# Objective

The objective of this project is to implement Lamport’s mutual exclusion algorithm on 10 distributed nodes and conclude the performance by finding out number of protocol messaged required by one node per critical section. To further extend, it is also required to see the impact of the optimization over communication overhead.

# Implementation

## Environment Details

1. Language chosen is JAVA.
2. 10 nodes are chosen in a way, that Node 0 is on net30.utdallas.edu, Node 1 is on net31.utdallas.edu and so on till Node 9 which is on net39.utdallas.edu.
3. The common shared log fill is on net40.utdallas.edu running a server. Each node when enters its critical section sends a message to the logging server to log its timestamps of entering and leaving the critical section.

## Node Details

1. Each node runs three threads simultaneously:
   1. Sender: This thread is responsible to send message to other node. It pops up the message from sending queue and sends it to the respective destination.
   2. Listener: This thread runs the TCP socket responsible to receive any incoming message and process it.
   3. Execution: This thread runs till the node has entered into its critical section 20 times. It does all the work of the node of generating request messages, application messages, release messages and entering into critical section.

## How to Run

### Running the logging server

1. Log into net40.utdallas.edu and run

$bash: java –jar LoggingServer.jar

### Running a distributed node

1. Since the node id depending on the server you login so the generic command is as follows - Log on into net3<Node id>.utdallas.edu and run

$bash: java –jar LamportMutex.jar <Node id> <total nodes>

For example: if total nodes present are 10 and to run Node 5, log on into net35.utdallas.edu and run

$bash: java –jar LamportMutex.jar 5 10

# Conclusion

The output shows the total number of protocol message exchanged for 20 critical section by which we can find out total protocol message exchanges per critical section.

1. For first variation (without optimization) it was 3(N-1) where N is the number of nodes. For our simulation for 10 nodes each node had to exchange 27 protocol messages per critical section when tried to enter critical section for 20 times.
2. For second variation (with optimization) the number of protocol message declined to somewhere between 2(N-1) to 3(N-1) per critical section for N nodes. This optimization reduces the communication overhead to achieve mutual exclusion.

# 