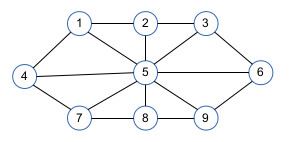
CSE 461 14wi Final Exam

Page **1** of 1

This exam will close at**5:00 PM on Monday, 3/17.**  
  
You can consult any existing resources that you want, but you cannot pose questions to other people (except the 461 instructional staff).  
  
Don't simply cut-and-paste material that you find.  (For the most part, that wouldn't be an appropriate answer in any case.)  
  
If you have a question about the exam, please send it by email to the entire instructional staff.  All exam-related questions and answers will be posted in a [read-only forum area](https://catalyst.uw.edu/gopost/area/zahorjan/127987).

**Question 1.**

The first three questions make use of this graph:  
  
  
  
The nodes represent IP routers.  We assume all routers have just powered on, and so know only about themselves and the states of the links connected to them.  We further assume that messages are sent sequentially, in order of the router numbers.  A message "is sent" once it has been transmitted and all intended recipients have seen and processed it.  So, router 1 sends the first message, and when all recipients have seen and processed it, router 2 sends the next message.   We start over at router 1 whenever router 9 has sent a message.  
  
To keep things simple, we use router numbers to represent network destinations -- all IP networks connected to router 3 are described as destination 3, for instance.  
  
Assuming that routing table maintenance is done using a **distance vector** scheme, how many messages must have been sent before every node knows how to reach every destination?  Briefly explain.

By the wording of the question, I assume that the router only send the message to the routers which have link to it but not exchange message with them. So router 1 sends a message to router 2, 4, 5 and those routers update their tables, but router 1 doesn’t update its table until it receives a message. My answers for question 1 and 2 are based on this assumption.

14 messages are required to be sent before every node knows how to reach every destination. After the first round of messages (9 messages) sent by each node, router 5 holds the information about how to get to all the other routers but all the other routers only have partial knowledge of the whole graph; therefore, all the routers except router 5 need to receive at least another message to fill the whole table. After the router 5 sends a message to its recipients (all the other routers in the graph), every router gets the information about how to get to any router in the graph. So 9+5=14 messages are required to be sent.

**Question 2.**

For the same situation (and starting from the same initial state), how many messages must be sent before all packets sent from any node to any other node are guaranteed to follow a shortest path?  Briefly explain.

Assuming the every single path cost in the graph is the same, 14 messages are required to be sent. In the case that the minimum distance from one router to another router is 1, the routing table is updated during the first round of sending messages in this case. In the case that the minimum distance from one router to another router is 2, the path that passes through router 5 to the destination router (i.e. 1->5->3) has the same distance as all the other shortest paths (if exist) to the destination router. And the minimum distance from one router to another router in this graph is less than or equal to 2; therefore, when the router 5’s second message is sent, all the routers know how to get to other routers following the shortest path. So 14 messages are required to be sent.

72 messages are required to be sent before all packets sent from any node to any other node are guaranteed to follow a shortest path. Although the weights are not shown, the graph is a weighted graph, which means the minimum-hop path doesn’t necessary equals to the shortest path. Assume that a path doesn’t visit the same router twice. Under the assumption, the maximum-hop path from one router to another router is 8 hops, which means each router needs to send message 8 times to be sure that they are guaranteed to follow the shortest path.

**Question 3.**

For the same starting situation, but this time using a **link state** routing table update scheme, how many messages must be sent before every node knows how to reach every other node?  Briefly explain.

5 messages are required to be sent. After the fourth message is sent, router 1 to 4 still don’t know how to get to every other router in the graph. After router 5’s message is sent, all the router knows how to get to others since every router in the graph is connected to router 5. After that, each router can start to compute the shortest path to other routers. So 5 messages are required to be sent.

**Question 4.**

The IPv6 address space is so big that it's just about the size required to allocate an entire IPv4 network per

molecule in the universe ~10^53

star in the universe ~10^11

X star in our galaxy ~10^22-10^24

person on Earth ~10^9

**Question 5.**

TCP uses ARQ and sliding window with cumulative ACKs.  TCP is said to be a reliable protocol.  Briefly, and as precisely as you can, define what we mean when we say TCP is reliable.  Make sure you address both of these:  
**(A)**Just what reliability properties does TCP guarantee?  
**(B)**What properties, if any, must be satisfied by the physical network over which TCP data is transferred for TCP to achieve those goals?

(A) TCP guarantees that packages are transferred in order (of sequence number) and retransmitting the packages when they are lost.

(B) The wire that TCP connections are on must be connected and not broken so that the signal can be transfer over the wire.

**Question 6.**

Tor61 streams are carried over TCP links.  Is there any difference in the reliability properties provided by Tor61 streams and TCP connections?  If so, give a convincing example of how they are different.  If not, briefly list the reliability properties they both provide and explain why Tor61 doesn't provide any additional properties.  (Note: assume that both TCP and Tor61 are correctly implemented.)

Since Tor61 is implemented on top of TCP, it also has the reliability properties that TCP connections provide. But there are some differences between them. Tor61 also ensure extra reliabilities of stream and circuit of Tor61 nodes by using cells that indicate failures occurred, such as Relay Extend/Begin Failed.

**Question 7.**

802.11 (wireless) operates much like a CSMA/CD protocol.  It performs carrier sense, by listening before sending and deferring transmission if the medium is sensed busy.  It is multiple accesses, in that all stations can try to use the medium at any time (subject to the other protocol rules).  It is collision detecting in the sense that 802.11 requires that receivers of frames send explicit ACKs, and the failure to receive an ACK is taken to mean a collision occurred.  802.11 employs binary exponential backoff for collision resolution.  
  
802.11 also allows each individual station to choose a transmission rate best suited to its conditions.  A station located near the AP may transmit at twice the bit rate of one located further from the AP, for instance, because of its higher received signal strength.  
  
Explain how (in the absence of other mechanisms designed to address it) these properties result in the effective goodput rate of all stations being (largely) determined by the worst connected one.  For example, if two stations always have something to send and one is transmitting at 10 Mbps and the other at 100 Mbps, both will achieve goodput a little below 10 Mbps.

Assume same frame size s. Since only one can transmit data at a single moment, so average goodput would be s/(s/100 + s/10) = 100/11 ~ 9.09 Mbps < 10 Mbps. Goodput rate of all stations is determined by the worst connected one is because carrier sense, which only allow one to transmit at the same time and the other has to wait for the station done with its transmission.

**Question 8.**

The Domain Name System (DNS) is a service that maps domain names (e.g., attu1.cs.washington.edu) to IP addresses.  DNS is largely intended for statically assigned IP addresses, ones that are "permanently" assigned to specific machines.  
  
It can be useful to allow DNS lookup for machines that go offline frequently, and that use DHCP to acquire an address when they come back online.  Various "Dynamic DNS" (DDNS) systems have been built to provide this.  
  
This question asks you to provide a bare bones design of a DDNS system, for the moment ignoring all security issues.  In particular, please answer the following sub-questions:  
**(A)**What message types does your DDNS system require?   That is, your message header will have some kind of command field.  What are the commands, and what are they used for?  
**(B)**For each message type, what data must be included in messages of that type?  (We don't care about specific layout of the data, we just want to know what is sent.)  
**(C)**When a DDNS client machine comes online, how will it discover the IP address of a DDNS server with which it should register?

(A) The message header should contain sequence number and a value that indicates what type of command this message is. Command A (or A records) maps hostname to IPv4. Command AAAA maps hostname to IPv6. Command PTR maps IP to hostname. Command NS specifies the authoritative nameservers for the domain. Command UPDATE updates the master name when transferring entire zone file from the master name server to secondary name servers.

(B) A/AAAA records contain the string of hostname as data. PTR records contain an IP address as data. NS records contain a domain name as data. UPDATE records contain a zone which specifies the zone name to update to.

(C) The client broadcast to discover all nodes on the network, which discover the IP of DDNS server. After the broadcast, the DDNS server offers it an IP address using DHCP, following by request from client and ACK from server. At that the, the client is registered.

**Question 9.**

Because there are no security mechanisms in DDNS, anyone can register an address for any name.  Explain how you could secure DDNS so that only the authorized machine could update the address associated with its name, even if attackers could make copies of all messages exchanged between that machine and the DDNS system (and replay those messages later).

Using the RSA asymmetric encryption would help to achieve the security mechanisms. The client uses its private key to encrypt the message to be sent and uses the system’s public key to encrypt the checksum of the message, and then sends the whole encrypt message plus checksum to the server, and the server receives the message and decrypt using the client’s public key to get the message, decrypts the checksum using its private key and checks if it matches up with checksum of the decrypted message. Drop the message if they don’t match each other. Sequence number also needs to be included in message to be sent so that the system can prevent the duplicate messages result in bad effect.

**Question 10.**

The infrastructure for the Bitcoins project includes a server that streams the binary data file over a TCP connection (the server at cse461.cs.washington.edu:46114).  Implement that server, in any of Java, C/C++, or Python, and put the code in the answer to this question.  Your server should take a single command line argument, the name of the file to stream.  You should use port 46100 to wait for incoming TCP connections.  When there is a connection, you simply start sending the binary contents of the file on it; you never read from the connection.  Your implementation must support more than one concurrent connection at a time.  We will compile and run your server (so you should as well!).

import java.io.BufferedInputStream;

import java.io.BufferedOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.net.InetAddress;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.UnknownHostException;

public class StreamBinaryFileServer {

private static final int port = 46100;

private static ServerSocket sSocket = null;

public static void main(String[] args) throws IOException {

if (args.length != 1) {

System.out.println("Usage: java StreamBinaryFileServer <binary file name>");

System.exit(1);

}

String filename = args[0];

File fileToStream = new File(filename);

InetAddress ip;

sSocket = new ServerSocket(port);

try {

ip = InetAddress.getLocalHost();

System.out.println("IP address : " + ip.getHostAddress() + " port : " + port);

} catch (UnknownHostException e) {

e.printStackTrace();

}

Socket cSocket = null;

while (true) {

cSocket = sSocket.accept();

Thread connectionHandler = new Thread(new StreamingHelper(cSocket, fileToStream));

connectionHandler.start();

}

}

static class StreamingHelper implements Runnable {

private Socket cSocket;

private File fileToStream;

public StreamingHelper(Socket cs, File fts) {

cSocket = cs;

fileToStream = fts;

}

@Override

public void run() {

BufferedInputStream fileReader = null;

try {

fileReader = new BufferedInputStream(new FileInputStream(fileToStream));

BufferedOutputStream outToClient = new BufferedOutputStream(cSocket.getOutputStream());

System.out.println("Start streaming");

byte[] buffer = new byte[1024];

int byteToWrite;

while ((byteToWrite = fileReader.read(buffer)) != -1) {

outToClient.write(buffer, 0, byteToWrite);

outToClient.flush();

}

fileReader.close();

System.out.println("Done with streaming");

cSocket.close();

} catch (FileNotFoundException e) {

System.out.println("File not found");

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**Question 11.**

**(A)**Briefly explain why not all of the registration server protocol of Project 1 is compatible with NATs.  
**(B)** Describe how to modify the protocol so that it is compatible with NATs.  (Your modifications don't need to be compatible with clients running the original protocol.)

(A) The second port of the registration agent for receiving probe message is not compatible with NATs. That port cannot receive the server’s probe since it never sends message to the server before it receive a message from the server. But the NAT translation table doesn’t have the translation for that port and IP yet, so it cannot receive a message from the server unless it sends something to the server first.

(B) Using the second port to send any message to the server before the register message is sent by the first port, so that the second port and IP are stored in the translation table and are able to receive probe message from the server.

**Question 12.**

Suppose Alice creates a transaction representing a payment of 5 Bitcoins to Bob.  What prevents Bob from making a copy of that transaction and resubmitting it later to earn another 5 Bitcoins from Alice?

For a transaction to be valid, no previous output can have been already used as an input by any other transaction. When Bob makes a copy of the transaction, the input specifier still contains the same reference to the same previous transaction output. In this case, we have two transactions reference to one previous transaction output, which violates the definition of valid transaction; therefore, the transaction verifier will see that and drop one of the transactions.

**Question 13.**

Briefly argue that hierarchically organized domain names (e.g., attu1.cs.washington.edu) are essential to achieving scalability.  Try to be brief but specific.

Hierarchically organized domain names manage the domain names in a tree structure. At each level domains simply just looking for the distributed directory specified by the client and going down to the next level then do the same thing again until it reaches the start of the host name; therefore, we can add new node/layer to the DNS namespace tree easily without modifying the mechanism. So it achieves scalability.

**Question 14.**

CIDR addresses enable supernetting.  Explain how supernetting is useful to achieve Internet scalability.

Supernetting when maximum number of hosts that a subnet can handle needs to be increased. It can combine two or more subnets with continuous IP address (and lower subnet mask bit/number) to create a larger network in order to satisfy the need of handling more hosts.

**Question 15.**

Internet routers do not provide reliability guarantees, but offer instead only best-effort delivery.  That allows routers to drop packets whenever they feel like it.  
  
Suppose the designers of the Internet had instead decided that routers should be reliable.  What serious problem could arise in that case (due to the decision to be reliable)?

Sequence numbers are needed for routers to be reliable. By adding sequence numbers to the protocol, extra memory is also needed for storing the ACK number of each host connected to the router, which might be a lot of memory needed if it’s a wireless router. Retransmission is also needed for it to be reliable. Since it’s using TCP, the packets of the response corresponds to the request must all arrived to the host’s end and must arrived in order. That means the router needs to have some like sliding window for each host to achieve retransmission, which is another thing that needs a lot of memory. When the memory in the router is jammed up due to too many host connected at the same time, it will not be able to accept any new connection.

**Question 16.**

Protocols consisting entirely of idempotent operations can be attractive, when they are possible.  Why?

Since idempotent operations ensure that the same operation executed multiple time will have the same effect as executed it one time, so that programmers don’t have to worry about how to deal with duplicate same packages themselves, which makes programmers’ lives much easier.

**Question 17.**

Is the registration service protocol (Project 1) idempotent?

X Yes

No

**Question 18.**

Is the Tor61 protocol (Project 3) idempotent?

X Yes

No

**Question 19.**

**(A)**UDP provides framing information visible to programs using it in the form of packets.  Briefly explain what program visible framing information TCP provides.  
**(B)**UDP framing isn't very useful to an application using it if that application needs frames larger than a UDP packet.  When might TCP framing not be very useful to an application?  (Either describe the general case or else give an individual, specific example case.)

(A) source port, destination port, sequence number, and ACK number

(B) TCP framing is not very useful for online real-time video streaming applications since TCP ensures that every package arrived in order, which may causes some delay due to retransmissions and not sending packages because it haven’t received the ACK of the previous sent package.

**Question 20.**

**(A)**Briefly explain what the TCP "3-way handshake" is and what it is used for.  
**(B)**As with all uses of sliding window, ACKs in TCP must be reliable.  Briefly explain how reliability of ACKs is ensured by TCP.

(A) TCP 3-way handshake is the steps required to initiate a TCP connection. First the host A sends his sequence number x to another host B, then B responds with his sequence number y and ACK number x+1. Finally, A replies ACK number y+1 back to B. Usually, A will start to transmit the data he wants to send at the final step. 3-way handshake is used for synchronizing the sequence number on both ends. By not starting at sequence number 0, it also prevents two incarnations of the identical connection reusing the same sequence number, which might cause two connection interfere with each other.

(B) TCP uses cumulative acknowledgment to ensure the reliability. Cumulative acknowledgment means the receiver send an ACK with next package’s sequence number that indicates he receive all the packets preceding that sequence number, so that some ACKs lost are not a big problem since the ACK with sequence number succeed them will acknowledge the receiver gets all those packages whose ACKs are lost.