Assignment 4

1) What is a full node in blockchain?

Ans: Full nodes are full of data; they store and can distribute all of the blockchain data from the Ethereum network. A full node will additionally participate in block.An advantage of implementing a full node is that it can directly interact with any smart contract on the public blockchain. Full nodes can also directly deploy smart contracts into the public blockchain.

However, the full use and storage of data, as well as direct smart contract functionality, comes at a cost. Full nodes can be taxing on your computer's hardware and bandwidth resources. Retrieving full data can also be very time consuming, sometimes taking multiple days to sync your data when the node is first deployed. Then, the node must be maintained, upgraded and kept online in order to not have to repeat the full synchronization process.

2) Difference between remote client and light client?

Ans:

Remote Client: Similar to a light client. The main difference being, a remote client does not store its own copy of the blockchain, nor does it validate transactions or block headers. Instead, remote clients fully rely on a full or light client to provide them with access to the Ethereum blockchain network. These types of clients are predominantly used as a wallet for sending and receiving transactions.

light client :Ethereum clients may be implemented in full or in part. The above overview gives an explanation of how a "full" client works, however it is important to know that you don't always need to run a full client. Typically when data storage and speed are at issue, developers will elect to use what are called "light clients."Light clients offer a subset of the functionality of a full client. Light clients can provide faster speeds and free up data storage availability because, unlike the full clients, they do not store the full Ethereum blockchain.The scope of a light client's functionality is tailored toward the goals of the Ethereum client. For example, light clients are frequently used for private keys and Ethereum address management within a wallet. Additionally, they tend to handle smart contract interactions and

transaction broadcasts. Other uses for remote clients include web3 instances within JavaScript objects, dapp browsers and retrieving exchange rate data.

3) When should a full node be run?

Ans:

1) Helps the Network: Running your own full node is the only way to have full control and to ensure that all the rules of Bitcoin are being followed. Nodes do this by rejecting blocks and transactions that don't follow the consensus rules and by rejecting connections from peers that send them.

2) Keeps you safe: Running any kind of Bitcoin wallet that does not require you to run a full node means that you have to place a certain degree of trust in the service provider, something that shouldn't be necessary with Bitcoin.

A web-based wallet requires you to trust the central node through which your transactions are broadcasted and even lightweight clients like Electrum require that you trust their servers. Even an SPV client requires trust in a third party since you are only downloading the blockchain headers and not verifying if the blockchain you follow respects the rules of the network or if it's the longest blockchain.

3) Privacy: Using a centralized, lightweight or even SPV wallet will never be as private as running a full node. Since you rely on third-party servers to broadcast your transactions for you, those servers will be aware of which addresses belong to you. This is true both for online wallets and lightweight clients. SPV nodes that rely on bloom filters leak considerable information about the addresses of Bitcoin users.

4) If I want to mine new blocks then I should run a full node. True or False.

Ans: True.

This allows light nodes to connect to and transact on the Bitcoin network, without the need to store its full history. Importantly, however, this means that light nodes cannot independently verify the Bitcoin network rules and, therefore, must connect to full nodes in order to get the block data.

5) What are upgradeable smart contracts?

Ans: A smart contract upgrade involves changing the business logic of a smart contract while preserving the contract's state. It is important to clarify that upgradeability and mutability are not the same, especially in the context of smart contracts. A deployed program to an address on the Ethereum network still cannot be changed. But you can change the code that's executed when users interact with a smart contract.