Distributed Systems Coursework

Rpxr98

1a)
The generic workflow of a distributed system that is implementing passive replication begins with the frontend sending a request to one of the servers. This is the 'master' server. The 'master' server processes the responses in the order it receives them checking each time to make sure that it has not already handled the request. If this is the same request as before, it just sends the previous response again. The 'master' then executes the request and propagates the requests to the two 'slave' servers. These then precede to send confirmation of the request arriving and being processed. The primary finally returns the response to the front end.

1b)
The way passive replication ensures that the information is available is by swapping the master server if it fails. This means that an up-to-date 'slave' server becomes the 'master' meaning the data is still available to the front-end. A distributed system implementing passive replication can survive 'k' server failures in a system containing 'k+1' servers. This is a is a very effective way of maintaining data availability as if these systems are properly distributed the probability of them each going down should be independent from each other and therefore the chance of the system not having accessing the data greatly decrease even with only a few servers added.

3a)
When the 'master' server goes down, the frontend chooses a new 'master' server. This means that all the servers must be capable of being both a 'master' and a 'slave' server. Once a new 'master' is selected, the 'slave' server must change itself to being a master and then the distributed system can function normally except for the fact there is one less 'slave' server. Once the old 'master' server recovers, the new 'master' needs to reset the old 'master' server. Resetting the server would have to involve setting it to be a slave and adding all the data to the server. There are a couple of ways that the 'master' server could update the fallen 'master' server. The first being to clear all the data on the server and then send as complete set of data to it. The second would be for the server to determine which data the fallen server is missing.

Running Order

Server0.py-->Server1.py-->Server2.py-->Frontend.py-->Client.py
I have made the ports and host static as in terms of system resilience using a name server would be worse so therefore several ports need to be free. (50610,50611,50612,12001)