



# LAMPORT'S ALGORITHM FOR DISTRIBUTED FILE SYSTEM

# Introduction

- Lamport's Algorithm is a distributed mutual exclusion algorithm.
- It ensures only one node accesses a shared resource at a time.
- Widely used in Distributed File Systems.

# Problem Statement

- Multiple nodes may try to access the same file simultaneously.
- This can cause data inconsistency and race conditions.
- A mechanism is required to ensure mutual exclusion.

# Why Lamport's Algorithm

- No centralized coordinator
- Uses logical clocks
- Ensures fairness and deadlock-free access

# System Requirements

- Nodes in a distributed environment
- Reliable message passing
- Lamport logical clocks
- Request queue

# Message Types

- REQUEST – To ask for file access
- REPLY – Permission granted
- RELEASE – File access completed

# Algorithm

- Node  $P_i$  wants to access a file
- Increment logical clock
- Broadcast **REQUEST** to all other nodes
- Add request to local priority queue
- Receive **REPLY** from all nodes
- Enter **Critical Section (File Access)**
- After completion, broadcast **RELEASE**
- Remove request from queue

# Critical Section

- Represents file access in DFS
- Only one node allowed at a time
- Ensures data consistency



# Sample Execution

- P1 requests file
- P2 requests file
- P1 enters critical section
- P2 waits until release

# Advantages

- Ensures mutual exclusion
- Fair ordering of requests
- No deadlock or starvation

# Limitations

- High message overhead
- Not suitable for very large systems

# Applications

- Distributed File Systems
- Distributed Databases
- Cloud Storage Systems

# Conclusion

- Lamport's Algorithm ensures safe file access
- Uses logical clocks and message passing
- Effective for distributed environments