Operating Systems – II Review Questions

Week 3 - Answer all the Questions

Remote Procedure Calls

1. Evaluate the effectiveness of RPCs compared to message-passing systems for implementing distributed applications. Justify your answer based on performance, transparency, and error handling.
2. Given two implementations of an RPC system—one using static binding and the other dynamic binding—which would you recommend for a highly scalable cloud-based service, and why?
3. Critically assess the trade-offs between exactly-once and at-least-once invocation semantics in RPC. Which would you recommend for financial transaction systems, and why?
4. An organization is choosing between implementing gRPC and a custom lightweight RPC framework. As a system analyst, evaluate the two options based on extensibility, maintainability, and support for multiple languages.
5. Judge the suitability of RPC in a real-time system (e.g., embedded control system in aviation). What are the critical limitations, and would you recommend using RPC in such a scenario?
6. You are tasked with designing a distributed file storage system. Evaluate whether RPC or RESTful APIs would be more appropriate for client-server communication. Justify your choice.
7. Evaluate the fault tolerance mechanisms typically used in RPC systems. Are they sufficient for mission-critical applications? What would you recommend improving?
8. Assess the impact of asynchronous RPC on system responsiveness and resource utilization in a microservices architecture. Would you recommend asynchronous over synchronous RPC in such contexts?
9. A university student project team wants to use RPC over HTTP to develop a distributed voting system. Critique their approach and suggest whether this is advisable. Support your judgment with reasoning.
10. Compare and evaluate the use of middleware frameworks (like CORBA, Java RMI, or gRPC) for implementing RPC in a distributed e-commerce platform. Which framework would you recommend, and on what basis?

Distributed Processing

1. Evaluate the design principles of a distributed operating system (DOS). Which principle (e.g., transparency, fault tolerance, scalability) do you consider most critical for system performance, and why?
2. Given a choice between a centralized system and a distributed operating system for a smart city infrastructure project, which would you recommend? Justify your recommendation based on system requirements such as fault tolerance, scalability, and responsiveness.
3. Critique the rationale for adopting distributed systems in large-scale enterprises. Are the benefits (e.g., resource sharing, modular growth) always worth the increased complexity and overhead?

4. Assess the effectiveness of location transparency in distributed operating systems. When could this feature become a liability rather than an advantage?
5. You are tasked with building a distributed application across a heterogeneous network of devices. Evaluate how the principles of distributed OS design (such as transparency and concurrency) help or hinder your objective.
6. Judge the appropriateness of using a distributed operating system to manage resources in a university campus network. What limitations or risks should be considered?
7. Evaluate the role of fault tolerance and recovery mechanisms in a distributed OS versus a traditional centralized OS. Are the mechanisms in distributed OSs sufficient for mission-critical systems?
8. Compare and assess two architectures for distributed systems: peer-to-peer (P2P) and client-server. Which architecture better supports the principles of a distributed OS and under what conditions?
9. A startup is considering implementing a distributed system for its logistics operations. Evaluate whether this decision is suitable at their current scale and justify what conditions must be met for distributed processing to be beneficial.
10. Assess the trade-offs between performance and transparency in distributed operating systems. Should system designers prioritize one over the other? Defend your position with examples.