Homework8

Part 1 (Off-chain Scalability)

Read the following materials about payment channel and lightning network:

1. Bitcoin Payment Channel:
   https://en.bitcoin.it/wiki/Payment_channels

2. Lightning Network Paper:
   https://lightning.network/lightning-network-paper.pdf

3. Lightning Network Slides:
   https://lightning.network/lightning-network.pdf

4. (Optional but helpful) Simplified article about lightning network:
   https://bitcoinmagazine.com/articles/understanding-the-lightning-network-part-building-a-bidirectional-payment-channel-1464710791/
   https://bitcoinmagazine.com/articles/understanding-the-lightning-network-part-creating-the-network-1465326903/
   https://bitcoinmagazine.com/articles/understanding-the-lightning-network-part-completing-the-puzzle-and-closing-the-channel-1466178980/

Then answer the following questions.

The timelock parameter in a Bitcoin transaction specifies a specific block number for the transaction to be unlocked. Why not use a timestamp directly? What could go wrong with using timestamp instead of block number?

Suppose a simple lightning network with three nodes, A, B, and C. There are two payment channels, one between A and B and another one between B and C. Now suppose A wants to send 10 BTC to C via this network. How many BTC, B has to have in order to proceed this transaction off-chain?

Now suppose the above scenario, what would happen if the payment channel between A and B expires before the payment channel between B and C? What could go wrong?

Do you think it is possible to support complicated smart contract transactions with the idea of lightning network on top of Ethereum? If so, how? (Open Question)

People claim that off-chain solutions like lightning network will make the blockchain systems increasingly centralized. Do you agree or disagree? Why? (Open Question)

Part 2 (Filecoin)

Read the Filecoin whitepaper (https://filecoin.io/filecoin.pdf). Then answer the following questions:

Can an attacker cause problems for a decentralized storage network that implements Proof-of-Retrievability but not Proof-of-Replication? Why?

The introduced proof of replication includes a Seal operation as part of the setup. Why do you think this Seal operation is required?

Is it possible that a piece of data can get lost? How and what happens then? Do you think this is a problem?