Homework5

Achieving consensus in distributed systems with arbitrary (Byzantine) failure is a classical problem that has been studied by researchers for twenty years. Read the Practical Byzantine Fault Tolerant paper (http://pmg.csail.mit.edu/papers/osdi99.pdf) and answer the following questions. Note that optionally, you may also want to read Tendermint white paper to see the application of Byzantine Fault Tolerant in blockchain (https://tendermint.com/static/docs/tendermint.pdf).

Suppose we run PBFT algorithm on a cluster of 21 machines. What is the maximum number of machines we will be able to tolerant for failure simultaneously?

A node in PBFT waits for prepare/commit messages from **two thirds of nodes** (replicas) during the prepare/commit stages. What would happen if the node instead waits for prepare and commit messages from **only a simple majority** of other nodes? Would the modified PBFT be secure? If not, please explain with a counter example.

PBFT relies on the primary node to send out pre-prepare messages to drive the consensus process. What would happen if the primary node is malicious or fails? How does PBFT handle this situation?

Could PBFT skip the pre-commit messages? Suppose nodes in PBFT commit-local directly after receiving 2f + 1 prepare messages. What could go wrong? Please explain with a counter example (Consider the case where a view change happens immediately after one or two nodes commit-locally for a request).

A permissioned blockchain is a blockchain system in which the membership of the blockchain is fixed and known to all participants. The membership may be managed offline by a potentially centralized organization. Explain how PBFT could be used to implement such a permissioned blockchain.

Suppose the number of replicas in the system is N. During the normal-case operation in PBFT, roughly how many messages will be broadcasted for each request? Suppose the broadcast is implemented via point to point transmission between nodes one by one. What is the total number of transmitted messages for processing each request? (Answer in Big O notation like O(1), O(N), O(N^2), …)

Suppose the number of replicas in the system is N. Suppose the broadcast is implemented via point to point transmission between nodes one by one. What is the total number of transmitted messages during a view-change in PBFT? What is the size of each message during a view-change? (Answer in Bit O notations like O(1), O(N), O(N^2), …)

Given the above analysis, explain why it might be impractical to use PBFT to implement a large scale blockchain.