BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)

II Semester 2019-20 CS F301 Computer Networks

Mid Semester Exam (CLOSED BOOK)

Date: March 05, 2020 Duration: 90 minutes Weightage: 60M (30%)

Note:

Q1[6M]. Answer the following:

a) Suppose Alice, with a Web-based e-mail account (such as Hotmail or gmail), sends a message to Bob, who accesses his mail from his mail server using POP3. Discuss how the message gets from Alice's host to Bob's host. Be sure to list the series of application-layer protocols (and the corresponding transport layer protocols) that are used to move the message between the two hosts. Write this answer as a sequence of steps (Step-1 to Step-n).

Sol:

Step 1: Alice's host to Alice's mail server: HTTP (using TCP)
Step 2: Alice's mail server to Bob's mail server: SMTP (using TCP)
Step 3: Bob's mail server to Bob's host: POP3 (using TCP)

b) Suppose the URL of a webpage is www.abc.com/index.html. Besides basic HTML, the webpage has one video file and two gif files. The URL associated with the video file is www.cdn.com/video1.mpg. Also, the URL associated with the two gif files are www.abc.com/one.gif, www.abc.com/two.gif. Consider a user who opens up a web browser and puts www.abc.com. Assuming that all the caches are empty, write the sequence of steps with underlying protocols, along with the message type, used to access the webpage. Assume here that TCP uses persistent connection with no pipelining.

Sol: Below steps assume that video1.mpg is sent using TCP. Otherwise, it can be sent using UDP also

Step 1: Invoke DNS to get the IP address of www.abc.com.
Step 2: Establish TCP connection with www.abc.com.
Step 3: HTTP GET request to abc.com. HTTP response containing basic HTML of abc.com.
Step 4: Two HTTP GET requests for one.gif and two.gif. HTTP response containing .gif files.
Step 5: Invoke DNS to get the IP address of www.cdn.com.
Step 6: TCP connection with www.cdn.com
Step 7: HTTP GET request for video1.mpg. HTTP response containing video1.mpg.

Q2[10M]. Answer the following. Assume $\mathbf{T_p}$ to be propagation delay, $\mathbf{T_{pr}}$ to be the processing delay, $\mathbf{S_d}$ be the DATA packet size (in bits), $\mathbf{S_a}$ be the ACK packet size (in bits), **R** is the rate of transmission at both sender and receiver, and $\mathbf{S_h}$ be the size of header in the data packet. Ignore the size of header in ACK packet and assume the processing delay is same for both DATA as well as for the ACK packet.

a) Derive the efficiency of Stop and Wait protocol in terms of the above parameters.

Solution: The time required to send a frame and receive an ACK in the absence of error

$$T_0 = 2T_p + 2T_{pr} + \frac{S_d}{R} + \frac{S_a}{R}$$

$$\text{Efficiency} = \frac{(S_d - S_h)/T_0}{R}$$

$$\text{Efficiency} = \frac{1 - \frac{S_h}{S_d}}{1 + \frac{S_a}{S_f} + \frac{2(T_p + T_r)R}{S_d}}$$

b) Now, extend the formula derived in **part (a)** to calculate the expected number packets that are required to be transmitted by the sender to send **M** packets when the probability of a packet loss is **P**.

Solution: **Expected number of attempts required to transmit a frame successfully is $\frac{1}{1-P}$ (Geometric Distribution)**

Therefore, the value of T_0 in case of error will become $\frac{T_0}{1-p}$

$$\textbf{Efficiency} = \frac{1-\frac{S_h}{S_d}}{1+\frac{S_a}{S_f}+\frac{2(T_p+T_r)R}{S_d}} \quad (1-P)$$

Q3 [8M]. Answer the following:

a) In TCP, after the three-way handshake, a TCP session is established. Now onwards, both ends can send data to each other. In a "TCP session hijacking attack", an attacker can generate a stream of spoofed packets and send it to the receiver such that the receiver cannot tell whether the packets are coming from the real sender or an attacker. Write the name of TCP/IP header fields an attacker has to spoof correctly to launch the "TCP session hijacking" attack successfully.
Sol: An attacker will have to spoof Source IP address, Source Port Number, and Sequence Number to launch the "TCP Hijacking Attack".

b) Suppose a sender has a window size of **5 packets**, and it has to send **10 packets**. Also, every **5th packet** is lost by the network. How many total packets will be transmitted, if we use Go-Back_N, and Selective Repeat protocol?

Sol: Go-Back N, infinite packets will be sent and the sequence would be as follows:

1, 2, 3, 4, 5, 6, 7, 8, 9, 5, 6, 7, 8, 9, 5, 6, 7, 8, 9, ……. and the sequence will continue.

With Selective Repeat, 12 packets will be sent and the sequence would be as follows:

1, 2, 3, 4, 5, 6, 7, 8, 9, 5, 5, 10

Q4 [10M]. Answer the following:

a. Suppose a TCP payload of **2000 bytes** of data and **20 bytes** of TCP header is passed to the IP layer. Suppose this host is connected to a link with an MTU size of **512 Bytes**. What is the total length of the last IP fragment (in bytes), and the value of the offset field in the last fragment?

Ans. Total length of the las IP fragment 88
Answer Description
Total payload to the IP layer is 2020 bytes.
The MTU size is 512, out of which payload size is only 492. Since 492 is not a multiple of 8, payload is 488. 2020 bytes of data has to be distributed over 5 fragments, where the first 4 carry 488 bytes and the last carries the remaining 68. Total length of the last fragment is 68+20 = 88

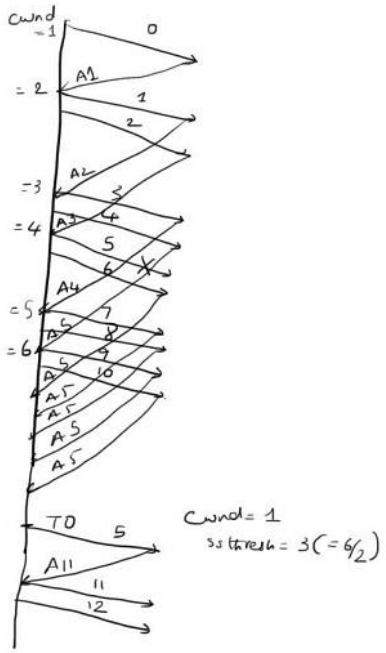
The value of the offset field in the last fragment = 244

b. Suppose the maximum time an IP datagram can stay in the network before being delivered to the receiver is **60 sec** (MSL: Maximum segment lifetime is 60 sec). What is the maximum rate a host should send out datagrams so as to avoid confusion during reassembly of fragments at the receiver? Assume the datagram size of **1000** Bytes and the maximum rate at which the underlying link layer technology can transmit the frames is **10 Mbps**. Express the answer in Mbps.

Ans. The correct answer is 8.7 Mbps
Answer Description
We need to ensure that the identification field does not wrap around within 60 sec. Since ident field is 16 bits, that means a host can send 2^16 datagrams in 60 sec before wrap around occurs. So, data rate of sending should be kept under 2^16 * 1000 * 8 bits / 60 sec to prevent wrap around. As long as its under, given the fact that no datagram stays past 60 sec in the network, wrap around does not happen.

Q5 [10M]. Assume TCP version Tahoe. Draw the packet transmission timeline when packet with sequence number **5** is lost and answer the following series of questions. Assume that the sequence number of packets starts with **0**. Consider the time out period is sufficiently large so that, when timeout occurs, all the acknowledgements sent by the receiver are received by the sender if they are not lost.

Solution:



a) How many duplicate *acks* are received by the sender before the timeout event for packet 5?

Ans. The correct answer is 5
The dupacks ask for seqno 5. The first (nonduplicate) is generated by packet with seqno4. Since packet 5 is lost, packets 6,7,8,9,10 generate dupacks leading to a total of 5 dupacks.

b) What is the value of *ssthresh* right after the timeout event?

Ans. The correct answer is 3
Answer Description
Proper acks are received for packets with seqno 0,1,2,3,4. Hence cwnd is incremented 5 times leading to a value of 6. After timeout ssthresh is set to 1/2 the value i.e. 3.

c) Once the retransmitted packet is successfully received, the receiver asks for what packet?

Ans. The correct answer is 11
Answer Description
In the past packets with seqno upto 10 were sent. So, the receiver will ask for packet 11.

d) Once the retransmitted packet is successfully received and the receiver *acks* this, what packets are transmitted by the sender on receiving this *ack*?

Ans. The correct answer is 11,12
Answer Description
In slow start, you can send two packets for every ack. So, 11 and 12 are sent. Its still in slow start here since ssthresh is 3.

Q6[10M]. State TRUE/FALSE with justification:

a) If underlying Network does not drop or corrupt packets, reliability at transport layer is not required.

False

(i) The Transport layer still has to deal with out-of-order delivery

(ii) The network layer ensuring node-to-node delivery without packet corruption or drops, does not ensure process-to-process reliable delivery, as the packets may get corrupted or lost at the host during multiplexing and de-multiplexing.

b) Consider a reliable data transfer protocol that uses only negative acknowledgements (i.e. messages from the receiver indicating that particular data was not received). Suppose the sender sends data only infrequently. In this case, a *NAK-only* protocol would be preferable then the *ACK-only* protocol.

False

In case of infrequent data transmission and NAK-only policy, the receiver has to keep on sending the NAKs to the sender, in anticipation that the frame transmitted by the sender might have been dropped. And this would incur a lot of overhead. On the other hand, in ACK-only policy the receiver has to acknowledge the receipt of a frame infrequently.

c) TCP uses congestion window header field for congestion control.

False

TCP used a field named "window" for the flow control purpose. For congestion control the sender maintains a variable called congestion window (cwnd) locally and does not communicate that value to the receiver using any field of the packet header.

d) The shared bus backplane of a router operates at 4 times the line speed and the router supports 3 line cards. This may result in queue build up at the output ports?

True

In worst cast, packets arriving at all the input port may be routed through the same output port and this would cause the queue to build up at that output port.

e) Every intermediate router calculates the checksum of every IP packet received by it, to detect the error, but does not modify it.

False

The router not only validated the checksum but also modifies it, as the IP checksum covers all the header field and the TTL field do get changed at every router.

Q7[6M]. The IP protocol provides best effort service. But just for argument sake, suppose it wanted to provide in-order delivery service as well. Is it feasible to implement this? If so, how would you implement it? Be clear on what functionality is needed and where would this functionality be placed?

Solution:

Theoretically, guarantees of any kind are impossible. TCP provides reliable delivery. If a link is so bad, it drops all packets it receives, what can TCP do, it will try and try and give up after some time.

Now coming to the answer. In order service is feasible at the network layer. Note, we don't really need to insist that all packets of a flow take the same path. All we need to do is sequence the packets (network layer header has this new field) and at the receiver network layer, buffer them and send them in order. This is a bit like the fragmentation and reassembly function done at the network layer. This will add additional delay to the packets. If some packet is lost, one will time out eventually and pass packets in buffer to the higher layers. Note that TCP if being used, can recover the lost packet, network layer has to wait sufficiently long for this recovery to finish before giving up on the lost packet. In a way, instead of TCP implementing it, you are implementing it at network layer.

In practice though this in-order-delivery functionality fits in best at transport layer since reliable delivery anyways needs to use sequence numbers and has provisions for recovery of lost packets and a sense of how to set 'give-up-timers' correctly.