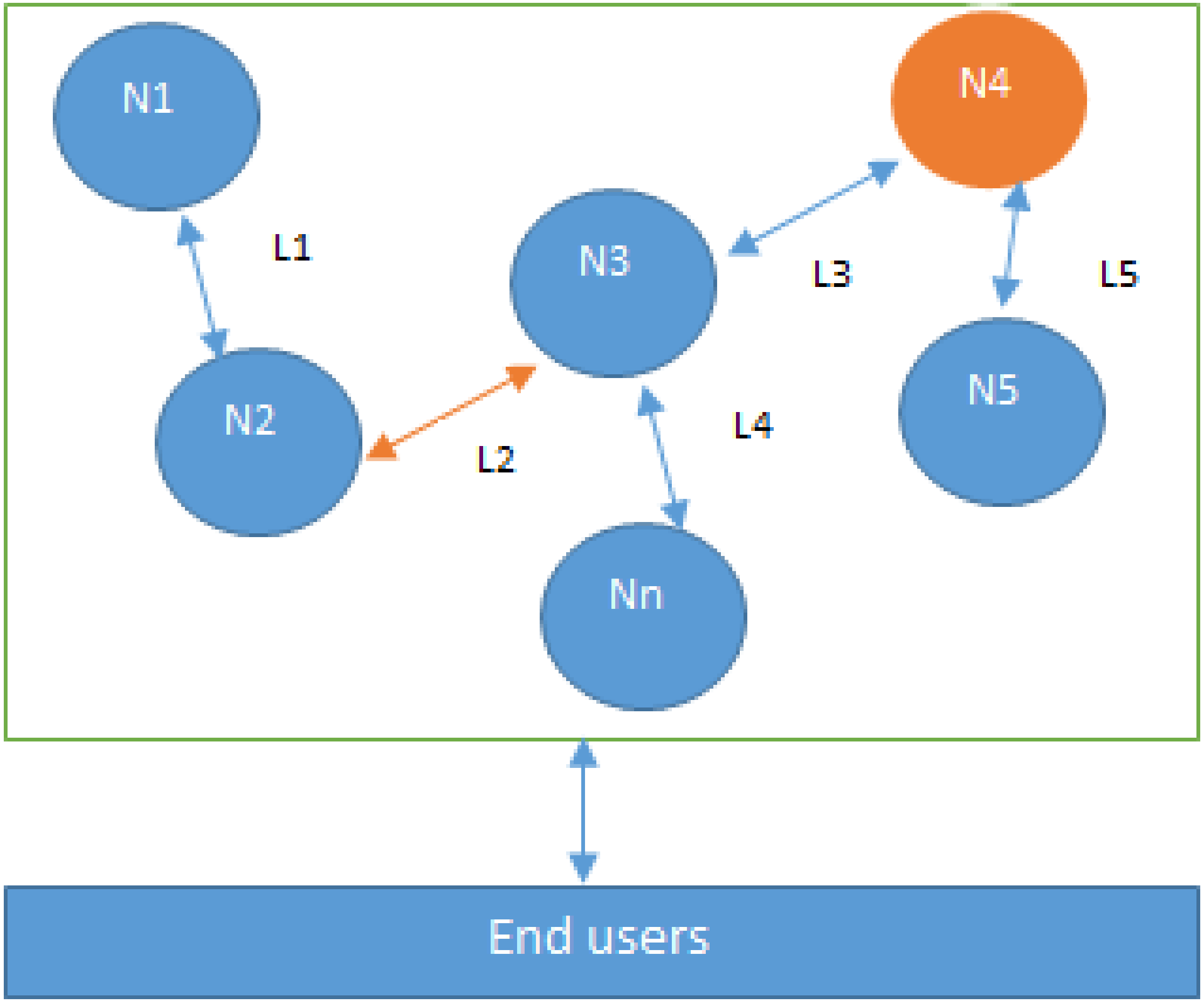
The Byzantine Generals Problem

- In 1999, Castro and Liskov presented the **Practical Byzantine Fault Tolerance** (**PBFT**) algorithm, which solves the consensus problem in the presence of Byzantine faults in asynchronous networks by utilizing the state machine replication protocol.
- PBFT goes through a number of rounds to eventually reach an agreement between nodes on the proposed value.

The Byzantine Generals Problem

- This type of inconsistent behavior of Byzantine nodes can be intentionally malicious, which is detrimental to the operation of the network.
- Any unexpected behavior by a node on the network, whether malicious or not, can be categorized as Byzantine.

Design of a distributed system



N4 is a Byzantine node, L2 is broken or a slow network link

Lecture Outline

- Distributed Systems Design
- **<u>Blockchain defined</u>**
- Consensus
- CAP theorem and Blockchain
- Decentralization

Defining ‘Blockchain’

Layman's definition: Blockchain is an ever-growing, secure, shared recordkeeping system in which each user of the data holds a copy of the records, which can only be updated if all parties involved in a transaction agree to update.

Technical definition: Blockchain is a peer-to-peer distributed ledger that is cryptographically-secure, append-only, immutable (extremely hard to change), and updateable only via consensus or agreement among peers.

Blockchain definition

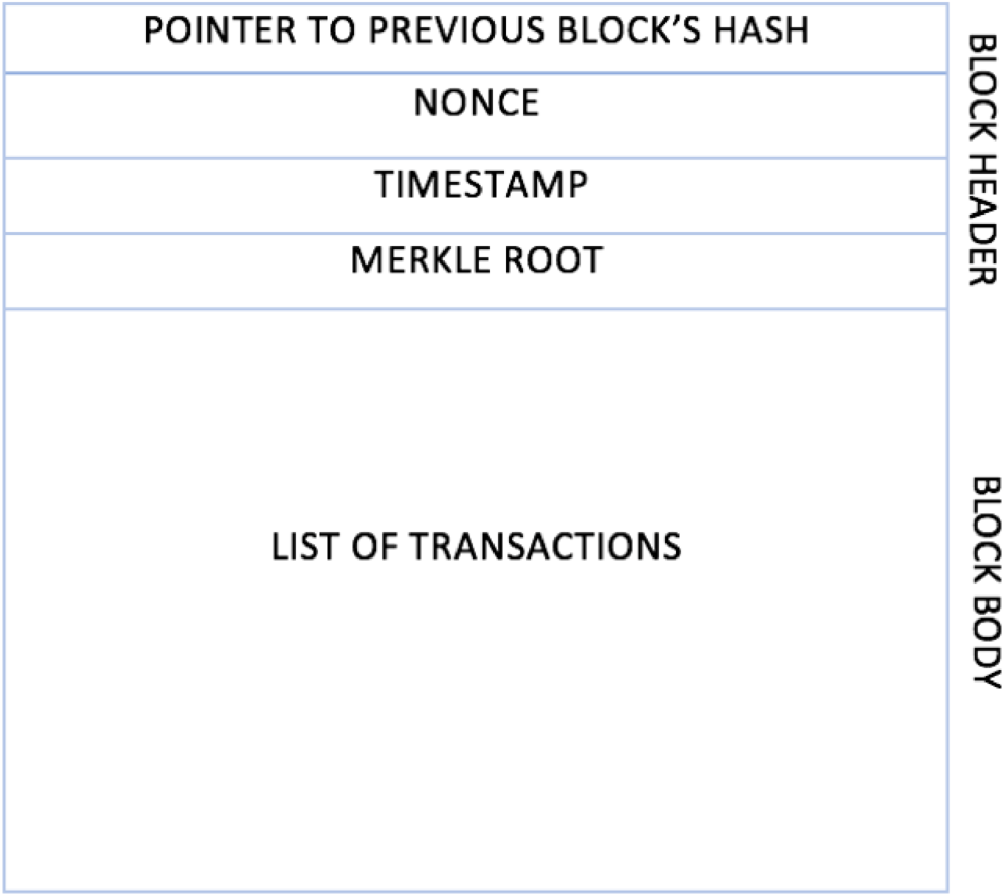
- Peer-to-peer: This means that there is no central controller in the network , and all participants (nodes) talk to each other directly. (Implication?)
- Distributed ledger: means that a ledger is spread across the network among all peers in the network, and each peer holds a copy of the complete ledger.
- Cryptographically secure: which means that cryptography has been used to provide security services that make this ledger secure against tampering and misuse. (e.g., non-repudiation, data integrity)

Blockchain definition

Can they be changed?

- Append only: that blockchain is "append-only," which means that data can only be added to the blockchain in *time-sequential order*. Blocks added to the chain cannot be changed, thus making the blockchain practically immutable.
- Updateable via consensus (consensus-driven): In this scenario, no central authority is in control of updating the ledger.
- Any update made to the blockchain is validated against **strict criteria defined by the blockchain protocol** and added to the blockchain only after a **consensus** has been reached among all participating peers/nodes on the network. (**Consensus algorithms**)

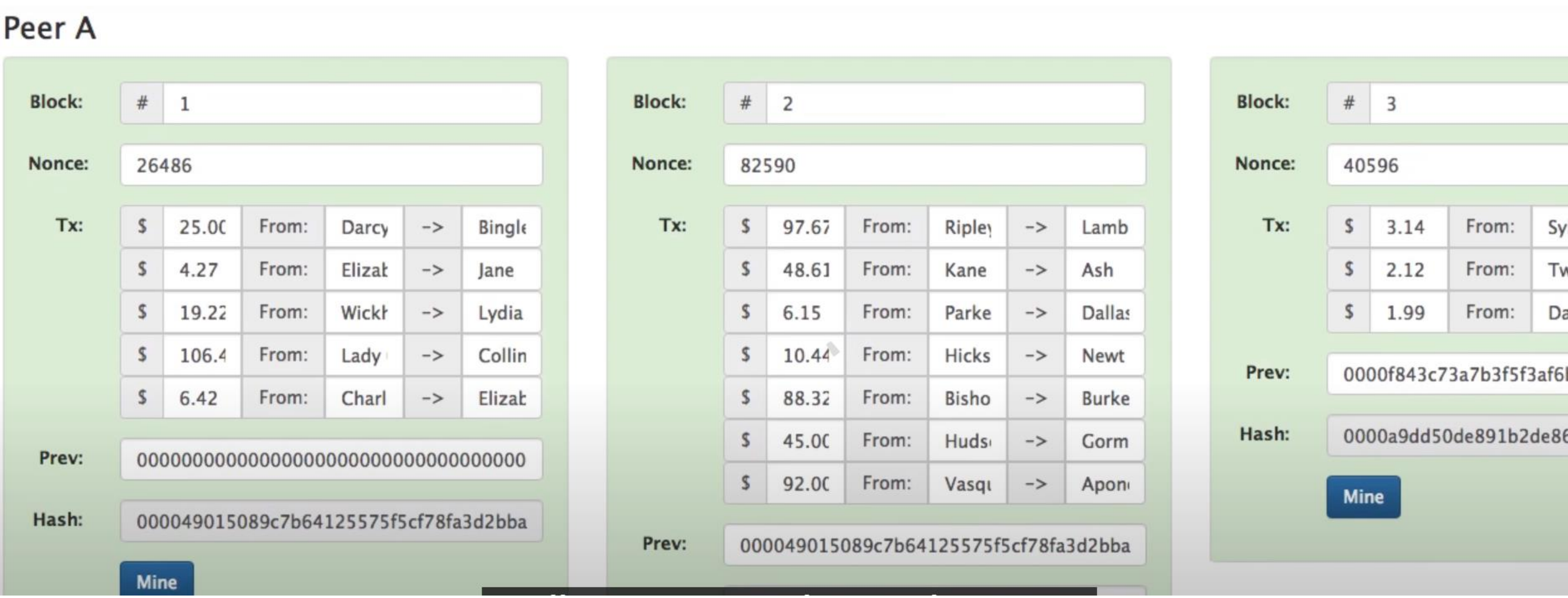
The Generic Structure of a Block



The Generic Structure of a Block

- **Address**: Addresses are unique identifiers used in a blockchain transaction to denote senders and recipients. An address is usually a public key or derived from a public key.
- **Transaction**: A transaction is the fundamental unit of a blockchain. A transaction represents a transfer of value from one address to another.

The Generic Structure of a Block



The Generic Structure of a Block

- **Block**: A block is composed of a block header and a selection of transactions bundled together and organized logically. A block contains several elements, which we introduce as follows:
    - A reference to a previous block is also included in the block unless it is a genesis block. This reference is the hash of the header of the previous block.
    - A **genesis block** is the first block in the blockchain that is hardcoded at the time the blockchain was first started. The structure of a block is also dependent on the type and design of a blockchain.

The Generic Structure of a Block

Block: A block contains several elements, which we introduce as follows:

- A **nonce** is a number that is generated and used only once. A nonce is used extensively in many cryptographic operations to provide replay protection, authentication, and encryption. In blockchain, it's used in PoW consensus algorithms and for transaction replay protection. A block also includes the nonce value.
- A **timestamp** is the creation time of the block.
- **Merkle root** is a hash of all of the nodes of a Merkle tree. In a blockchain block, it is the combined hash of the transactions in the block. Merkle trees are widely
  used to validate large data structures securely and efficiently

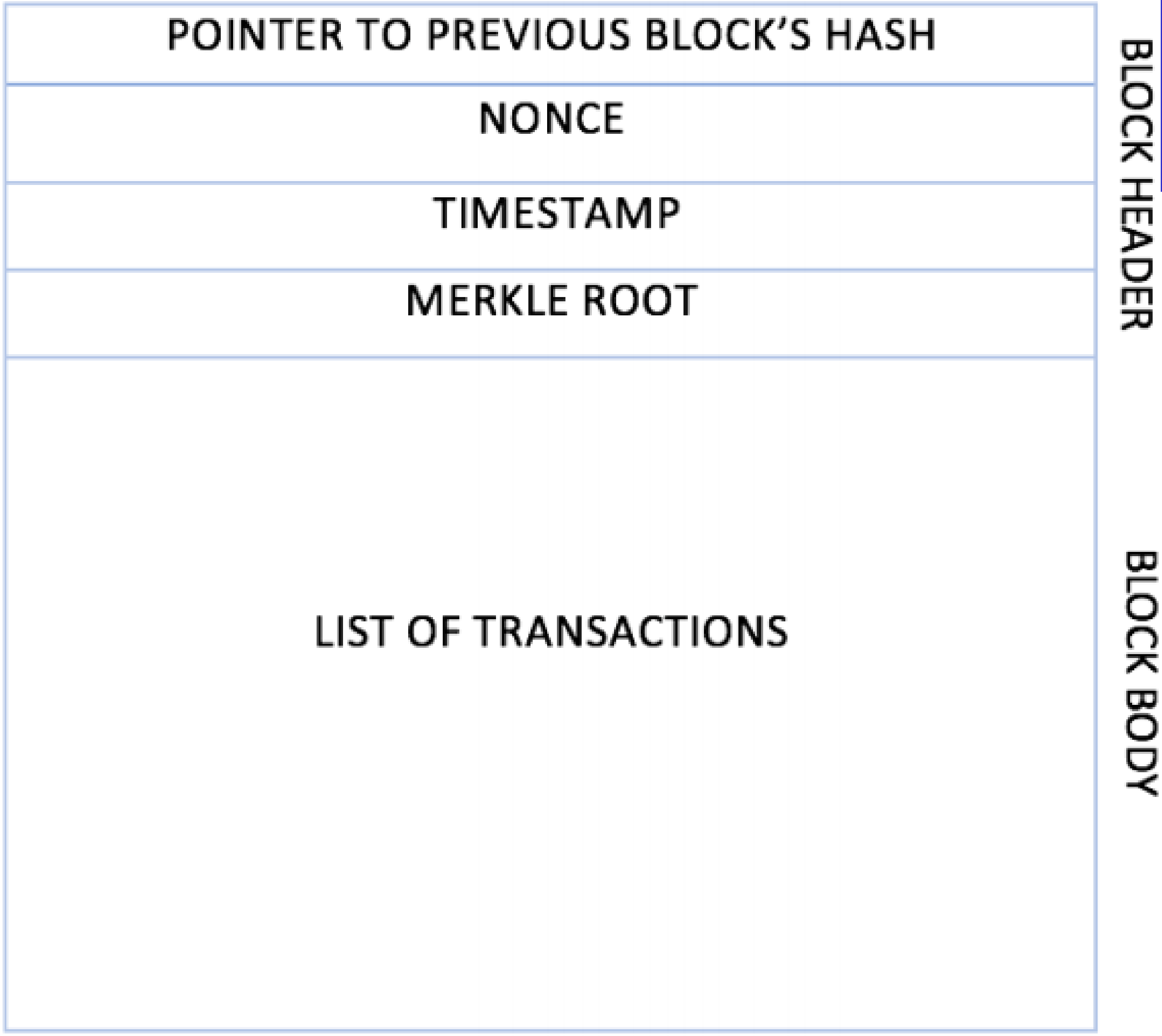
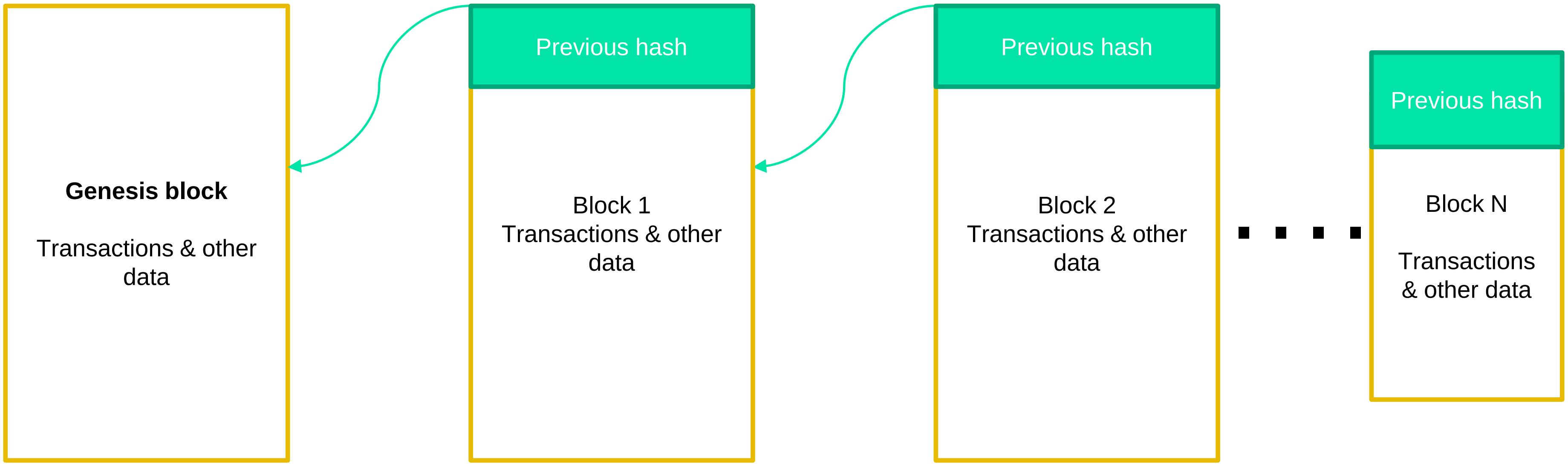


Figure 1.7: The generic structure of a block

Generic structure of a blockchain



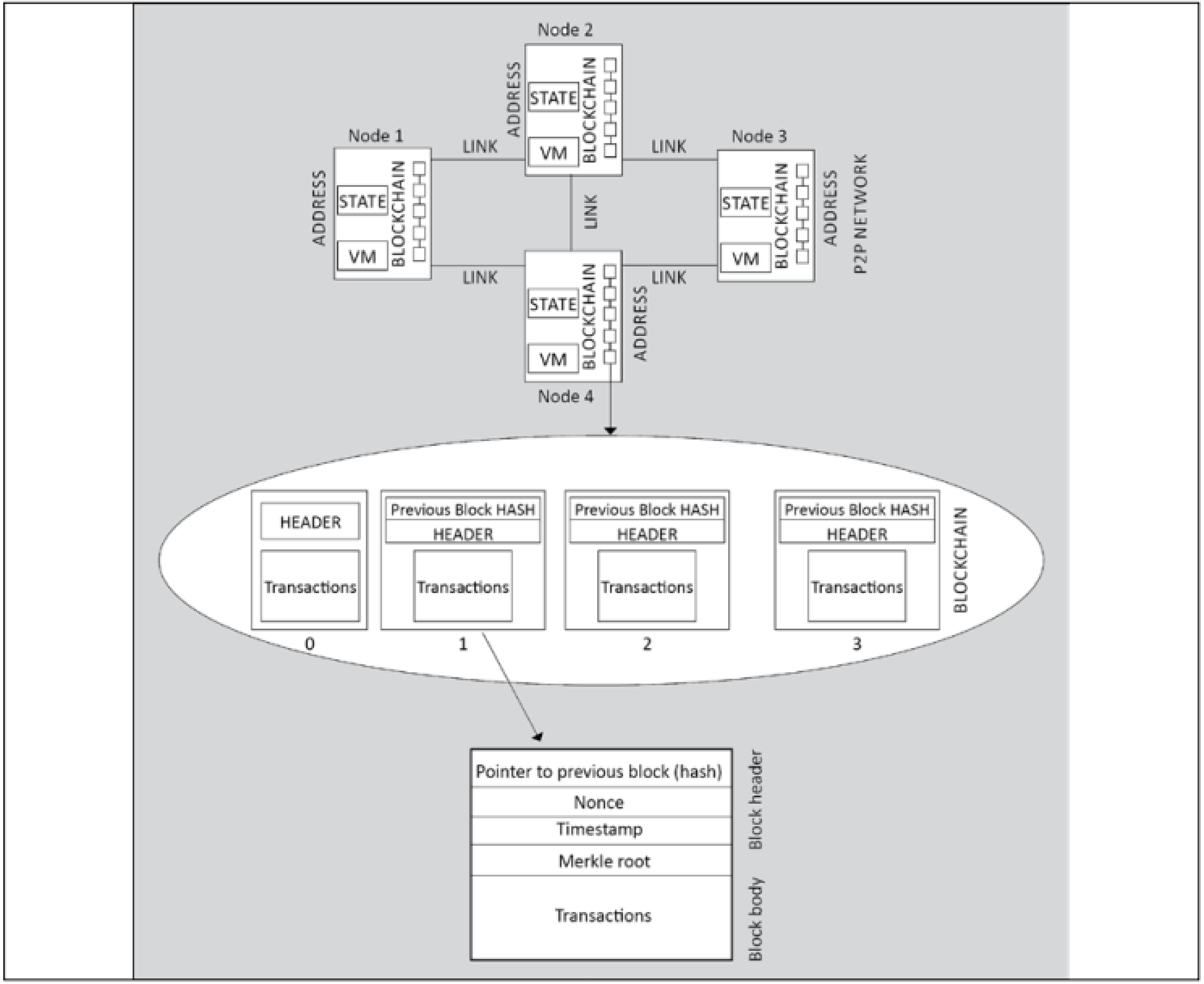
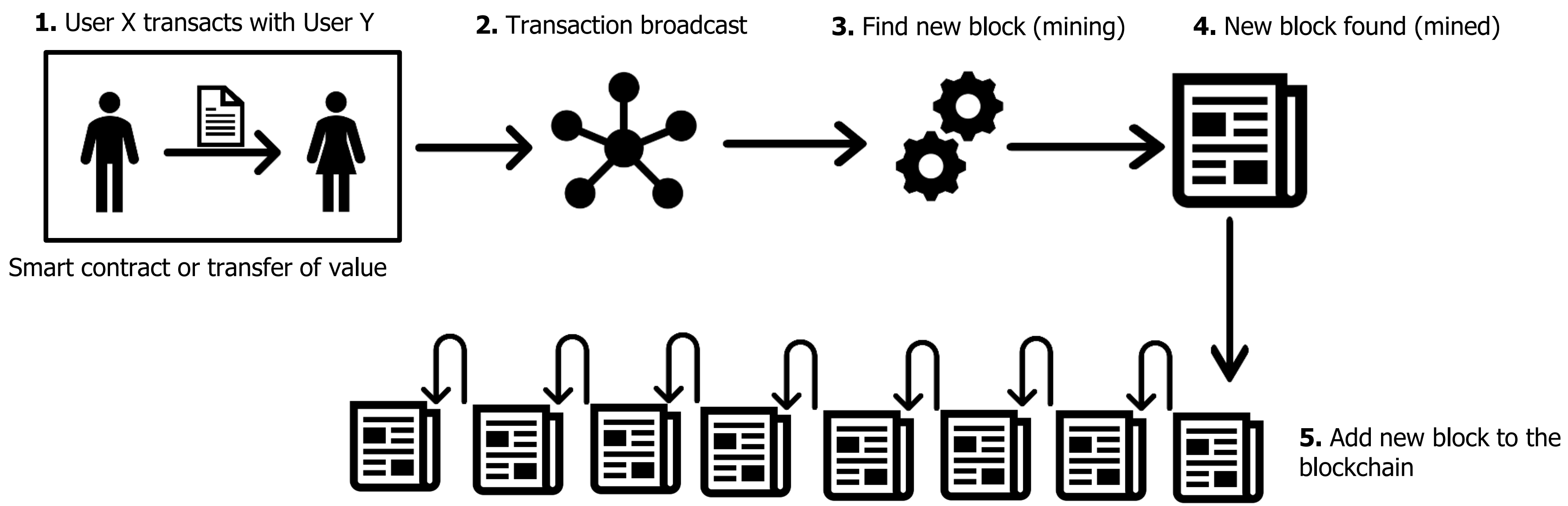


Figure 1.8: Generic structure of a blockchain network

How a blockchain works



What is a Hash Function

Cryptographic Hash Functions

Digital Fingerprints for Data

- General Properties
  - Maps Input **x** of any size to an Output of fixed size – called a 'Hash'
  - Deterministic: Always the same Hash for the same **x**
  - Efficiently computed
- Cryptographic Properties
  - Preimage resistant (One way): infeasible to determine **x** from Hash(x)
  - Collision resistant: infeasible to find and **x** and **y** where Hash(**x**) = Hash(**y**)
  - Avalanche effect: Change **x** slightly and Hash(**x**) changes significantly
  - Puzzle friendliness: knowing Hash(**x**) and part of **x** it is still very hard to find rest of **x**

Timestamped Append-only Log - Blockchain

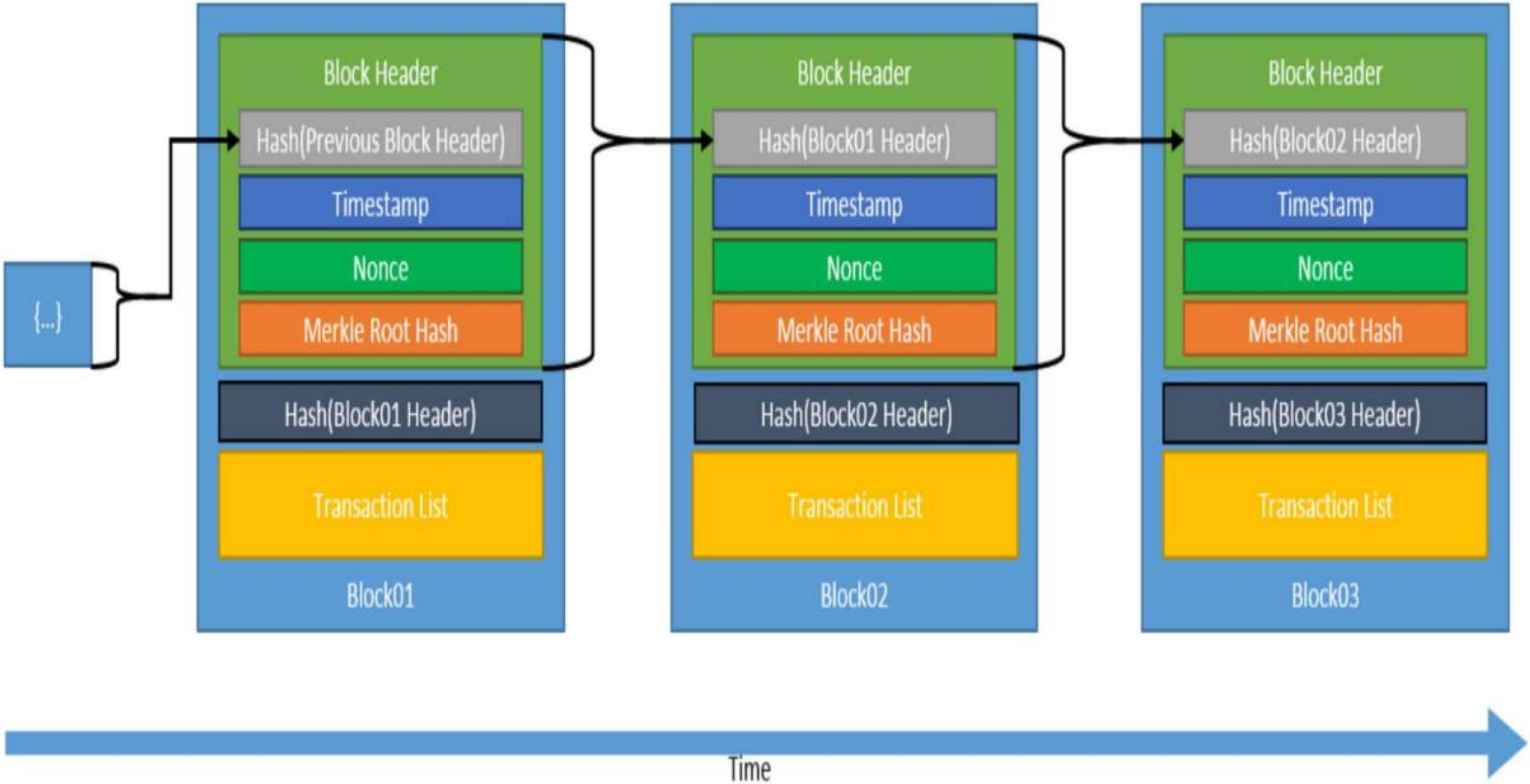
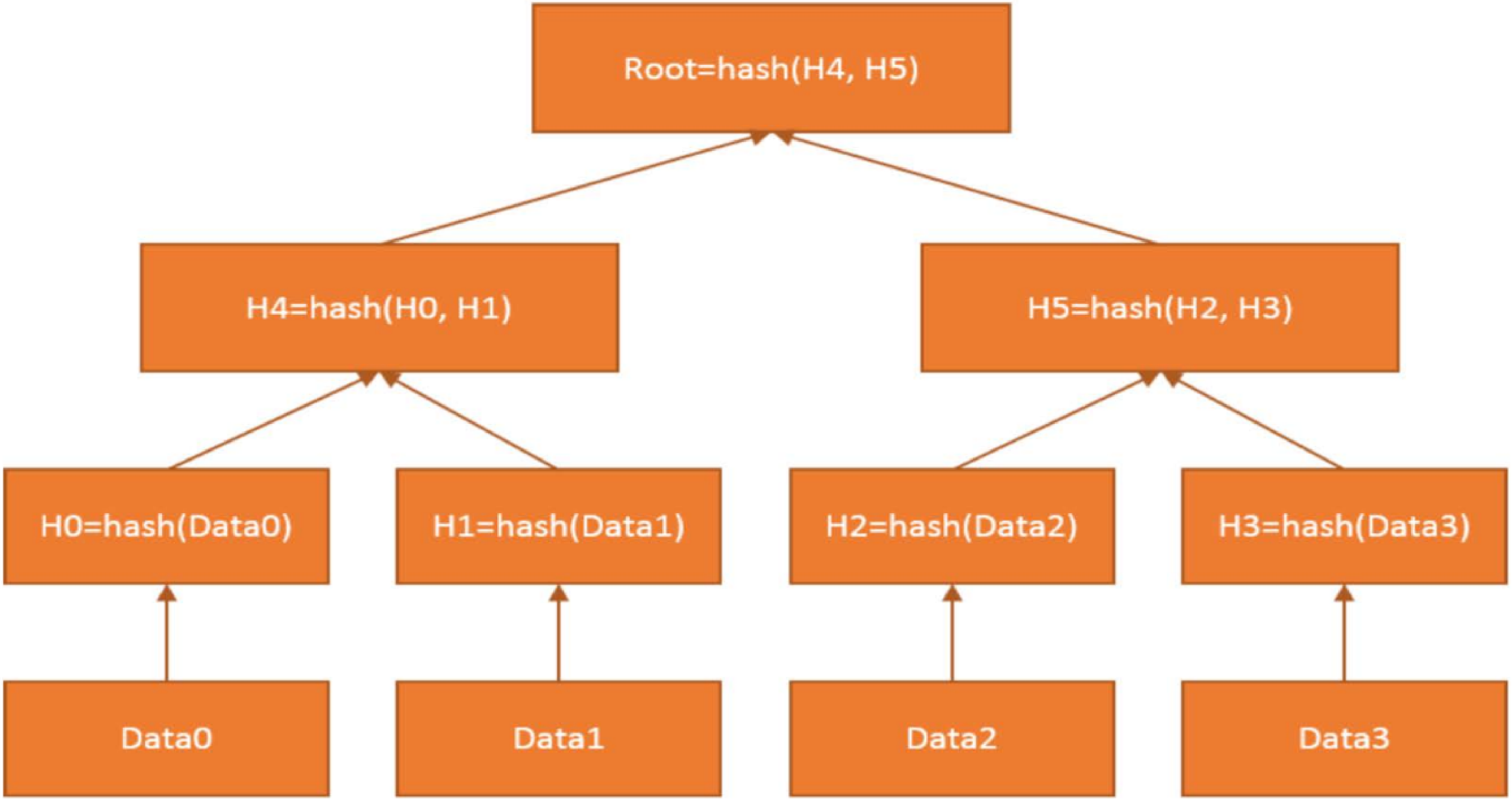


Image is in the public domain by National Institute of Standards and Technology.

- Block header:
  - Version
  - Previous block hash
  - Merkle root hash(hash for the current block)
  - Timestamp
  - Nonce

Merkle Tree – Binary Data Tree with Hashes



Required Readings/Videos

- Blockchain 101 – A Visual Demo: https://www.youtube.com/watch?v=_160oMzblY8&t=183s
- Live Demo https://andersbrownworth.com/blockchain/block
- Chapter 1 from “Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more, 3rd Edition”. Imran Bashir. Packt Publishing