We divided the implementation broadly into three files:

**1.      Prop\_node.py**

**2.      Network\_ctrl.py**

**3.      Client.py**

**Network.ctrl.py**

It contains the network implementation i.e it has the methods defined to send messages across different

servers based on their IP address. The important methods are listed below:

* **wait\_for\_ctrl\_connections** – This method is run in a background thread in each server so that servers can read messages from other servers.
* **send\_ctrl\_message\_with\_ACK** – This method has 5 parameters :
  + Message -  The data you want to pass.
  + messageType – What type of message it is.
  + extra – Any extra data you want to send.
  + requestNode – The node to which you want to send this message.
  + timeout – the timeout value

**Prop\_node.py**

This is the main file where the BFT Raft algorithm is implemented

Main methods:

* **join\_network** – Used by servers to join a cluster
* **start\_leader\_election** – Used by a server to start leader election
* **stabilization\_routine** – Used by servers to know about other servers in the cluster.
* **heartbeat\_routine** - As soon as a server becomes a leader it starts sending out heartbeat signals to existing nodes in the system to show that it is alive.
* **leader\_timeout\_routine** - When a node in the system does not receive heartbeat signal from the leader within its timeout period, it announces itself as a CANDIDATE and starts election by calling start\_leader\_election

**Client.py**

Client has **two major tasks.**

* Send commands to be executed to the server.
* If the command takes too long to get executed it intervenes to start a new leader election.

sudo mn --topo single,5

Leader Election test cases

1. When only <=3 servers in the system no election is possible.
2. When 4th server joins, one of the servers times out and starts leader election. Leader is elected and heartbeat mechanism is working properly. Tested this by inserting an adversary server who sleeps for time greater than leader timeout, in which case the FOLOWER times out and announces itself as a CANDIDATE and starts election. When someone joins after leader is elected, that node is able to get heartbeats.
3. When a node joins after election, it has to figure out the current leader. With heartbeat, we also send the server information. So, the newly joined node knows who the leader is. As of now we assume that heartbeat sending node is the leader. Also, when a CANDIDATE announces itself as a LEADER, we send a quorum which the other nodes in the system verify. This quorum right now is just a list of IP addresses the CANDIDATE got its votes from.
4. I have created 5 servers. Until there are minimum 4 servers no election is possible. If leader server dies and there are still 4 nodes in the system a new leader is elected. When we kill 2 servers, only 3 remain. So, all the nodes become FOLLOWERS again as no election is possible. When a node joins again, election is triggered and new leader is elected.

Client Intervention test cases:

1. When client does not get a reply from the PBFT servers in a stipulated time frame, it asks for a new leader to be elected. We are working on certain testcases which are still troublesome.