blockchain

- blockchain: distributed network of computers that need to agree on what happens inside the network (transactions, data creation or editing, …) without having to trust each other

- blocks contain information/data

- blocks are identified and linked by hashes

- a hash is a value that summarizes something / a little bit like a signature

- proof of work: easy-to-check evidence that a computationally hard operation has been performed

- nonce: number that is hashed along a block that constitutes proof of work

- changing something in a block modifies its hash, and then this new hash is different from its neighboring blocks counterparts recorded original hash -> alert!

- mining: creation of a new block

Many angles to a blockchain’s definition:

1. it is a **ledger**: it records data and stores them in a way that is tamper-proof.

2. it is **chain of blocks**, literally: data are contained in blocks (computer/information objects) and identified by a signature (hash) that both makes them recognizable and works as their ‘address’ along the chain, and serves as link between blocks.

3. it is a data structure that puts emphasis on the **historicity** of the data that it contains: the blocks are added to one another in the order of their creation (mining).

4. it is structured in such a way (hashes) that every **attempt to modify the data** in a previous block automatically creates a discrepancy in the following blocks (through hash mismatching), thereby raising a flag.

**Blockchains work as data structures**, mainly for systems of distributed computers (although that is not their fundamental characteristic), that both:

1) record data from whatever activity they serve as a ledger for and

2) must agree on what gets written in the ‘official’ version of the data without having to resort to trust.

The data structure is as follows: it consists of **linked blocks (computer/information objects) that contain the data and a signature of the data**. That signature is called a **hash**. A hash is a value that summarizes the data it represents, without needing as much information (bytes) and is calculated by an algorithm. Every block contains its own hash and the hash of the previous block, thereby constituting the link. An exception is the first block, called the genesis block: it contains its own hash, but no previous block’s hash, obviously (hash = 0).

The utility of hash functions is that **a hash gets calculated every time a manipulation (legit or not) gets attempted on a block**. Since the hash’s value is tied to the data content itself, modifying the data (for example retroactively paying oneself more that owed) also modifies the hash, and since this new modified hash will be different from the following block’s ‘previous block’s’ hash, that will raise a flag in the chain.

The only problem with using a simple hash function (such as python’s sha256) is that it is easy **to (re)create an entire fake/manipulated blockchain** by recomputing all the hashes oneself. The solution to this problem is to structure the block creation so that it is both **difficult to compute and easy to check**. This is where the notion of **proof of work** enters stage. A proof of work is a computing task that must get carried out during the block’s creation process, and in such a hard way that it simultaneously couldn’t be done for an entire blockchain without massive computing power and that it is difficulty-scalable (adjustable to the level of the miner’s abilities or the amount of active miners), so that one couldn’t create a fake blockchain. An example of proof of work is the task of finding the prime factors of large integers. It take time to carry through, but once the solution has been found, it is easy to check. Example: the prime numbers of 969 = 3 \* 17 \* 19.

This control process is done during what is called the **mining phase**, when a created block gets reviewed by the decentralized network. In the case of Bitcoin, the miner that ‘wins’ the mining’ process (= first to post proof of work, for example) gets rewarded with Bitcoins. It should be noted that there exist other types of control processes (proof of stake, proof of burn, …).

The proof of work used in bitcoin, for instance, consist in finding the hash of a transaction ending with an arbitrary number of 0’s. **Solving the proof of work results in the adoption and network-propagation of a new block**. This criterion allows users to add blocks to the chain. The result of the proof of work computing is called the **nonce**.

**To sum it up**: with every new transaction (or any data addition in the ledger) that must be recorded, a new block is created, along with its hash. But in order for the created block (and thus the data) to get agreed on and propagated (= copied) to the whole network/ledger, the miner must first have solved the proof of work. Once this is done, that block gets integrated in the blockchain and the data is propagated to the network, making the blockchain history. Attempts to retroactively modify historical/previous blocks either result in a different hash than the one that is recorded in the whole network, or imply such massive computing power in order to recreate a fake blockchain containing the tampered data that it is unfeasible (notwithstanding the scaling condition: as the number of miners/computing power increases in the network, the difficulty of the proof of work automatically increases as well).