CSC2125: Homework #4

Due on October 15

Fan Long

Thomas Hollis

Problem 1

Why the PoW problem has to consider the hash of previous block? What would happen if the miner only needs to find a nonce such that the one-way hash result of the concatenation of the nonce, the other block header information, and the transactions has enough leading zeros? Any security problem?

Solution

Most PoW protocols (like BTC) mainly rely on 3 types of hashes: public-private key hashes, Merkle tree hashes and block header hashes.

Public-private key hashes are used as part of the UTXO model to allow people to cryptographically generate addresses. Each receiving address is a 20-byte hash of the public key. These addresses are in turn used to send and receive funds.

Merkle tree hashes are used to hash each individual transaction up a Merkle tree into a single Merkle root. This root can be used to check that a transaction has not been tampered with without having to check the entire tree. The Merkle root hash is added to the block header.

Block header hashes are generated by hashing the block header twice (SHA256) to create a 32-byte hash (the block header hash). Since the header contains both the Merkle tree hash and the hash of the previous header, the block header hash is used to connect all blocks in the blockchain iteratively in order to establish a tamper-proof chain.

This question is referring to the block header hashes of the previous block. Indeed, the PoW needs to consider the hash of the previous block to ensure that the work being done is being done to mine on top of a particular block (rather than mining on an outdated or invalid block). Otherwise this would allow people to publish solutions to blocks before they are even added to the blockchain and the hierarchy would be lost.

The nonce in BTC is a 4-byte field whose value is by the network so that the hash of the current block is less than or equal to the target difficulty. Indeed, any hash which is smaller than the target hash results in the person who found this hash to win the reward. To find this hash you need to brute force all possible nonce to luckily find a hash smaller than the target. Therefore, the miner indeed needs to find a valid nonce with enough leading zeros (i.e. the small enough hash) that validates the hash result of the concatenation of nonce, block header and transactions. This can be done securely without causing any security issues due to the brute force nature of hashing described previously. However, picking any nonce (even an invalid one) just because it has the right number of leading zeros would definitely cause security issues. This is because if the only thing we look at is the number of leading zeros (and dont bother checking the hash is valid), then no work actually needs to be done. Users could simply submit any random hash of the correct size which is trivial as it has a complexity $O(1)$. Since we lose the difficulty of PoW, we therefore lose the trust consensus and the BTC protocol is broken.

Problem 2

What would happen if the PoW problem does not consider the timestamp? (Consider the case a malicious miner solves PoW once and put multiple different timestamp)

Solution

Timestamps are primarily used by the BTC protocol to establish difficulty. The faster the blocks get published (according to their timestamps) the larger the difficulty will become.

If the PoW problem did not consider timestamps it would not be able to track how fast blocks are being published. Thus, it would not be able to adjust difficulty according to the increased computational abilities. As difficulty remains constant while technology improves blocks would be published faster and faster up to the point where blocks will be published every second.

This is highly insecure as blocks no longer have time to propagate through the network. The reason this is insecure is because it violates the one CPU, one vote law established by Satoshi Nakamoto. This is because the bottleneck is no longer which chain is the strongest (has the most work). The bottleneck is now who has the best internet connection and can broadcast the most blocks as fast as possible.

Not only is it highly insecure but it is also highly wasteful. It would cause an extremely large amount of orphan blocks that are abandoned due to propagation delay. An interesting list of currently orphaned blocks can be seen here: `https://www.blockchain.com/btc/orphaned-blocks`.

In the case where a malicious miner solves a PoW once and submits this solution multiple times with multiple different timestamps, something unusual would occur. I believe that if he attempts to submit multiple PoW solutions the block that propagates the fastest will be the one that gets accepted, regardless of the timestamp. Only one block will survive, irrespective of the timestamp.

However, in the alternative case where a malicious miner submits multiple PoW solutions with multiple different timestamps, something surprising also occurs. If each block is mined on top of the previous block forming a chain and the entire chain is submitted at once with spoofed timestamps then the difficulty could be artificially altered. Indeed, if the spoofed timestamps are very close together (in time) then the difficulty will increase. Conversely, if the spoofed timestamps are highly delayed from each other, then difficulty will decrease as the protocol will believe that blocks have become harder to mine.

Problem 3

What would happen if the PoW problem does not consider the transactions?

Solution

If the PoW problem does not consider transactions then each block would not be examined for double spending within the block.

This is because if PoW solutions that do not consider transactions would be submitted and approved by all miners for any block as long as the other criteria are met, such as the number of leading zeros. Therefore regardless of the transaction contents of the new blocks, newly mined blocks would be added to the blockchain. Indeed, this will cause the presence conflicting transactions (including double spending) in the blockchain.

As a consequence, this would allow malicious users to successfully launch double spending attacks without needing over 50% of the hashing power. These fraudulent transactions would enable theft, contradiction and lying within the blockchain, nullifying the trust consensus originally created.

This would indeed cause a complete collapse of the BTC protocol rending it worthless.