CS447-Network and Data Communication
Possible Quiz Questions for
Quiz #3 on January 31st, 2024

The following is a list of possible questions for our Quiz #3 on January 31st. Some of the questions will not be asked in the quiz. All the questions that will appear in the quiz will appear exactly as shown below (however, parameters may be changed). The quiz is closed textbook, closed notes and closed neighbors. Note that the questions, which did not appear in this quiz, still may appear in the exams.

#1: What is "error control" for packet-switching networks?

#2: Show the structure (the flowchart) of error control for packet-switching networks.

#3: What are "undetectable errors"?

#4: Demonstrate how "undetectable errors" happen using parity-error detection.

#5: What are "uncorrectable errors"?

#6: Can "undetectable errors" be "correctable"?

#7: How does "sliding-window flow control" work?

#8: What is the primary advantage of "sliding-window flow-control" over "stop-and-wait flow-control"?

#9: How can we calculate the link utilization for the sliding-window flow control (as mathematical formula)?

********** the topics covered on January 29th **********

#10: What are "blocking functions"? Mention three socket APIs that are "blocking function".

#11: What are "non-blocking functions"? Mention three socket APIs that are "non-blocking function".

#12: What is "client and server model"? Mention three (real) network applications that are based on "client and server model" (this particular question is for your homework).

#13: What is the counter concept of "client and server model?

#14: As we discussed in the classroom, it is possible for us to have more than one socket connections between a server and a client host. If each of the multiple socket connections is for a different client-side network process, how the client-side host distinguishes each socket connection to each network application process at the client side?

#15: What is the information that is used for uniquely identify each host computer in a network (e.g., the Internet)? What is the information that is used for uniquely identify each network application process (program) at a host computer connected to a network?

#16: What is "bind" socket API for (explain the purpose of the API)?

#17: It is possible for a client-side network application process not to use "bind" (it can be skipped at the client side network process). What will happen, if "bind" is skipped at the client side network process?

#18: Although it is possible to skip "bind" at the client-side network process, bind can not be skipped at the server-side network process. Why (why bind can not be skipped at the server side)?

#19: What is "bind" socket API for (explain the purpose of the API)? What will be performed by "close", if it is performed at the server side? What will be performed by "close", if it is performed at the client side?

#20: Show the general socket API structure for network applications that are based on client and server mode. Show the messages exchanged between a server and a client.

#21: If a file (or payload) that is transferred from a server to a client is large (i.e., larger than the amount of the payload transmitted by one call to "send"), how will it be performed?

#22: "accept" socket API duplicates a socket connection through another port at the server side as soon as a connection request from a client is established. Why (explain the purpose of doing it)?

#23: Some of the ports are "reserved". Why?

#24: HTTP is a stateless application-layer protocol. What does "stateless" mean?

#25: Show the message pattern expected for downloading a webpage using HTTP (assume that most of the websites consist of multiple other "component files".

#26: What is the difference between "fierwalls" and "(security) proxies"?

#27: Which socket API uses "sockaddr_in" structure? What is that data structure for (describe its primary purpose)?

#28: Explain why multiple threads are needed at a proxy for handling HTTP network traffic?

#29: Show the message structure of HTTP responses.