**Experiment 6–Clock Synchronization**

**Learning Objective:** Student should be able to develop a program for Clock Synchronization.

**Tools :**Java

**Theory:**

Clock synchronization algorithms may be broadly classified as centralized and distributed.

1. **Centralized Algorithms**

In centralized clock synchronization algorithms one node has a real-time receiver. This node is usually called the time server node, and the clock time of this node is regarded as correct and used as the reference time. The goal of the algorithm is to keep the clocks of all other nodes synchronized with the clock time of the time server node. Depending on the role of the time server node, centralized clock synchronization algorithms are again of two types-passive time server and active time server.

**Passive Time Server Centralized Algorithm**

In this method, each node periodically (with the interval between two periods being less than or equal to δ/2p) sends a message ("time = ?") to the time server. When the time server receives the message, it quickly responds with a message ("time = T"), where T is the current time in the clock of the time server node. Let us assume that when the client node sends the "time=?" message, its clock time is To, and when it receives the "time = T" message, its clock time is T). Since To and T) are measured using the same clock, in the absence of any other information, the best estimate of the time required for the propagation of the message "time = T" from the time server node to the client's node is (T 1 - To)/2. Therefore, when the reply is received at the client's node, its clock is readjusted to' T+(T1 - To)/2.

**Active Time Server Centralized Algorithm**

In the passive time server approach, the time server only responds to requests for time from other nodes. On the other hand, in the active time server approach, the time server periodically broadcasts its clock time ("time = T"). The other nodes receive the broadcast message and use the clock time in the message for correcting their own clocks. Each node has a priori knowledge of the approximate time (To) required for the propagation of the message "time =T" from the time server node to its own node. Therefore, when the broadcast message is received at a node, the node's clock is readjusted to the time T+To. A major drawback of this method is that it is not fault tolerant. If the broadcast message reaches too late at a node due to some communication fault, the clock of that node will be readjusted to an incorrect value. Another drawback of this approach is that it requires broadcast facility to be supported by the

network.

Another active time server algorithm that overcomes the drawbacks of the above algorithm is the Berkeley algorithm. It was proposed by Gusella and Zatti [1989] for internal synchronization of clocks of a group of computers running the Berkeley UNIX. In this algorithm, the time server periodically sends a message ("time = ?") to all the computers in

the group. On receiving this message, each computer sends back its clock value to the time

server.

The time server has a priori knowledge of the approximate time required for the propagation

of a message from each node to its own node. Based on this knowledge, it first readjusts the clock values of the reply messages. It then takes a fault-tolerant average of the clock values of all the computers (including its own). To take the fault-tolerant average, the time server chooses a subset of all clock values that do not differ from one another by more than a specified amount, and the average is taken only for the clock values in this subset. This approach eliminates readings from unreliable clocks whose clock values could have a significant adverse effect if an ordinary average was taken.

Centralized clock synchronization algorithms suffer from two major drawbacks:

I. They are subject to single-point failure. If the time server node fails, the clock synchronization operation cannot be performed. This makes the system unreliable. Ideally, a distributed system should be more reliable than its individual nodes. If one goes down, the rest should continue to function correctly.

2. From a scalability point of view it is generally not acceptable to get all the time requests serviced by a single time server. In a large system, such a solution puts a heavy burden on that one process.

Distributed algorithms overcome these drawbacks.

1. **Distributed Algorithms**

Recall that externally synchronized clocks are also internally synchronized. That is, if each node's clock is independently synchronized with real time, all the clocks of the system remain mutually synchronized. Therefore, a simple method for clock synchronization may be to equip each node of the system with a real-time receiver so that each node's clock can be independently synchronized with real time. Multiple real-time clocks (one for each node) are normally used for this purpose.

Theoretically, internal synchronization of clocks is not required in this approach. However, in practice, due to the inherent inaccuracy of real-time clocks, different real time clocks produce different time. Therefore, internal synchronization is normally performed for better accuracy.

**Exercise:**

1. **Explain Cristian's Probabilistic Algorithm?**
2. **Explain Berkeley’s Algorithm clock synchronization technique?**

**Result and Discussion:** .…………………………………………………………………………………………………

…………………………………………………………………………………………………..

**Learning Outcomes:** The student should have the ability to

LO1: Describe Clock Synchronization algorithm

LO2: Write a Program to demonstrate Clock Synchronization Algorithm

**Course Outcomes:** Upon completion of the course students will be able to understand Clock Synchronization Algorithm.

**Conclusion:**……………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………...

**Viva Questions:**

1. [What is the difference between 'clock synchronization' and 'time synchronization'?](https://networkengineering.stackexchange.com/questions/37979/what-is-the-difference-between-clock-synchronization-and-time-synchronization)?
2. Explain Logical Clock?
3. Explain Physical Clock?

**For Faculty Use**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Correction Parameters** | **Formative Assessment [40%]** | **Timely completion of Practical [ 40%]** | **Attendance / Learning Attitude [20%]** |  |
| **Marks Obtained** |  |  |  |