Distributed Operating System Sample Questions for Mid Semester (Mutual Exclusion)

Short Questions

1. **What are the potential drawbacks of using a centralized coordinator for mutual exclusion?**

Ans: The potential drawbacks of using centralized coordinator for mutual exclusion are:

i.The coordinator is a single point of failure, so if it crashes, the entire system may go down.

ii. It cannot distinguish between a dead coordinator from permission denied.

iii. It cannot handle a large volume of requests coming simultaneously and leads to performance bottlenecks.

2. **What is the consequence of a process holding the token but not needing the critical section?**

Ans: The token is simply passed Using a token passing mechanism, where a process holding the token but not needing the critical section passes the token to the next process in the ring. This ensures that the token circulates and eventually reaches processes that require access to the critical section.

Long Questions

1. **A distributed system may have multiple, independent critical regions. Imagine that process 0 wants to enter critical region A and process 1 wants to enter critical region B. Can distributed algorithms lead to deadlocks? Explain**

Ans: Yes, a distributed algorithm can lead to deadlocks even if processes are trying to enter different critical regions. Here's how deadlocks can occur:

i. Request overlap:
Both processes might be requesting access to their respective critical regions at the same time. As the distributed mutual exclusion algorithm involves requesting and granting access via message passing, there could be a situation where both processes are waiting for a resource or acknowledgment that leads to a deadlock.

ii. Resource dependency:
If the algorithm involves acquiring resources that are required for accessing critical regions, a situation can arise where process 0 holds a resource needed by process 1, and process 1 holds

a resource needed by process 0. This interdependency can lead to a deadlock if neither process can proceed without the resource held by the other.

iii. Inconsistent State: In distributed systems, maintaining a consistent state across processes is challenging. If the algorithm does not correctly manage the state transitions (e.g., process states) inconsistencies might arise, leading to scenarios where processes end up waiting indefinitely.

2. **Suppose in a centralized approach to mutual exclusion in a distributed system, the coordinator crashes. Does this always bring the system down? If not, under what circumstances does this happen?**

Ans: In a centralized approach to mutual exclusion in a distributed system, a single coordinator is responsible for granting access to the critical section. If this coordinator crashes, it doesn't necessarily bring the entire system down, but it does lead to significant issues.

Impact of Coordinator crash:

i. Blocking of requests: If the coordinator crashes, any process that requests access to the critical section will be blocked indefinitely because there is no one to grant or deny access. This could lead to a system halt as processes wait indefinitely.

ii. System Down: The system might effectively be down in terms of its ability to perform critical operations that require mutual exclusion. However, other parts of the system might continue functioning if they don't require mutual exclusion.

Some of the circumstances where the system doesn’t go down:

i. No pending requests: If there are no current requests for the critical section when the coordinator crashes, the system may continue operating normally until a process requests access.

ii. Redundancy or Backup Coordinator: If the system has mechanisms to detect the crash and elect a new coordinator or has a backup coordinator ready to take over, the impact can be minimized.