A Scalable and Highly Available SNS Service

1 Overview

The objective of this assignment is to develop the next version of the Tiny SNS service that is scalable to a large number of users with significantly more workload, is fault tolerant and highly available (i.e., failures in the system are handled transparently to the user. You are not required to use the provided client, but you are required to follow the provided gRPC interface. For this assignment the following must be considered:

1. You have 4 machines available. Your implementation can start at most 8 master/slave servers and 1 routing server. You should have at most 1 master and 1 slave server on each machine.
    a. One of the 4 masters is also Routing server for giving clients ip/port information of an available master.
    b. One of the master servers is an Available master. It is waiting for client's connection and provide SNS service.
    c. The two remaining masters are standing by and one of those will be elected as a new Available server if the current Available server crashes (it will take over the previously available master's job when crashed).

    An example deployment is (refer to Figure1)

    In Machine1 : 1 routing/master server and its slave are running.

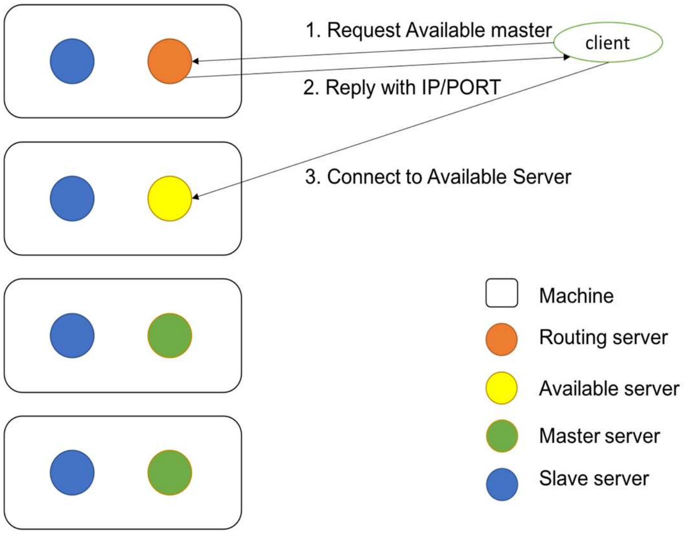
    In Machine2 : 1 available master server and its slave are running.

    In Machine3: a master server and its slave are running.

    In Machine4: a master server and its slave are running.

2. Your system should contain startup scripts that start your system.
3. You can assume that the machine on which the master process runs, is always available, although processes on it may still crash (i.e., killed by us).

4. The failure of ANY process except for Routing Server in the system is possible and you will need to take this into account.
5. We will "kill" at most 1 process within any 30 seconds time window. Thus, we will not attempt to crash a second process immediately after crashing a first process. We will wait 30 seconds. You can use this time interval to restart the crashed process (hint: slave can monitor whether master is crashed or not).
6. Data that has been stored persistently on hard disk will not be corrupted, i.e., you can rely on its accuracy.
    a. When an Available server crashes, the new elected Available server doesn't need to take over data stored previously by the Available server
7. Client should be always available for SNS service. That is, when an Available server crashes, the clients need to reconnect to new elected Available server. (Hint: client can ask the routing server a new available server)
8. The routing server will never crash in our scenario



<Figure 1. An example deployment>