**CMSC 621 AOS Project2**

**ASSIGNMENT 1 :**

In the first assignment we are implementing Berkeley’s algorithm for multiple processes each having their own logical clock in order to synchronize time of each of these individual clocks with the clock value of daemon process.

**1] REQUIREMENTS:**

We have implemented a n-node distributed system where our number of nodes that is number of clients are 3. Our Berkeley algorithm will be implemented on this distributed machine.

Here each node/thread has its own logical clock value generated arbitrarily. Each node should synchronize its time to a same clock value. Here our distributed system has been implemented on the same machine for simplification purposes where each running process is a machine having their individual clock values. We have implemented socket programming and have considered our time daemon to be the server process.

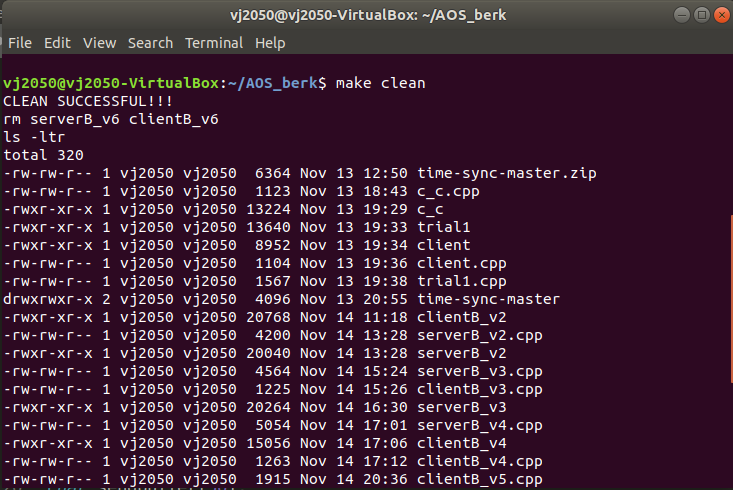
**2] LOGIC USED :**

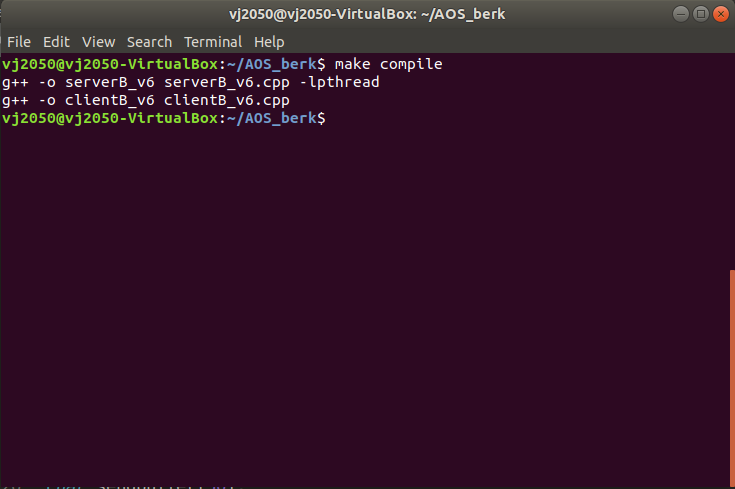
1. Spawn three node processes to be connected with the time daemon process.
2. Establish a connection between the time daemon process and the rest 3 nodes/threads.
3. Take a server clock value arbitrarily using srand() function.
4. When all the nodes/threads have been connected, then send the clock value of the time daemon process to each of the individual node processes.
5. At each node process, calculate the difference between the local clock value and the daemon clock value.
6. Send each of these differences back to the time daemon. Also the difference of time daemon with itself will always be 0.
7. Time daemon will calculate the average of these differences.
8. Time daemon calculates offset =average –each of the differences. Send these individual offsets to all the nodes.
9. Each process (including the time daemon at its own end) will then adjust its own local time by adding this offset received with its current clock value.
10. Each process will display its synchronized value which is same at at each end.
11. Exit.

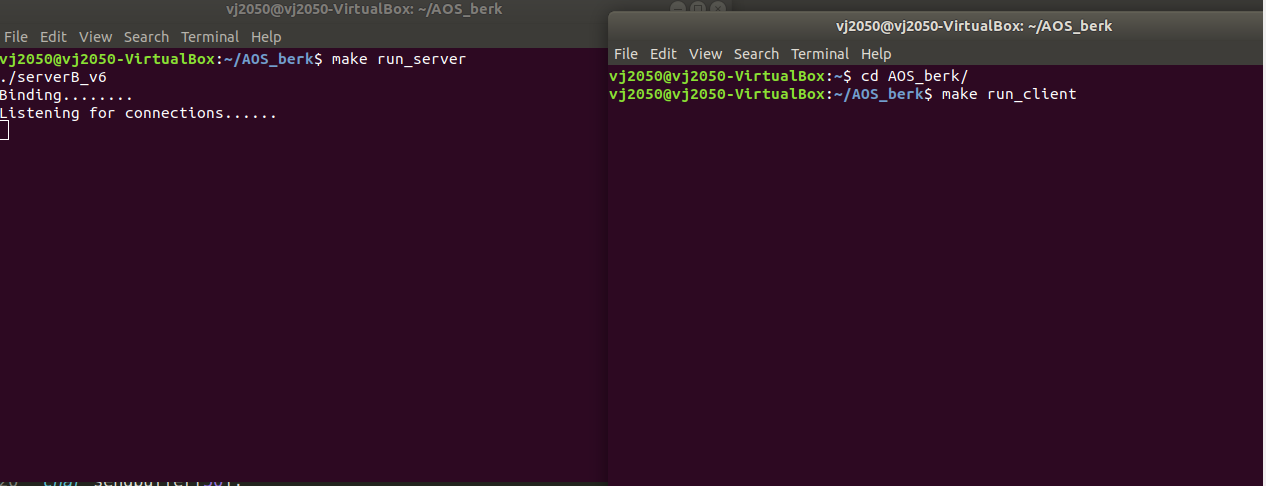
The 3 clients(node processes) will sent connection request to server(time daemon process) . After the connections have been established, server(time daemon) sends its own clock value to clients. The clients will calculate differences at their ends and respond to server with their differences. Server will calculate average and offsets and then will send individual offset clock values relative to the calculated average to each of the clients. The clients and server will synchronize their own time and will display updated logical clock values which will all be the same.

**Please Note :** The port numbers for each of the processes are same. The number of clients have been taken as 3 here.

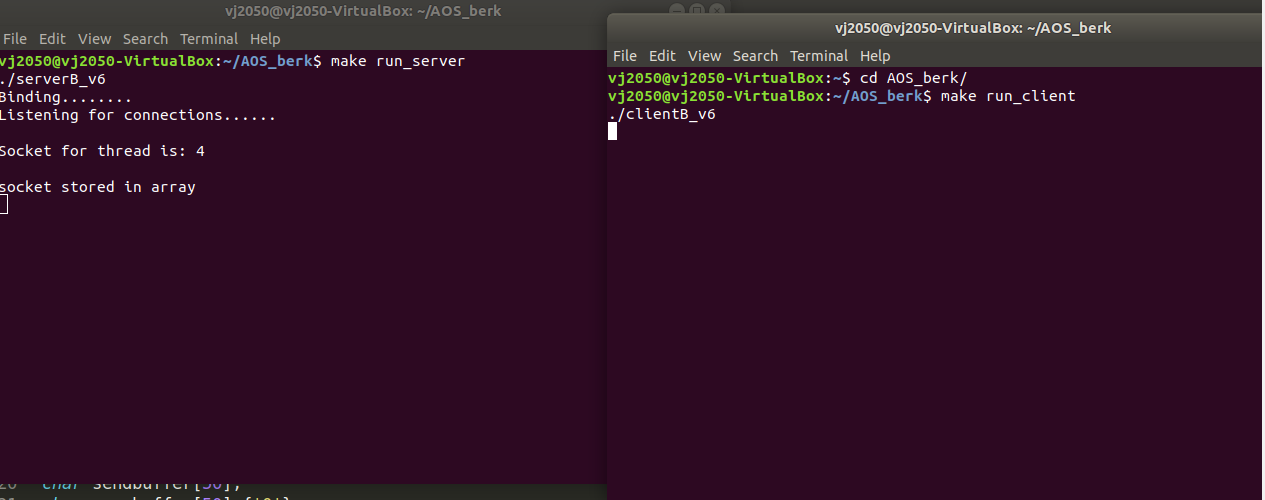
**OUTPUT:**



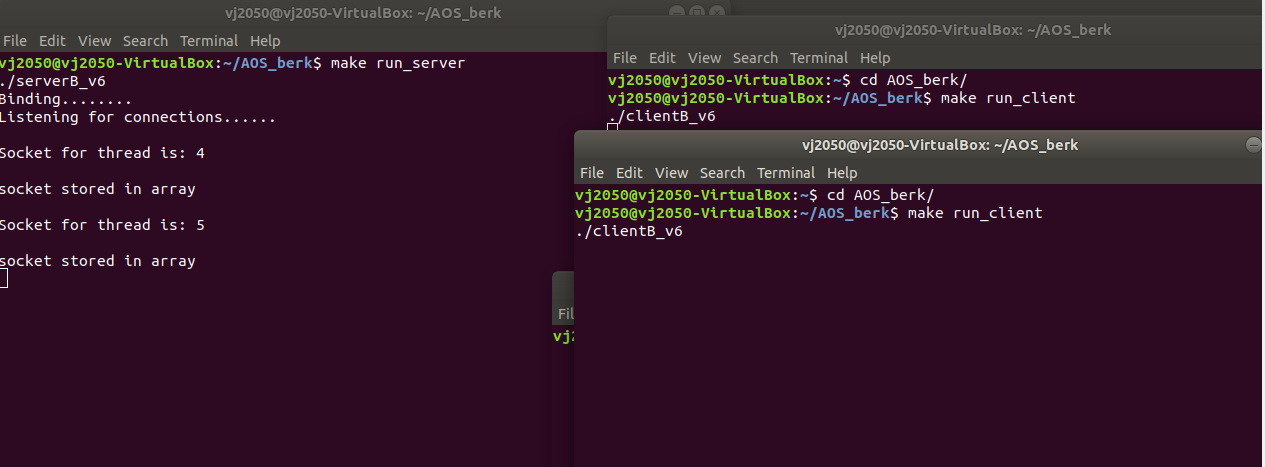




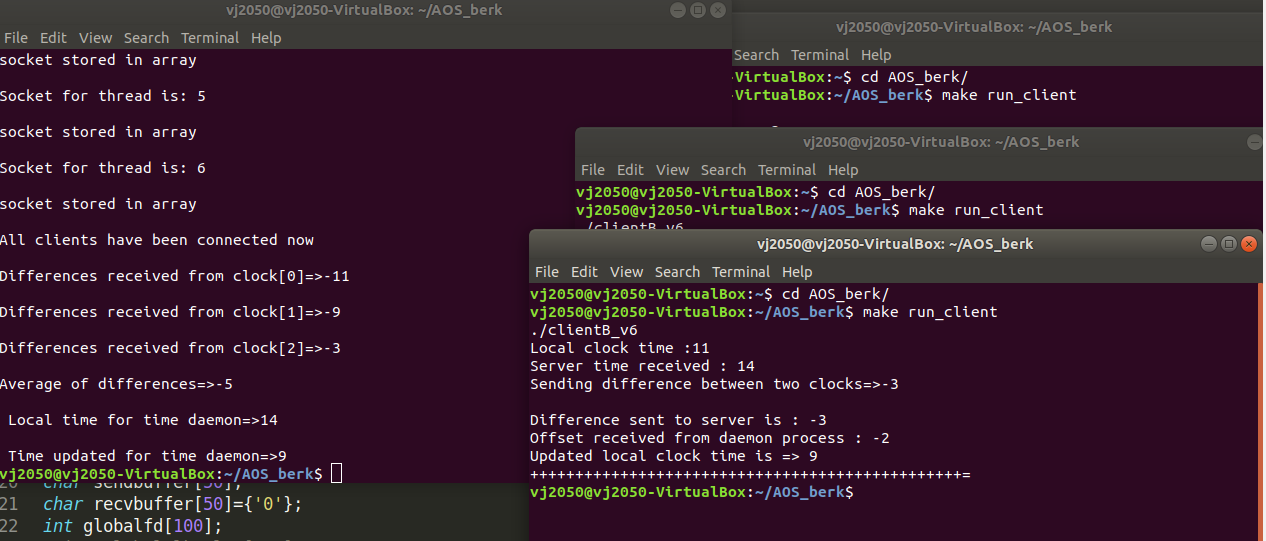
First client spawned:



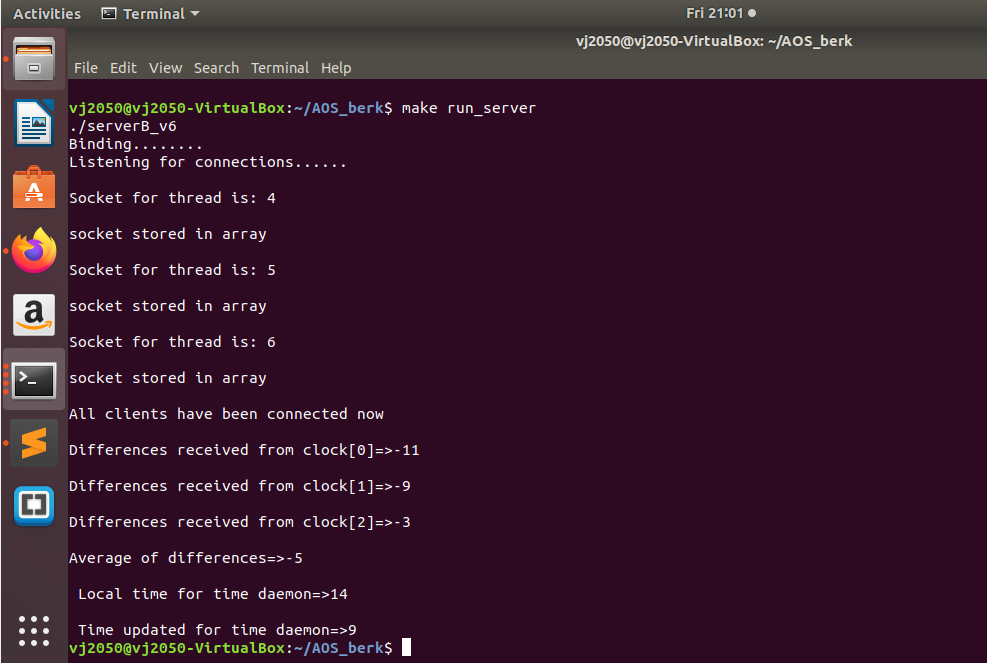
Second client spawned :



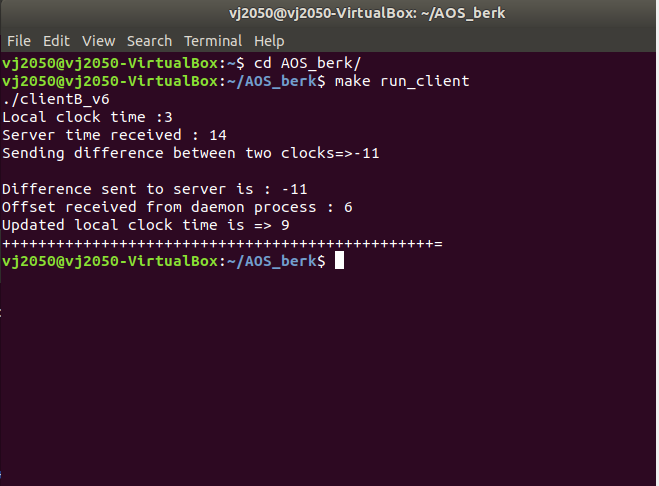
Third thread spawned:



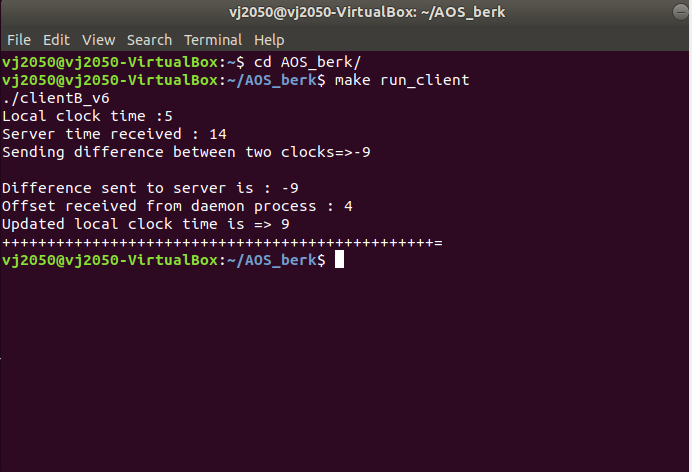
Time Daemon’s (server) synchronized time :



client 1’s updated time :



2nd client’s updated time :



3rd client’s updated time :

