Blockchain Theory: Understanding the "Proof of Work"

Section 6, Lecture 106

In the last video, you learned about the "**Proof of Work**" (**PoW**). What is it all about?

In our blockchain, we're already validating that Block #2 has the same hash of Block #1 stored which we would get if we would re-calculate the hash value of Block #2 (which we do as part of our verify-chain method!).

**But this is not enough.**

Someone could edit Block #1 (e.g. manipulate the transactions stored there) and then also update the block after it (Block #2) to hold the updated hash in the previous\_hash  field. And that could then be done for the entire chain.

If we then validate the chain (on other nodes - we'll add nodes later in the course), the chain would be valid.

**That's a problem PoW solves.**

Just having equal hashes is then NOT enough anymore. Instead, a so-called PoW number (also referred to as "Nonce") is guessed for each block. PoW is just a number, but it's a number which, together with other input data, **yields a hash that suffices a condition defined by us**(=> the creators of the blockchain).

That hash **is NOT the hash we store in the previous\_hash  field**! It's not the hash we compare between blocks! We're simply using a hash for this check since it's convenient to check starting characters of a fixed-length (always 64 characters for SHA256) string.

A **typical requirement is for the hash to start with X 0s** (though you can come up with any condition of your choice). The more 0s we require, the longer it'll take to find a fitting PoW number - that's why for Bitcoin it can take multiple years to generate new block. It of course depends on your hardware setup and whether you're in a mining pool or not. It'll always take longer than just a few seconds though. For Bitcoin, a new block is added roughly every 10 minutes for example.

The hash we use to check the PoW typically takes the transactions that should be included in the new block as well as the previous block's hash and that PoW number into account.

And only if these three inputs yield a hash that starts with two 0s, we accept the hash and therefore the PoW. Since the transactions of the next block as well as the previous block's hash are static, only the PoW is adjusted until a matching hash is found.

The guessed (and fitting) PoW is then also stored in the mined block.

Other nodes can then easily confirm or deny the validity of the block since they just have to create a hash from the three input values (which are all known as they are included in the block). If the PoW doesn't lead to a valid hash, the block is not treated as valid and hence the overall chain is not accepted.

If some node tries to cheat, manipulating a block's transactions + the subsequent blocks previous\_hash  values **would NOT suffice anymore**. For changed transactions and/ or previous\_hash values, the old PoW wouldn't yield a valid PoW-hash anymore. A cheater would therefore have to re-calculate all PoW numbers for all subsequent blocks. This will take a huge amount of time since you typically don't check for two 0s but multiple leading 0s. And every additional leading 0 you add, vastly increases required computing time.

"Honest nodes" on the other hand only need to mine one new block in the time the cheater takes to re-calculate potentially dozens of blocks to render the cheater's chain invalid.

That's how PoW secures the blockchain.