

## Operating Systems Design Fall 2010 Exam 3 Review

Paul Krzyzanowski  
pxk@cs.rutgers.edu

### 1. How does the functionality of a client stub differ from a server stub in RPC?

The purpose of a client stub is to provide a functional interface that matches the remote procedure. Internally, its task is to marshal parameters into a message, send the message to the server stub, and get the return.

The server stub (aka skeleton) is the listener and front-end to the implementation of the remote procedures. It registers the remote procedures, receives messages from clients, calls the user's functions, and sends back return data.

### 2. How does a callback promise achieve cache coherence in AFS?

Whenever a client downloads a file from an AFS server, the server maintains state: it keeps a list of clients that have the file downloaded.

As long as the file is not modified at the server, the client can use its copy without worrying that it is out of date.

When any client changes its cached copy of the file, the updated file is sent to the server when the file is closed and the server sends a callback message to each client that has a cached copy of the file, telling it to invalidate the copy. The next time the client opens the file, it will have to download the latest version from the server.

### 3. What is a *nonce* and how would Alice use it to authenticate Bob with public key cryptography?

A nonce is just a random bunch of bits – a big random number.

Alice can use it to authenticate Bob by asking Bob to prove he can do something that only Bob can do: transform the nonce with his private key. There are two options:

- (1) Alice sends a nonce to Bob and asks him to encrypt it with his private key. If she can decrypt the message with his public key to reveal the nonce, she's convinced it's Bob.
- (2) Alice encrypts a nonce with Bob's public key and asks him to decrypt it with his private key. If he sends back the nonce then she's convinced that Bob had the private key and it's Bob.

### Questions 4-6

4. A virtual circuit is a:
  - a) Connection-oriented protocol. \*\*
  - b) Connectionless protocol.
  - c) Circuit-switched service.
  - d) Datagram service.
4. Which layer manages the communication of data from one application to another?
  - a) Network Layer
  - b) Transport Layer \*\*
  - c) Presentation Layer
  - d) Data Layer
4. Under IP, port numbers are found in this layer of the OSI reference model:
  - a) Data link layer.
  - b) Network layer.
  - c) Physical layer.
  - d) Transport layer. \*\*

### Questions 7-8

7. In the OSI Reference Model, what is the purpose of the Transport layer?
  - a) Conversion of data into a machine-independent representation.
  - b) Routing packets from one machine to another, possibly through routers.
  - c) Providing end-to-end communication services for applications. \*\*
  - d) Transferring packets from one machine to another within a physical local area network.
7. Which is true about the Transport Control Protocol (TCP)?
  - a) It provides datagram (connectionless) service.
  - b) It sends an acknowledgment for each packet received. \*\*
  - c) It permits incoming data to arrive out of sequence.
  - d) It was designed for Ethernet networks.

## Questions 9-10

9. Why was the development of Classless Inter-Domain Routing (CIDR) necessary?
- CIDR expands the size of an IP address from 32 bits to 128 bits, thus increasing the number of assignable addresses.
  - Network Address Translation (NAT) proved too ineffective for dealing with the limited number of assignable IP addresses.
  - The original method of assigning Class A, B, or C IP address blocks proved too restrictive. \*\*
  - Network routers needed a mechanism to efficiently manage rapidly growing routing tables.

*The class-based system allocated 8-bit, 16-bit, or 24-bit blocks of addresses. If you needed to support 300 hosts, a class C (8-bit) block was not sufficient so you would have to get a class B (16-bit) block. This meant that over 65,000 addresses were never used.*

10. What is the purpose of the Address Resolution Protocol (ARP)?

- To discover the MAC address that corresponds to a given IP address. \*\*
- To discover the IP address that corresponds to a specific service.
- To discover the network on which an ethernet host resides.
- To discover the IP address that corresponds to a given ethernet MAC address.

## Questions 11-12

11. The IP driver is responsible for all of the following EXCEPT:

- Polling network routers with network service requests. \*\*
- Dropping data with bad checksums in the IP header.
- Receiving data from the device driver.
- Routing a packet from one physical network to another.

*Polling routers does not make sense. A router transmits packets onto the network; other nodes listen for incoming packets (and vice versa).*

12. Which socket system call is not needed for a server process:

- socket
- connect \*\*
- accept
- listen

*A connect call is used by a client to connect to a server.*

## Questions 13-14

13. What part of the Linux networking stack contains a common set of functions for low-level network drivers to interface with the higher-level protocol stack?

- Generic Network Interface (sockets layer)
- Network Protocols (`proto`, `proto_ops`)
- Abstract Device Interface (`net_device`) \*\*
- Network Device Drivers (`dev`)

*The abstract device interface defines a common interface for interacting with network devices.*

14. Which component cannot be loaded as modules?

- Network protocol
- Network device driver
- File system
- Abstract device interface \*\*

*The abstract device interface isn't a driver. All others are and can be dynamically loaded and can register themselves as a network protocol, device driver (block, char, or network), or a file system type.*

## Questions 15-16

15. Which statement is FALSE? The socket buffer (`sk_buff`):

- Avoids the need to copy packet data from one protocol layer to another.
- Is allocated for user data sent on a socket.
- Is allocated for packets received by the network interface
- Is a pool of memory from which sockets are allocated. \*\*

*A socket buffer represents a packet of data that stays in one place as different layers of the network process it.*

16. Differing from previous approaches, the Linux NAPI (New API) packet processing framework:

- Disables network device interrupts after receiving a packet and then relies on polling until there are no more packets to process. \*\*
- Enables network device interrupts to ensure that the kernel can respond to new packets immediately.
- Provides a partitioning between the network device driver and protocol processing logic.
- Allows devices to be modular and added or removed dynamically.

## Questions 17-18

17. An interface definition language is:

- Processed by the RPC precompiler to generate stub functions. \*\*
- A machine-independent output language generated by the RPC precompiler.
- Used to implement remote procedures.
- Used by the operating system to provide user processes with an interface to remote procedure calls.

*The Interface Definition Language describes the remote functions (names, input, output) so that the RPC compiler can create both client-side and server-side stub functions to marshal & unmarshal parameters.*

18. Which file system was designed to be stateless?

- NFS \*\*
- CIFS
- AFS
- SMB

*Even though the latest version of NFS is stateful, the file system was originally designed to be stateless.*

## Questions 19-20

19. Which of the following was not a remote operation in the original versions of NFS?

- READ (read bytes from a remote file)
- SYMLINK (create a symbolic link file)
- REMOVE (remove a remote file)
- CLOSE (close a remote file) \*\*

*NFS was designed to be stateless. A close procedure would serve no purpose since there is nothing to close on the server (nothing was open; no state was kept).*

20. The principle of least privilege means:

- A system should have at least two types of accounts: administrative as well as normal user accounts.
- Do not allow a process to communicate with a process running at a higher privilege level.
- Do not allow a process to create files that could be accessed by anyone with a lower privilege level.
- Never give a user more permissions than he or she needs. \*\*

*(c) And (d) relate to Mandatory Access Control and the Bell-LaPadula model. The principle of least privilege is the concept of giving a process the bare minimum access rights that are necessary for the process to do its job.*

## Questions 21-22

21. An access control list (ACL) is:

- a) Associated with an object and identifies access permissions for various domains in the system. \*\*
- b) Associated with a domain and identifies access permissions for various objects in the system.
- c) A structure that allows one to look up access permissions given a domain and object.
- d) A list of users that are allowed access onto the system

*An access control list is typically stored as a list of permissions in the file's metadata.*

22. Why did earlier versions of Linux, BSD, and Unix not implement ACLs?

- a) They don't fit in an inode. \*\*
- b) They are redundant with protection mechanisms already in place.
- c) They opted to use capability lists instead.
- d) The operating system only supports discretionary access control

*An access control list is a variable size structure (you might itemize permissions for lots of users). As such, it's not feasible to fit it into a fixed-length inode. Instead, Unix systems opted for a restricted form of an ACL: enumerating permissions only for the owner of the file, the group of the file, and anyone else.*

## Questions 23-24

23. The Bell-LaPadula model is NOT an example of:

- a) Mandatory access control.
- b) Multi-level secure access control.
- c) Discretionary access control. \*\*
- d) Hierarchical sensitivity levels.

*Discretionary access control allows a user to make decisions on who can access files owned by the user.*

24. Which mechanism cannot be provided by the operating system?

- a) Address Space Layout Randomization
- b) Non-executable stacks
- c) Stack canaries \*\*
- d) All of the above

*Stack canaries are extra code generated by the compiler to add a number to the stack at the start of the function and, at every return from the function, validate that the number has not been modified.*

## Questions 25-26

25. For Alice to send a secret message to Bob:

- a) Alice encrypts the message with her private key.
- b) Alice encrypts the message with her public key.
- c) Alice encrypts the message with Bob's private key.
- d) Alice encrypts the message with Bob's public key. \*\*

*(a) No. Anyone can decrypt with Alice's public key; (b) No. Nobody but Alice would be able to decrypt; (c) No. Alice doesn't have Bob's private key; (d) Yes. Only Bob will be able to decrypt since you need to have Bob's private key to do so.*

26. Cryptographic hash functions take an input string S and output a hash value H. Which statement is true?

- a) It is difficult to compute H for every S.
- b) Given H and the original hash function, it is feasible to find S.
- c) Two different values of H may be computed from one value of S and the same hash function.
- d) It is not feasible to modify S to hash to the same value H. \*\*

*(a) Hash functions should be efficient; (b) Hashes are fixed-length values, typically much smaller than the original message. Finding a message that hashes to S is not feasible; (c) If you apply the same hash function to the same bunch of bits, you expect to get the same output; (d) Correct. A good hash function serves as a message authentication code. There should not be any easy way to modify a message and have the hash of that message have the same value as with the unmodified message.*

## Questions 27-28

27. The Diffie-Hellman algorithm:

- Allows two entities to negotiate a common key. \*\*
- Is an example of a public key encryption algorithm.
- Allows all participating entities to securely exchange public keys for direct communication.
- Requires entities to register their private key with a central secure database.

*(a) No. Anyone can decrypt with Alice's public key; (b) No. Nobody but Alice would be able to decrypt; (c) No. Alice doesn't have Bob's private key; (d) Yes. Only Bob will be able to decrypt since you need to have Bob's private key to do so.*

28. Cryptographic hash functions take an input string S and output a hash value H. Which statement is true?

- a) It is difficult to compute H for every S.
- b) Given H and the original hash function, it is feasible to find S.
- c) Two different values of H may be computed from one value of S and the same hash function.
- d) It is not feasible to modify S to hash to the same value H. \*\*

*(a) Hash functions should be efficient; (b) Hashes are fixed-length values, typically much smaller than the original message. Finding a message that hashes to S is not feasible; (c) If you apply the same hash function to the same bunch of bits, you expect to get the same output; (d) Correct. A good hash function serves as a message authentication code. There should not be any easy way to modify a message and have the hash of that message have the same value as with the unmodified message.*