Experiment 06: Develop a program for Clock Synchronization algorithms

Learning Objective: Learn and implement Clock Synchronization algorithms in Java to maintain synchronized time across multiple clocks.

Aim: To implement a program that demonstrates clock synchronization algorithms, ensuring consistency of time across distributed systems.

Tools: Java Development Kit (JDK), IDE (e.g., IntelliJ, Eclipse), Terminal.

Theory:

Principles of Clock Synchronization:

Clock synchronization ensures that all nodes in a distributed system maintain consistent time values. This is critical in distributed computing, where accurate timekeeping is essential for coordination, ordering events, and maintaining consistency.

The Protocol:

1. **Initialization:**
   - Multiple clocks with different times are considered.
   - A method is chosen to synchronize them.
2. **Clock Time Collection:**
   - Each clock reports its current time.
   - The times are collected and compared.
3. **Synchronization Algorithm:**
   - Calculate the average time of all clocks.
   - Adjust each clock to the computed average time.

Security Considerations:

- Ensure reliable time sources to prevent attacks.
- Use authenticated time synchronization protocols.
- Minimize clock drift to avoid inconsistencies.

Properties of Clock Synchronization Algorithm:

1. **Precision:** Ensures clocks maintain a minimal deviation from actual time.
2. **Accuracy:** Adjusts time to match a reliable time source.
3. **Fault Tolerance:** Can handle minor errors and drifts in clocks.
4. **Scalability:** Can be applied to large distributed systems.
5. **Efficiency:** Adjusts time with minimal computational overhead.

Implementation:

Java Code :-

```
import java.util.Scanner;

class ClockSynchronization {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of clocks: ");
        int numClocks = scanner.nextInt();
        int[] clocks = new int[numClocks];

        System.out.println("Enter the time (in seconds) for each clock:");
        for (int i = 0; i < numClocks; i++) {
            System.out.print("Clock " + (i + 1) + ": ");
            clocks[i] = scanner.nextInt();
        }

        int sum = 0;
        for (int time : clocks) {
            sum += time;
        }

        int avgTime = sum / numClocks;

        System.out.println("Synchronized Clock Time: " + avgTime + " seconds");

        scanner.close();
    }
}
```

Output:

```
Enter the number of clocks: 3
Enter the time (in seconds) for each clock:
Clock 1: 20
Clock 2: 30
Clock 3: 10
Synchronized Clock Time: 20 seconds

=== Code Execution Successful ===
```


Learning Outcomes: The student should have the ability to:

LO1.1 Explain the fundamental principles of clock synchronization.
LO1.2 Identify and describe the mechanisms that ensure synchronization accuracy.
LO1.3 Implement a basic clock synchronization algorithm in Java.
LO1.4 Analyze potential risks and ethical considerations.

Conclusion:

Viva Questions:

1. What is the need for clock synchronization in distributed systems?
2. How does clock drift affect system consistency?
3. What methods can be used to synchronize clocks in a network?
4. How can network latency impact clock synchronization?

For Faculty Use:

Correction Parameter s	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				