Computer Networks

January 22, 2019

Contents

KR. R13 (Chap 1 page 96)

Suppose users share a 2 Mbps link. Also suppose each user transmits continuously at 1 Mbps when transmitting, but each user transmits only 20 percent of the time. (See the discussion of statistical multiplexing in Section 1.3.)

a. When circuit switching is used, how many users can be supported?

b. For the remainder of this problem, suppose packet switching is used. Why will there be essentially no queuing delay before the link if two or fewer users transmit at the same time? Why will there be a queuing delay if three users transmit at the same time?

c. Find the probability that a given user is transmitting.

d. Suppose now there are three users. Find the probability that at any given time, all three users are transmitting simultaneously. Find the fraction of time during which the queue grows.

Answer

A.

From the task description we know that all users share a 2 Mbps link and each user transmits continuously at 1 Mbps. We can therefor conclude that only 2 users can be supported because each user requires half of the link bandwidth.

B.

The reason that 2 or fewer users dont course a delay is because all users share a 2 Mbps link. This link can be represented as a tube that has a size of 2 Mbps. We know that each user transmits at 1 Mbps. That means that we can only fit as many user that can be inside the tube which in this case is 2. More than that will therefor course a delay. If we have a case where we are 3 then we have to wait until a user has passed the tube befor the third can be transmitted. Because of the bandwidth/tube size of 2 Mbps we can't fit 3 but instead would need a bandwidth/tube size of 3 Mbps to cover 3 users.

C.

We can find the probability that a given user is transmitting by converting 20 percent too decimal. By doing this we get 0.2.

D.

Probability that all three users are transmitting simultaneously is computed as.

$$\binom{3}{3}\mathbb{P}^3(1-\mathbb{P})^{3-3} = (0.2)^3 = 0.008$$

Since the queue grows when all users are transmitting the fraction of time during which the queue grows (which is equal to the probability that all three users are transmitting simultaneously) is 0.008.

KR. R14 (Chap 1 page 96)

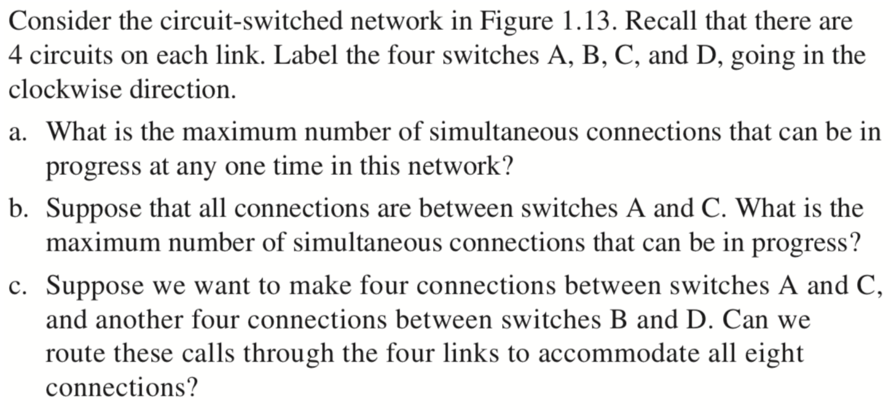
> Why will two ISPs at the same level of the hierarchy often peer with each other? How does an IXP earn money?

Answer

A.

The reason that the two ISP's at the same level of the hierarchy often peer with each other is because it is cheaper. Looking at figure 1.15 page 62 it would mean that if it was the case that there weren't any connection between the two regional ISP they would have to sent the given signal to the tier 1 ISP where the other regional ISP would have to pay to get that signal The IXP earns money by redirecting a signal directly to either the tier 1 ESP or another content provider.

KR. P4 (Chap 1 page 98)



Consider the circuit-switched network in Figure 1.13. Recall that there are 4 circuits on each link. Label the four switches A, B, C, and D, going in the clockwise direction.

a. What is the maximum number of simultaneous connections that can be in progress at any one time in this network?

b. Suppose that all connections are between switches A and C. What is the maximum number of simultaneous connections that can be in progress?

c. Suppose we want to make four connections between switches A and C, and another four connections between switches B and D. Can we route these calls through the four links to accommodate all eight connections?

Answer

A.

Because we have 4 links where each link has 4 circuits and each link can support 4 simultaneous connections we can make the following computation.

$$\text{Simultaneous connections at once}: \ 4 \cdot 4 = 16$$

B.

Because we have 2 links where each link has 4 circuits and each link can support 4 simultaneous connections we can then make the following computation.

$$\text{Simultaneous connections between A and C}: \ 4 \cdot 2 = 8$$

C.

Yes For the connection between A and C we route two connections through B and two connections through D. For the connection between B and D we route two connections through A and two connections through C. In this manna there at most 4 connections passing through any link.

KR. P13 (Chap 1 page 100)

a)

Suppose N packets arrive simultaneously to a link at which no packets are currently being transmitted or queued. Each packet is of length L and the link has transmission rate R. What is the average queuing delay for the N packets?

b)

Now suppose that N such packets arrive to the link every LN/R seconds. What is the average queuing delay of a packet?

Answer

a)

The queuing delay is 0 for the first transmitted packet, L/R for the second transmitted packet, and generally, $(n-1)L/R$ for the n^{th} transmitted packet. Thus, the average delay for the N packet is

$$\begin{aligned}(L/R + 2L/R + \cdots + (N-1)L/R)/N &= L(RN) \cdot (1 + 2 + \cdots + (N-1)) \\ &= L(RN) \cdot N(N-1)/2 \\ &= LN(N-1)/(2RN) \\ &= (N-1)L/(2R)\end{aligned}$$

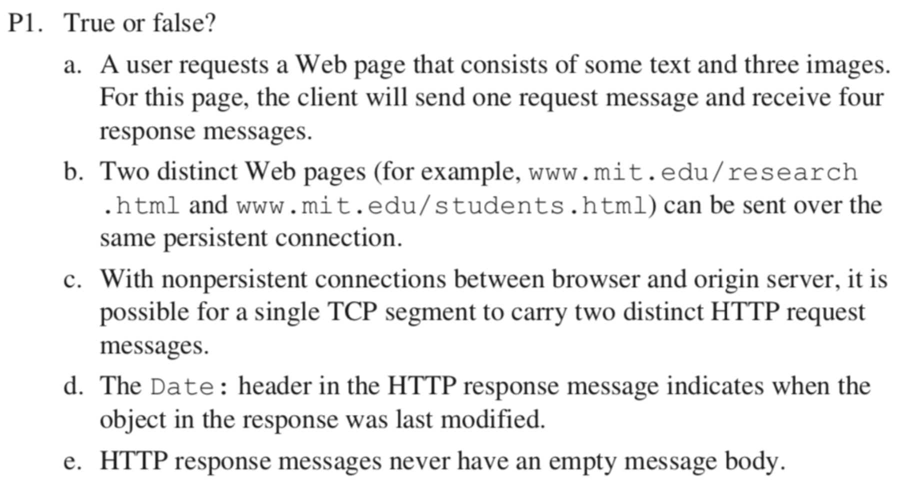
Note that here we use the well-known fact:

$$1 + 2 + \cdots + N = N(N+1)/2$$

b)

It takes LN/R seconds to transmit the N packets. Thus, the buffer is empty when a each batch of N arrive. Thus, the average delay of a packet across all batches is the average delay within one batch, i.e., $(N-1)L/2R$

KR. P1 (Chap 2 page 201)



P1. True or false?

a. A user requests a Web page that consists of some text and three images. For this page, the client will send one request message and receive four response messages.

b. Two distinct Web pages (for example, `www.mit.edu/research.html` and `www.mit.edu/students.html`) can be sent over the same persistent connection.

c. With nonpersistent connections between browser and origin server, it is possible for a single TCP segment to carry two distinct HTTP request messages.

d. The `Date:` header in the HTTP response message indicates when the object in the response was last modified.

e. HTTP response messages never have an empty message body.

Answer

a)

False. A server will only send one response per request message.

b)

True. With persistent connections, the server leaves the TCP connection open after sending a response. Subsequent requests and responses between the same client and server can be sent over the same connection.

c)

False. A brand-new connection must be established and maintained for each requested object.

d)

False. It's the `Last-Modified` header line that indicates the time and date when the object was last modified.

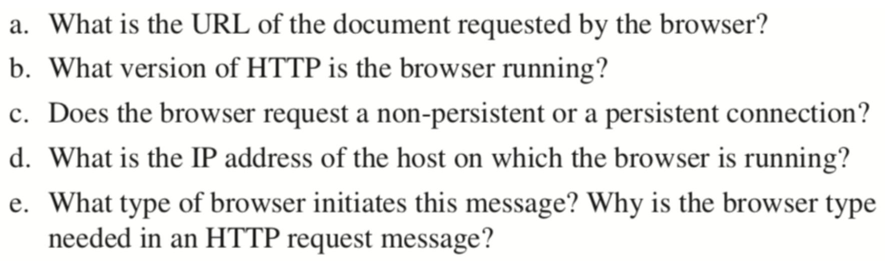
e)

False. A HTTP response message can have an empty massage body.

KR. P4 (Chap 2 page 201)

Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters *<cr><lf>* are carriage return and line-feed characters (that is, the italized character string *<cr>* in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.

```
GET /cs453/index.html HTTP/1.1<cr><lf>Host:  gai
a.cs.umass.edu<cr><lf>User-Agent:  Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8,image/png,*/*;q=0.5
<cr><lf>Accept-Language:  en-us,en;q=0.5<cr><lf>Accept-
Encoding:  zip,deflate<cr><lf>Accept-Charset:  ISO
-8859-1,utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive:  300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```



a. What is the URL of the document requested by the browser?
b. What version of HTTP is the browser running?
c. Does the browser request a non-persistent or a persistent connection?
d. What is the IP address of the host on which the browser is running?
e. What type of browser initiates this message? Why is the browser type needed in an HTTP request message?

Answer

a)

The URL of the document is `gaia.cs.umass.edu/cs453/index.html`.

b)

The version of HTTP that the browser use is `HTTP/1.1`.

c)

The browser requests a persistent connection, because of the `Connection:keep-alive` header.

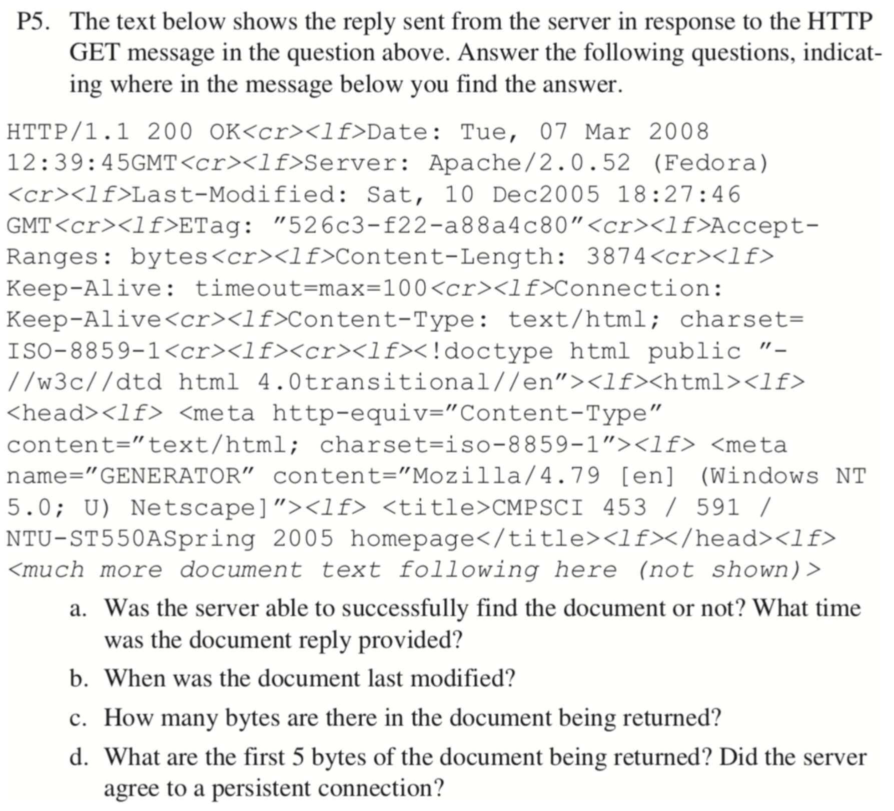
d)

You can't tell the IP-address from an HTTP message.

e)

Mozilla/5.0. Can be found under `User-Agent` header.

KR. P5 (Chap 2 page 202)



P5. The text below shows the reply sent from the server in response to the HTTP GET message in the question above. Answer the following questions, indicating where in the message below you find the answer.

```
HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008
12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)
<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46
GMT<cr><lf>ETag: "526c3-f22-a88a4c80"<cr><lf>Accept-
Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>
Keep-Alive: timeout=max=100<cr><lf>Connection:
Keep-Alive<cr><lf>Content-Type: text/html; charset=
ISO-8859-1<cr><lf><cr><lf><!doctype html public "-
//w3c//dtd html 4.0transitional//en"><lf><html><lf>
<head><lf> <meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1"><lf> <meta
name="GENERATOR" content="Mozilla/4.79 [en] (Windows NT
5.0; U) Netscape]"><lf> <title>CMPSCI 453 / 591 /
NTU-ST550ASpring 2005 homepage</title><lf></head><lf>
<much more document text following here (not shown)>
```

a. Was the server able to successfully find the document or not? What time was the document reply provided?

b. When was the document last modified?

c. How many bytes are there in the document being returned?

d. What are the first 5 bytes of the document being returned? Did the server agree to a persistent connection?

Answer

a)

Yes, the server was able to found the document. The document was provided on `Tue, 07 Mar 2008 12:39:45GMT`.

b)

The document was last modified on `Sat, 10 Dec 2005 18:27:46 GMT`.

c)

3874. See `Content-Length` header.

d)

The first five bytes of the document is `<!doc`. Yes the server agreed to persistent connection based on the `Connection:Keep-Alive` header.

KR. P10 (Chap 2 page 203)

Assume you request a web-page consisting of one document and five images. The document size is 1 kbyte, all images have the same size of 50 kbytes, the download rate is 1 Mbps, and the RTT is 100 ms. How long does it take to obtain the whole web-page under the following conditions? (Assume no DNS name query is needed and the impact of the request line and the headers in the HTTP messages is negligible).

a. Nonpersistent HTTP with serial connections.
b. Nonpersistent HTTP with two parallel connections.
c. Nonpersistent HTTP with six parallel connections.
d. Persistent HTTP with one connection.

Answer

a)

We know from our task description that we would have to make two computations. one for the one document and a second one for the five images. The most important thing of this exercise is to make sure that everything has the same value. In this case the easiest thing is to compute everything to bits. We do that with Document size: 1 Kbyte, Image size: 50 Kbytes and Download rate: 1 Mbps.
Now we can start our 1st computation.

$$1 \cdot \left(1 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 = 0.21$$

What is happening above is that we first have the number 1 which represents the number of documents. Then we have another 1 inside our parentheses which is the size of the document. This is multiplied with a fraction where the top frac is the document size computed to bits and the lower frac is the 1 Mbps download rate converted to bits also. On the other side of the parentheses we have 2 multiplied with the RTT because it is to the end and back. The reason why the RTT is 0.1 and not 100 is because it is converted to sec. By adding all this up we get the result above.

We can now make the second computation.

$$5 \cdot \left(50 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 5 \cdot 2 \cdot 0.1 = 3.8sec$$

Exactly the same thing happens as in the first computation we have now just changed the the first and second numbers with 5 and 50. where 5 represents the number of images and 50 the size of the images.
We can now add the two.

$$1 \cdot \left(1 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 + 5 \cdot \left(50 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 5 \cdot 2 \cdot 0.1 = 4.01sec$$

b)

We now have to make nonpersistent HTTP with only two parallel connections where we before had 6 connections. We just change the number 5 to 2 and get the following computation.

$$1 \cdot \left(1 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 + 5 \cdot \left(50 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 3 \cdot 2 \cdot 0.1 = 3.61sec$$

c)

When we have a nonpersistent HTTP with six parallel connections it means that all the segments gets send at one time. We can therefor just make the following computation.

$$1 \cdot \left(1 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 + 5 \cdot \left(50 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 = 3.21sec$$

d)

In this task we only to create a connection one time. That means that every request we make only needs one RTT.

$$1 \cdot \left(1 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 2 \cdot 0.1 + 5 \cdot \left(50 \cdot \frac{1000 \cdot 8}{1 \cdot 10^6}\right) + 0.1 + 1 = 3.11sec$$

KR. P21 (Chap 2 page 207)

Suppose that your department has a local DNS server for all computers in the department. You are an ordinary user (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

Answer

Yes, we can use *dig* to query that web-site in the local DNS server.

For example, `dig dr.dk` will return the query time for finding *dr.dk*. If *dr.dk* was accessed a couple of seconds ago, an entry for *dr.dk* is cached in the local DNS cache, so the query time is 0 msec. Otherwise the query time will be bigger.

KR. P25 (Chap 2 page 207)

Suppose Bob joins a BitTorrent torrent, but he does not want to upload any data to any other peers (so called free-riding).

a. Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is BobâĂŹs claim possible? Why or why not?

b. Bob further claims that he can further make his âĂIJfree-ridingâĂİ more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?

Answer

a)

Yes. As long as there is enough peers in the swarm for a long time enough time. Bob can always receive data through optimistic unchoking other peers.

b)

He can do that by running a client on each host and let each client "free-ride". He can then combine the collected chunks from the different hosts into a single file.

KR. P14 (Chap 3 page 319)

Consider a stop-and-wait data-transfer protocol that provides error checking and retransmissions but uses only negative acknowledgments. Assume that negative acknowledgments are never corrupted. Would such a protocol work over a channel with bit errors? What about over a lossy channel with bit errors?

Answer

Yes it would work over a channel with bit errors. The reason is that the receiver would just count that it has got packet if doesn't get any negative acknowledgments and can therefor keep sending packages until it gets a negative acknowledgments. It wouldn't work over a lossy channel because here it is underlying that a channel can lose its packets, and therefor it is necessary to send an ACK.

KR. P15 (Chap 3 page 320)

Consider the cross-country example shown in Figure 3.17. How big would the window size have to be for the channel utilization to be greater than 98 percent? Suppose that the size of a packet is 1,500 bytes, including both header fields and data.

Answer

L needs to be in bits and that is the reason we multiply with 8.

$$d_{trans} = \frac{L}{R} = \frac{1500 \cdot 8 \text{ bits/packet}}{10^9 \text{bits/sec}} = 0.12 msec$$

From the book we know that
t = RTT + L/R = 30.008 msec
We can now compute the utilisation at 98% of the time, we must set

$$0.98 = (0.012n)/30.012$$

Where n is the window
We can now conclude that the window is 2450.98 packets, or 2451 packets.

KR. P27 (Chap 3 page 322)

Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.

a. In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?

b. If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?

c. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?

d. Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost and the second acknowledgment arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgments sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes of data; for each acknowledgment that you add, provide the acknowledgment number.

Answer

a)

Sequence number = sequence number of the first segment + number of bytes of data in first segment.

$$127 + 80 = 207$$

Source port number = 302. Destination port number= 80.

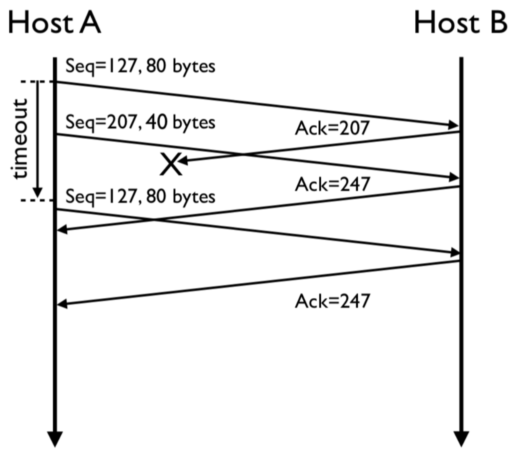
b)

Acknowledgment number = 207.
Source port number = 302. Destination port number= 80.

c)

The acknowledgment number is 127, indicating that it is still waiting for bytes 127 and onwards.

d)



KR. P36 (Chap 3 page 324)

In Section 3.5.4, we saw that TCP waits until it has received three duplicate ACKs before performing a fast retransmit. Why do you think the TCP designers chose not to perform a fast retransmit after the first duplicate ACK for a segment is received?

Answer

The reasoning for not doing the retransmit until the third duplicate seems to be that until that point it's more likely to just be out-of-order delivery and the retransmit isn't really needed.

KR. P37 (Chap 3 page 324)

Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been cor- rectly received by Host B.

a. How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.

b. If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in short- est time interval?

Answer

a)

GBN:
Segments: 9 in total, because it first has to send the five segment and then again send segment 2-5, meaning four more.
ACKs: 8 in total, because it sends four for the first four segments it receives (1, 3-5), and then four again for the next four segments (2-5)

SR:
Segments: 6 in total, because it sends all five first and then sends the one lost segment again.
ACKs: 5 in total, because it only sends an ACK for the segments it has succesfully received.

TCP:
Same number of segments and ACKs as in the case for CR, using the completely same logic. Note that the sequence number would differ from CR.

b)

TCP, since it implements Fast Retransmit and therefore doesn't have to wait for the timeout interval but retransmits instantly if it receives three duplicate ACK's.

KR. P40 (Chap 3 page 325)

Consider Figure 3.58. Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions. In all cases, you should provide a short discussion justifying your answer.

a. Identify the intervals of time when TCP slow start is operating.

b. Identify the intervals of time when TCP congestion avoidance is operating.

c. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

d. After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

e. What is the initial value of ssthresh at the first transmission round?

f. What is the value of ssthresh at the 18th transmission round?

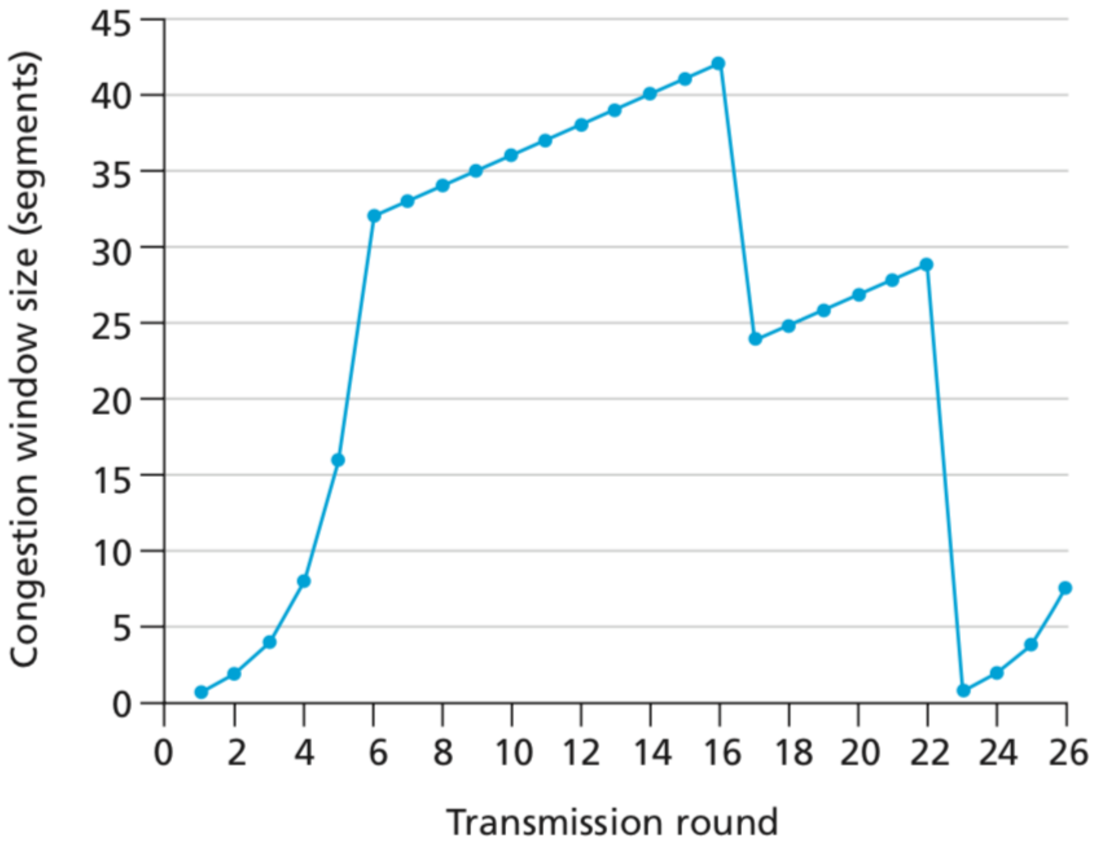
g. What is the value of ssthresh at the 24th transmission round?

h. During what transmission round is the 70th segment sent?

i. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of ssthresh?

j. Suppose TCP Tahoe is used (instead of TCP Reno), and assume that triple duplicate ACKs are received at the 16th round. What are the ssthresh and the congestion window size at the 19th round?

k. Again suppose TCP Tahoe is used, and there is a timeout event at 22nd round. How many packets have been sent out from 17th round till 22nd round, inclusive?



Answer

a)

It is from 0-6 and 23-26 because this is where it grows exponentially.

b)

It is from 6-16 and from 17-22 this is where it grows linearly.

c)

After the 22^{nd} transmission round, packet loss is recognised by a triple duplicate ACK. If there was a timeout, the congestion window size would have dropped to 1.

d)

After the 22^{nd} transmission round, segment loss is detected due to timeout, and hence the congestion window size is set to 1.

e)

It must at be, at congestion window size point 32 because it is here the first linearly groth starts to happen.

f)

We know that the `sshthresh` at the 18^{th} transmission is half of the congestion window, which means it must be 21 in this because the congestion window right before the 18^{th} transmission is 42.

g)

With the same principle as in the task f) we just take the half of the congestion window right before the 24^{th} transmission which is 29 and therefor must the `sshthresh` be 14.5. Because we want hole numbers we take the floor of this number and gets 14 instead.

h)

During the first transmission we see that 1 segment is sent, in the next transmission 2 segments are sent (meaning 1 + 2 in total). This exponential fasion keeps on until 32, and then we start adding linearly.

We therefore keep adding numbers until we get a sum ≥ 70, and then count the amount of numbers we added. We get $1+2+4+8+16+32 = 63 \leq 70$, meaning we aren't done yet at 6 transmission. At the 7^{th} we get $1+2+4+8+16+32+33 \geq 70$, meaning the 70^{th} segment is sent during the 7^{th} transmission.

i)

The `sshthresh` will be set to half of the current value which in this case is 8. Then for good measure to account for the triple duplicate ACK's receive we add 3 to the half of the `sshthresh`. We can conclude that the congestion window is 4 and the `sshthresh` is 7.

j)

`sshthresh` is 21 because its the half of the previous congestion window and congestion window at transmission 19 is 1 because that says the solution online.

k)

There will be sent the following numbers of packets exponentially for each of the transmission (written in "Transmission number: Amount of packets" format):
17: 1
18: 2
19: 4
20: 8
21: 16
22: 21 (fbecause our `sshthresh` is set to this).
The answer will be the sum of all these numbers, which is 52.

KR. P51 (Chap 3 page 328)

Consider the network described in the previous problem. Now suppose that the two TCP connections, C1 and C2, have the same RTT of 100 msec. Suppose that at time t0, C1âĂŹs congestion window size is 15 segments but C2âĂŹs congestion window size is 10 segments.

a. What are their congestion window sizes after 2200 msec?

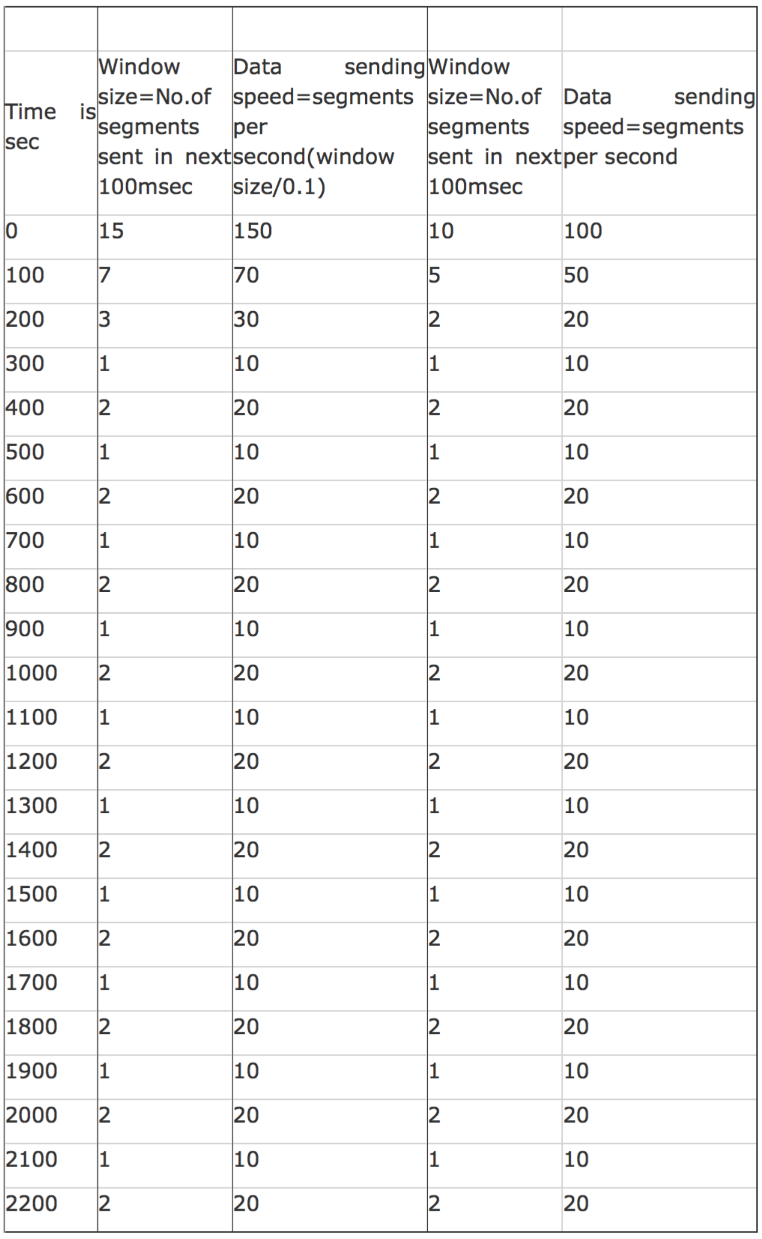
b. In the long run, will these two connections get about the same share of the bandwidth of the congested link?

c. We say that two connections are synchronized, if both connections reach their maximum window sizes at the same time and reach their minimum window sizes at the same time. In the long run, will these two connec- tions get synchronized eventually? If so, what are their maximum window sizes?

d. Will this synchronization help to improve the utilization of the shared link? Why? Sketch some idea to break this synchronization.

Answer

a)



Time is sec	Window size=No.of segments sent in next 100msec	Data sending speed=segments per second(window size/0.1)	Window size=No.of segments sent in next 100msec	Data sending speed=segments per second
0	15	150	10	100
100	7	70	5	50
200	3	30	2	20
300	1	10	1	10
400	2	20	2	20
500	1	10	1	10
600	2	20	2	20
700	1	10	1	10
800	2	20	2	20
900	1	10	1	10
1000	2	20	2	20
1100	1	10	1	10
1200	2	20	2	20
1300	1	10	1	10
1400	2	20	2	20
1500	1	10	1	10
1600	2	20	2	20
1700	1	10	1	10
1800	2	20	2	20
1900	1	10	1	10
2000	2	20	2	20
2100	1	10	1	10
2200	2	20	2	20

b)

Yes this is due to the AIMD algortihm of TCP and that both connections have the same RTT. We can also see in the table in task a that after the third that they have the same congestion window.

c)

Yes this can be seen clearly from the above table Their max window size is 2.

d)

No the synchronisation wont help to improve the utilisation of the shared link. The reason why is that it changes between minimum and maximum the whole time. We will also not use the full potential of the single congested link speed of 30 segments.

KR. P5 (Chap 4 page 394)

Consider a datagram network using 32-bit host addresses. Suppose a router has four links, numbered 0 through 3, and packets are to be forwarded to the link interfaces as follows:

Destination Address Range	Link Interface
11100000 00000000 00000000 00000000 through 11100000 00000000 11111111 11111111	0
11100000 00000001 00000000 00000000 through 11100000 00000001 11111111 11111111	1
11100000 00000010 00000000 00000000 through 11100001 11111111 11111111 11111111	2
otherwise	3

a. Provide a forwarding table that has five entries, uses longest prefix match- ing, and forwards packets to the correct link interfaces.

b. Describe how your forwarding table determines the appropriate link interface for datagrams with destination addresses:

11111000 10010001 01010001 01010101
11100000 00000000 11000011 00111100
11100001 10000000 00010001 01110111

Answer

a)

We can model it this way:

`11100000 00000000:`	`0`
`11100000 00000001:`	`1`
`1110000:`	`2`
`otherwise:`	`3`

This is because we look for the first part where the ends of the address range stops matching each other, and then just write the part prior to that.

b)

For 11111000 10010001 01010001 01010101:
We see that none of the addresses describe this, and then we get to otherwise and thus get 3.

For 11100000 00000000 11000011 00111100:
This will return 0, since it matches that. It also matches 2, but since we use longest prefix matching this interface will be picked.

For 11100001 10000000 00010001 01110111: This will match on 2, since this is the only destination address that matches.

KR. P8 (Chap 4 page 395)

Consider a router that interconnects three subnets: Subnet 1, Subnet 2, and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 223.1.17/24. Also suppose that Subnet 1 is required to support up to 62 interfaces, Subnet 2 is to support up to 106 interfaces, and Subnet 3 is to support up to 15 interfaces. Provide three network addresses (of the form a.b.c.d/x) that satisfy these constraints.

Answer

We see that our adresses with a prefix 223.1.17/24 will look like the following:
<u>11011111 00000001 000010001</u> xxxxxxxx

We now find the different address ranges, where we for example for subnet 1 that has to support 62 addresses will choose to add 64 because it is the first power of 2 above this value:

Subnet 1 suffix (62):

```
0000 0000
through
0011 1111
```

We see that we get:
<u>11011111 00000001 000010001 00</u>xxxxxx

Which yields us 223.1.17.0/26

Subnet 3 suffix (15):

```
0100 0000
through
0101 0000
```

We see that we get:
<u>11011111 00000001 000010001 010</u>xxxxx
Which yields us 223.1.17.64/27

Subnet 2 suffix (106):

```
1000 0000
through
1111 1111
```

We see that we get:
<u>11011111 00000001 000010001 1</u>xxxxxxx
Which yields us 223.1.17.128/25

KR. P14 (Chap 4 page 396)

Consider sending a 1,600-byte datagram into a link that has an MTU of 500 bytes. Suppose the original datagram is stamped with the identification number 291. How many fragments are generated? What are the values in the various fields in the IP datagram(s) generated related to fragmentation?

Answer:

We know, that we can send $MTU - IP \;\; Header = 500 - 20 = 480$ bytes of data each time. The amount of fragments is given by $\lceil \frac{1600}{480} \rceil = \lceil 3.33 \rceil = 4$ fragments. For each fragment, the identification number is incremented by 1. The flag bit signals if it is the last fragment, it is set to 0, while the others are set to 1. This means we get:

Fragment 1:
Identification number: 291
Flag bit: 1

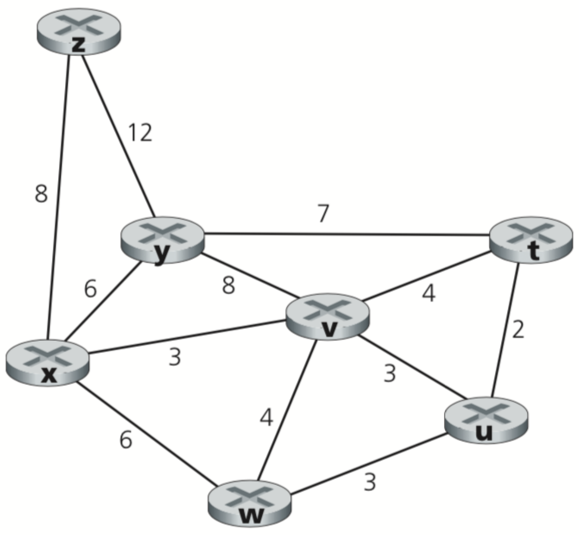
Fragment 2:
Identification number: 292
Flag bit: 1

Fragment 3:
Identification number: 293
Flag bit: 1

Fragment 4:
Identification number: 294
Flag bit: 0

KR. P3 (Chap 5 page 457)

Consider the following network. With the indicated link costs, use DijkstraâĂŹs shortest-path algorithm to compute the shortest path from x to all network nodes. Show how the algorithm works by computing a table similar to Table 5.1.



Answer

Step	N'	z	y	v	w	t	u
0	x	8, x	6, x	3, x	6, x	∞	∞
1	xv	8, x	6, x		6, x	7, v	6, v
2	xvy	8, x			6, x	7, v	6, v
3	xvyw	8, x				7, v	6, v
4	xvywu	8, x				7, v	
5	xvywut	8, x					
6	xvywutz						

KR. P4 (Chap 5 page 457)

Consider the network shown in Problem P3. Using DijkstraâĂŹs algorithm, and showing your work using a table similar to Table 5.1, do the following:

a. Compute the shortest path from t to all network nodes.

b. Compute the shortest path from u to all network nodes.

c. Compute the shortest path from v to all network nodes.

d. Compute the shortest path from w to all network nodes.

e. Compute the shortest path from y to all network nodes.

f. Compute the shortest path from z to all network nodes.

Answer

a)

Computing the shortest path from t to all network nodes.

Step	N'	v	u	w	x	y	z
0	t	4, t	2, t	∞	∞	7, t	∞
1	tu	4, t		5, u	∞	7, t	∞
2	tuv			5, u	7, v	7, t	∞
3	tuvw				7, v	7, t	∞
4	tuvwx					7, t	15, x
5	tuvwxy						15, x
6	tuvwxyz						

b)

Computing the shortest path from u to tall network nodes.

Step	N'	v	t	w	x	y	z
0	u	3, u	2, u	3, u	∞	∞	∞
1	ut	3, u		3, u	∞	9, t	∞
2	utv			3, u	6, v	9, t	∞
3	utvw				6, v	9, t	∞
4	utvwx					9, t	14, x
5	utvwxy						14, x
6	utvwxy						

c)

Computing the shortest path from v to all network nodes.

Step	N'	t	u	w	x	y	z
0	v	4, v	3, v	4, v	3, v	8, v	∞
1	vu	4, v		4, v	3, v	8, v	∞
2	vux	4, v		4, v		8, v	11, x
3	vuxt			4, v		8, v	11, x
4	vuxtw					8, v	11, x
5	vuxtwy						11, x
6	vuxtwyz						

KR. P8 (Chap 5 page 458)

Consider the three-node topology shown in Figure 5.6. Rather than having the link costs shown in Figure 5.6, the link costs are c(x,y) = 3, c(y,z) = 6, c(z,x) = 4. Compute the distance tables after the initialization step and after each iteration of a synchronous version of the distance-vector algorithm (as we did in our earlier discussion of Figure 5.6).

Answer

Node x tables

	x	y	z
x	0	3	4
y	∞	∞	∞
z	∞	∞	∞

	x	y	z
x	0	3	4
y	3	0	6
z	4	6	0

Node y tables

	x	y	z
x	∞	∞	∞
y	3	0	6
z	∞	∞	∞

	x	y	z
x	0	3	4
y	3	0	6
z	4	6	0

Node z tables

	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	4	6	0

	x	y	z
x	0	3	4
y	3	0	6
z	4	6	0

KR. P11 (Chap 5 page 458)

Consider Figure 5.7. Suppose there is another router w, connected to router y and z. The costs of all links are given as follows: c(x,y) = 4, c(x,z) = 50, c(y,w) = 1, c(z,w) = 1, c(y,z) = 3. Suppose that poisoned reverse is used in the distance-vector routing algorithm.

a. When the distance vector routing is stabilized, router w, y,and z inform their distances to x to each other. What distance values do they tell each other?

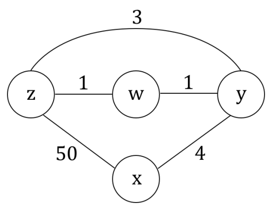
b. Now suppose that the link cost between x and y increases to 60. Will there be a count-to-infinity problem even if poisoned reverse is used? Why or why not? If there is a count-to-infinity problem, then how many iterations are needed for the distance-vector routing to reach a stable state again? Justify your answer.

c. How do you modify c(y,z) such that there is no count-to-infinity problem at all if c(y,x) changes from 4 to 60?

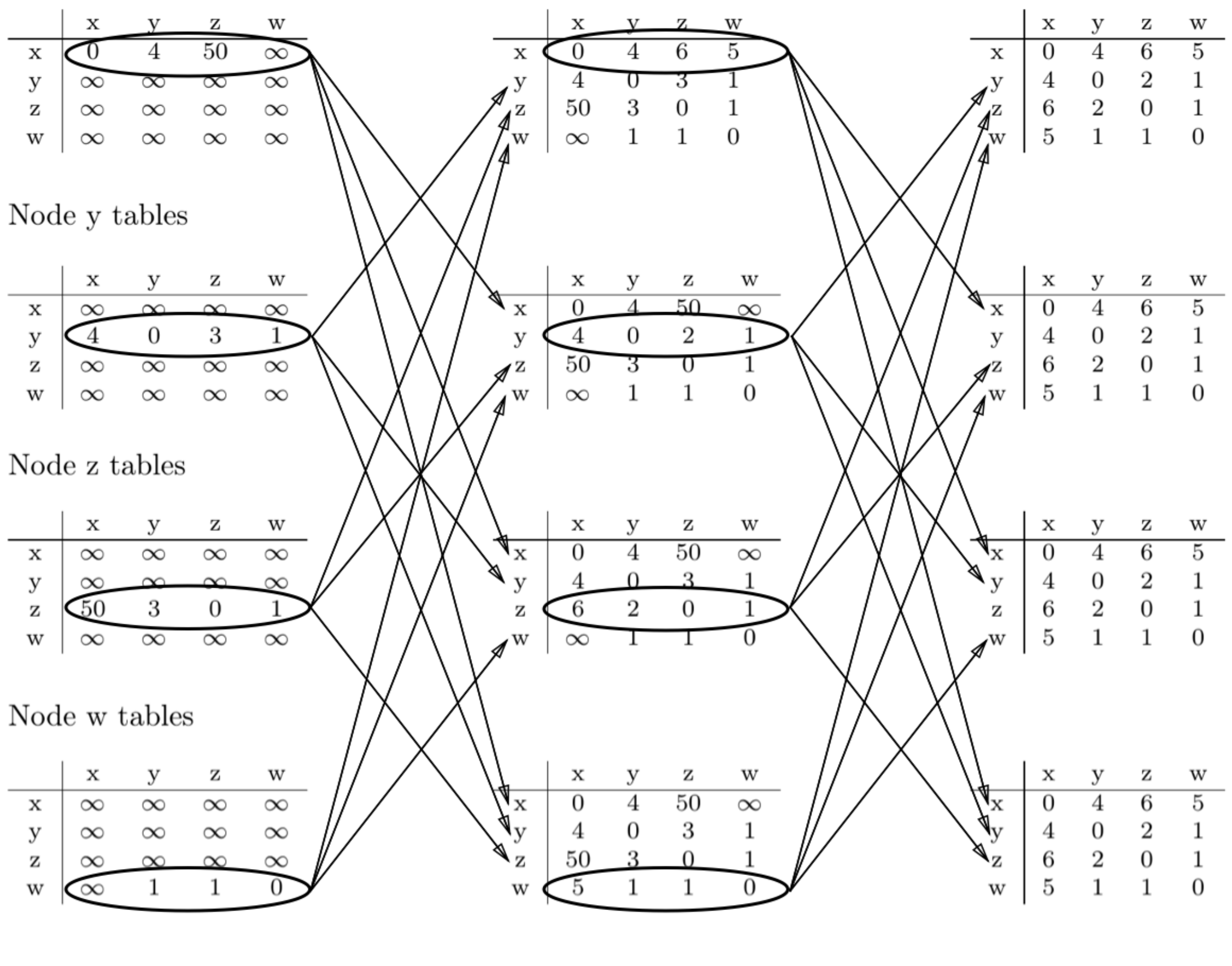
Answer

a)

First we draw the corresponding graph:



This then gives us the following results:



Node x tables

	x	y	z	w
x	0	4	50	∞
y	∞	∞	∞	∞
z	∞	∞	∞	∞
w	∞	∞	∞	∞

	x	y	z	w
x	0	4	6	5
y	4	0	3	1
z	50	3	0	1
w	∞	1	1	0

	x	y	z	w
x	0	4	6	5
y	4	0	2	1
z	6	2	0	1
w	5	1	1	0

Node y tables

	x	y	z	w
x	∞	∞	∞	∞
y	4	0	3	1
z	∞	∞	∞	∞
w	∞	∞	∞	∞

	x	y	z	w
x	0	4	50	∞
y	4	0	2	1
z	50	3	0	1
w	∞	1	1	0

	x	y	z	w
x	0	4	6	5
y	4	0	2	1
z	6	2	0	1
w	5	1	1	0

Node z tables

	x	y	z	w
x	∞	∞	∞	∞
y	∞	∞	∞	∞
z	50	3	0	1
w	∞	∞	∞	∞

	x	y	z	w
x	0	4	50	∞
y	4	0	3	1
z	6	2	0	1
w	∞	1	1	0

	x	y	z	w
x	0	4	6	5
y	4	0	2	1
z	6	2	0	1
w	5	1	1	0

Node w tables

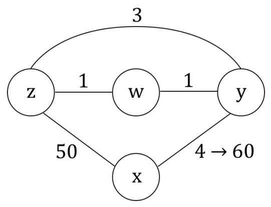
	x	y	z	w
x	∞	∞	∞	∞
y	∞	∞	∞	∞
z	∞	∞	∞	∞
w	∞	1	1	0

	x	y	z	w
x	0	4	50	∞
y	4	0	3	1
z	50	3	0	1
w	5	1	1	0

	x	y	z	w
x	0	4	6	5
y	4	0	2	1
z	6	2	0	1
w	5	1	1	0

b)

The graph now looks like this:



Because our graph involves loops of three or more nodes rather than simply two immediately neighbouring nodes, this will not be detected by the poisoned reverse technique. There will be $50 - 4$ iterations need for the distance-vector routing to reach a stable state again.

c)

cut the connection between z and y.

KR. P1 (Chap 6 page 536)

Suppose the information content of a packet is the bit pattern 1010 0111 0101 1001 and an even parity scheme is being used. What would the value of the field containing the parity bits be for the case of a two-dimensional parity scheme? Your answer should be such that a minimum-length checksum field is used.

Answer

We write our bit pattern up in a 4×4 matrix, and add an extra row and column for the parity bits. Because we are using an even parity scheme, we make sure there are an even number of 1's in the rows and columns. For the last bit in the diagonal, we make sure the sum of the row and column combined is even, meaning we set in a 0 here:

1	0	1	0	0
0	1	1	1	1
0	1	0	1	0
1	0	0	1	0
0	0	0	1	0

KR. P3 (Chap 6 page 537)

Suppose the information portion of a packet contains six bytes consisting of the 8-bit unsigned binary ASCII representation of string "CHKSUM"; compute the Internet checksum for this data.

Answer

We start out finding the bit patterns of the letters:
C = $67_{10} = 01000011_2$
H = $72_{10} = 01001000_2$
K = $75_{10} = 01001011_2$
S = $83_{10} = 01010011_2$
U = $85_{10} = 01010101_2$
M = $77_{10} = 01001101_2$

Because checksum is calculated using 16 bit sequences, we then get:

$$\begin{aligned} &01000011\ 01001000 \quad \text{(CH)} && (1)\\ +\ &01001011\ 01010011 \quad \text{(KS)} && (2)\\ +\ &01010101\ 01001101 \quad \text{(UM)} && (3)\end{aligned}$$

Taking the two first, Eq. (1) and Eq. (2), we get:

$$\begin{aligned} &01000011\ 01001000 \quad \text{('CH')}\\ +\ &01001011\ 01010011 \quad \text{('KS')}\\ =\ &10001110\ 10011011 \quad \text{('CHKS' sum)} && (4)\end{aligned}$$

We add this sum, Eq. (4), to the third part, Eq. (3):

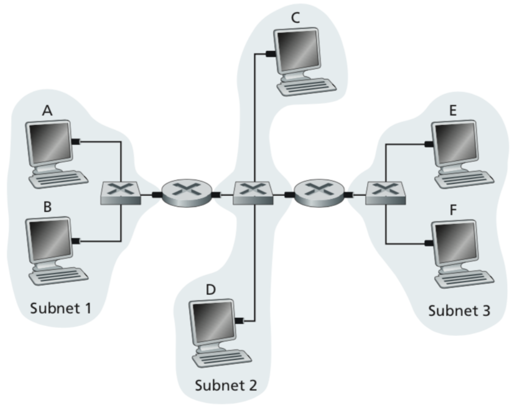
$$\begin{aligned} &10001110\ 10011011 \quad \text{('CHKS' sum)}\\ +\ &01010101\ 01001101 \quad \text{('UM')}\\ =\ &11100011\ 11101000 \quad \text{('CHKSUM' sum)} && (5)\end{aligned}$$

Taking the compliment of this final sum in Eq. (5), we get the checksum which is:

$$\begin{aligned} &\neg 11100011\ 11101000 \quad \text{('CHKSUM' sum)}\\ =\ &00011100\ 00010111 \quad \text{(Checksum value)}\end{aligned}$$

KR. P14 (Chap 6)

Consider three LANs interconnected by two routers, as shown below:



a. Assign IP addresses to all of the interfaces. For Subnet 1 use addresses of the form 192.168.1.xxx; for Subnet 2 uses addresses of the form 192.168.2.xxx; and for Subnet 3 use addresses of the form 192.168.3.xxx.

b. Assign MAC addresses to all of the adapters.

c. Consider sending an IP datagram from Host E to Host B. Suppose all of the ARP tables are up to date. Enumerate all the steps, as done for the single-router example in Section 6.4.1.

d. Repeat (c), now assuming that the ARP table in the sending host is empty (and the other tables are up to date).

Answer:

a) and b)

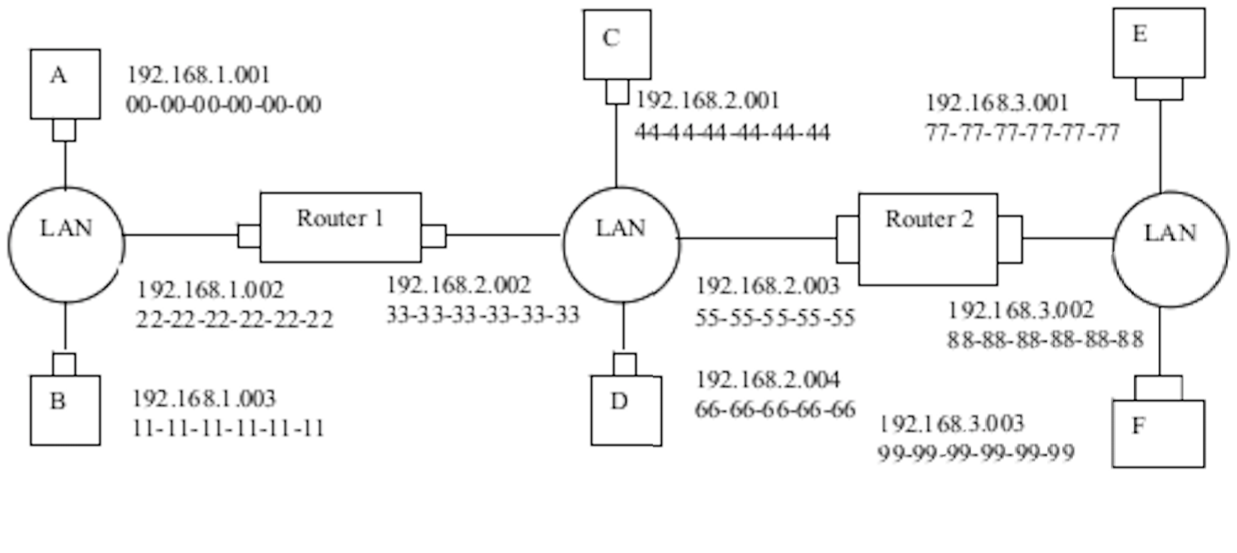


Figure 1: Answer to a and b

c)

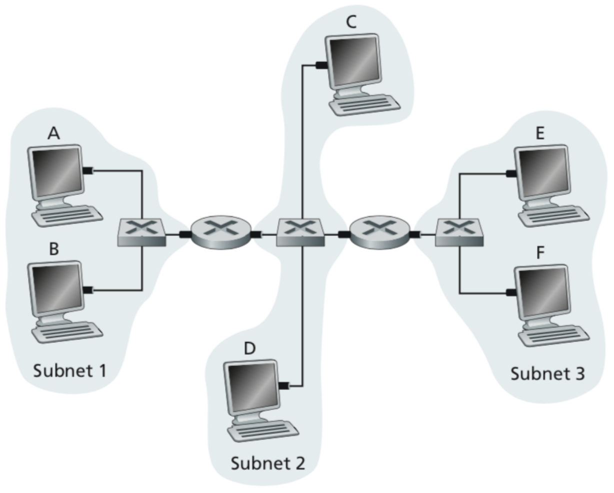
1. Forwarding table in E determines that the datagram should be routed to interface 192.168.3.002
2. The adapter in E creates and Ethernet packet with Ethernet destination address 88-88-88-88-88-88
3. Router 2 receives the packet and extracts the datagram. The forwarding table in this router indicates that the datagram is to be routed to 198.162.2.002
4. Router 2 then sends the Ethernet packet with the destination address of 33-33-33-33-33-33 and source address of 55-55-55-55-55-55 via its interface with IP address of 198.162.2.003
5. The process continues until the packet has reached Host B

d)

ARP in E must now determine the MAC address of 198.162.3.002. Host E sends out an ARP query packet within a broadcast Ethernet frame. Router 2 receives the query packet and sends toHost E an ARP response packet. This ARP response packet is carried by an Ethernet frame withEthernet destination address 77-77-77-77-77-77.

KR. P15 (Chap 6)

Consider the figure from the previous exercise. Now we replace the router between subnets 1 and 2 with a switch S1, and label the router between subnets 2 and 3 as R1.



a. Consider sending an IP datagram from Host E to Host F. Will Host E ask router R1 to help forward the datagram? Why? In the Ethernet frame containing the IP datagram, what are the source and destination IP and MAC addresses?

b. Suppose E would like to send an IP datagram to B, and assume that E's ARP cache does not contain B's MAC address. Will E perform an ARP query to find B's MAC address? Why? In the Ethernet frame (containing the IP datagram destined to B) that is delivered to router R1, what are the source and destination IP and MAC addresses?

c. Suppose Host A would like to send an IP datagram to Host B, and neither A's ARP cache contains B's MAC address nor does B's ARP cache contain A's MAC address. Further suppose that the switch S1's forwarding table contains entries for Host B and router R1 only. Thus, A will broadcast an ARP request message. What actions will switch S1 perform once it receives the ARP request message? Will router R1 also receive this ARP request message? If so, will R1 forward the message to Subnet 3? Once Host B receives this ARP request message, it will send back to Host A an ARP response message. But will it send an ARP query message to ask for A's MAC address? Why? What will switch S1 do once it receives an ARP response message from Host B?

Answer:

a)

No, because the switch can see Host F directly in its own forwarding table.
We then get the following source and dest, based on our assignments in the previous exercise in Figure 1:
Source: IP 192.168.3.1 and MAC 77:77:77:77:77:77.
Dest: IP 192.168.3.3 and MAC 99:99:99:99:99:99.

b)

No, because they are not on the same LAN. Host E can find this out by checking Host B's IP address.
Source: E's IP and MAC Address
Destination: B's IP address and R1's interface connecting to Subnet 3's MAC address.

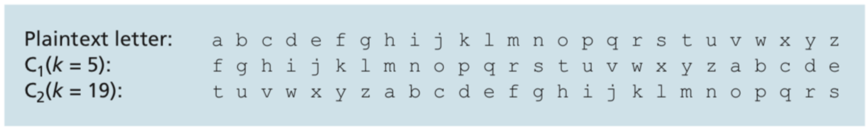
c)

Switch S1 will broadcast the Ethernet frame via both its interfaces as the received ARP frame's destination address is a broadcast address. And it learns that A resides on Subnet 1 which is connected to S1 at the interface connecting to Subnet 1. And, S1 will update its forwarding table to include an entry for Host A.

Yes, router R1 also receives this ARP request message, but R1 won't forward the message to Subnet 3. B won't send ARP query message asking for A's MAC address, as this address can be obtained from A's query message. Once switch S1 receives B's response message, it will add an entry for host B in its forwarding table, and then drop the received frame as destination host A is on the same interface as host B (i.e., A and B are on the same LAN segment).

KR. P3 (Chap 8)

Consider the polyalphabetic system shown in Figure 8.4. Will a chosen plain-text attack that is able to get the plain-text encoding of the message ”The quick brown fox jumps over the lazy dog.” be sufficient to decode all messages? Why or why not?



Plaintext letter:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$C_1(k = 5)$:	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
$C_2(k = 19)$:	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s

Figure 8.4 ♦ A polyalphabetic cipher using two Caesar ciphers

Answer:

The fact that the attacker knows the plain text, the cipher text and every letter in alphabet appears in the phrase, it would be possible for the attacker to break the caesar cipher. We also know thhat the intruder knows the cipher-text character for every plaintext character. The vigenere chipher doesn’t always translate a given plaintext character to the same siphertext character each time. This means that vigenere cipher want be broken right away but would still be possible.

KR. P8 (Chap 8)

Consider RSA with $p = 7$ and $q = 13$.

(a) What are n and z?

(b) Let e be 17. Why is this an acceptable choice for e?

(c) Find d such that $de = 1$ (mod z).

(d) Encrypt the message $m = 9$ using the key (n, e). Let c denote the corresponding ciphertext. Show all work.

Answer:

a)

We use the formulas:

$$\begin{aligned} n &= pq = 7 \cdot 13 = 91 \\ z &= (p-1)(q-1) = 6 \cdot 12 = 72 \end{aligned}$$

b)

Because e has to (1) be less than n and (2) have no common factors (other than 1) with z. We see, that e adheres to both the specifications.

c)

We use the Maple script to calculate d using the formula

$$ed \bmod z = 1$$

We then get $d = 17$.

d)

We know the formula to get the encrypted value c using the key (n, e). We use this to obtain:

$$c = m^e \bmod n = 9^{17} \bmod 91 = 81$$

Extra)

Suppose we received this cipher-text c and also wished to decrypt it. This will require the private key (n, d). The the formula is:

$$m' = c^d \bmod n = 81^{17} \bmod 91 = 9$$

KR. P10 (Chap 8)

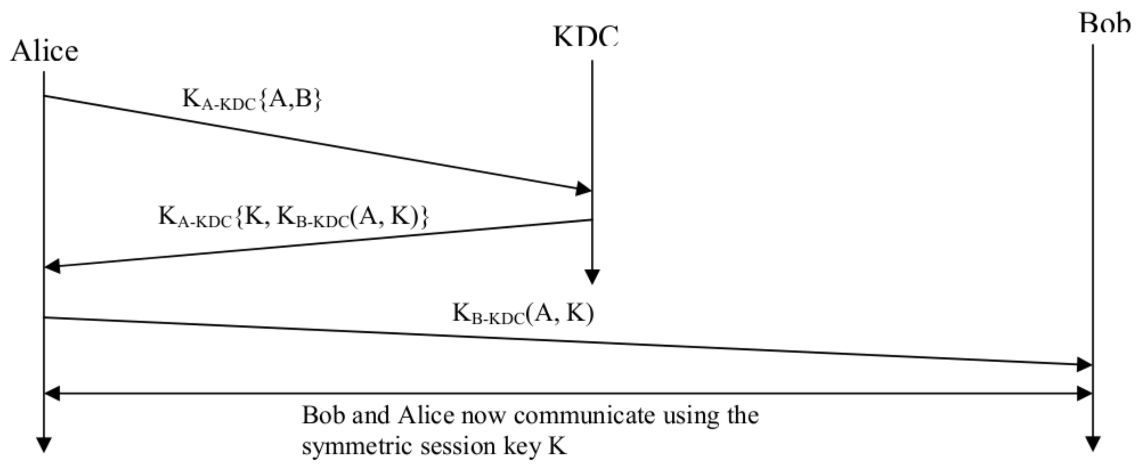
Suppose Alice wants to communicate with Bob using symmetric key cryptography using a session key K_S. In Section 8.2, we learned how public-key cryptography can be used to distribute the session key from Alice to Bob. In this problem, we explore how the session key can be distributed - without public key cryptography - using a key distribution center (KDC). The KDC is a server that shares a unique secret symmetric key with each registered user. For Alice and Bob, denote these keys by K_{A-KDC} and K_{B-KDC}. Design a scheme that uses the KDC to distribute K_S to Alice and Bob. Your scheme should use three messages to distribute the session key: a message from Alice to the KDC; a message from the KDC to Alice; and finally a message from Alice to Bob. The first message is K_{A-KDC} (A, B). Using the notation, K_{A-KDC}, K_{B-KDC}, S, A, and B answer the following questions.

(a) What is the second message?

(b) What is the third message?

Answer

Problem 10



Alice
KDC
Bob
$K_{A\text{-}KDC}\{A,B\}$
$K_{A\text{-}KDC}\{K, K_{B\text{-}KDC}(A, K)\}$
$K_{B\text{-}KDC}(A, K)$
Bob and Alice now communicate using the symmetric session key K

KR. P13 (Chap 8)

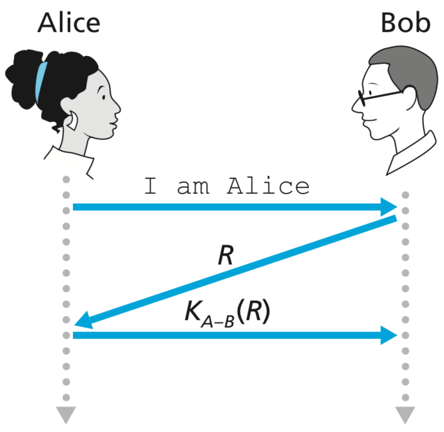
In the BitTorrent P2P file distribution protocol (see Chapter 2), the seed breaks the file into blocks, and the peers redistribute the blocks to each other. Without any protection, an attacker can easily wreak havoc in a torrent by masquerading as a benevolent peer and sending bogus blocks to a small subset of peers in the torrent. These unsuspecting peers then redistribute the bogus blocks to other peers, which in turn redistribute the bogus blocks to even more peers. Thus, it is critical for BitTorrent to have a mechanism that allows a peer to verify the integrity of a block, so that it doesn't redistribute bogus blocks. Assume that when a peer joins a torrent, it initially gets a `.torrent` file from a fully trusted source. Describe a simple scheme that allows peers to verify the integrity of blocks.

Answer

The file is broken into blocks of equal size. For each block, calculate the hash (for example with MD5 or SHA-1). The hashes for all the blocks are saved in the .torrent file. Whenever a peer downloads a block, it calculates the hash of this block and compares it to the hash in the .torrent file. If the two hashes are equal, the block is valid. Otherwise, the block is bogus, and should be discarded.

KR. P15 (Chap 8)

Consider our authentication protocol in Figure 8.18 in which Alice authenticates herself to Bob, which we saw works well (i.e., we found no flaws in it).



Now suppose that while Alice is authenticating herself to Bob, Bob must authenticate himself to Alice. Give a scenario by which Trudy, pretending to be Alice, can now authenticate herself to Bob as Alice. (Hint: Consider that the sequence of operations of the protocol, one with Trudy initiating and one with Bob initiating, can be arbitrarily interleaved. Pay particular attention to the fact that both Bob and Alice will use a nonce, and that if care is not taken, the same nonce can be used maliciously.)

Answer

Bob does not know if he is talking to Trudy or Alice initially. Bob and Alice share a secret key K_{A-b} that is unknown to Trudy. Trudy wants Bob to authenticate her (Trudy) as Alice. Trudy is going to have Bob authenticate himself, and waits for Bob to start:

1. **Bob-to-Trudy: "I am Bob"** Commentary: Bob starts to authenticate himself. Bob's authentication of himself to the other side then stops for a few steps.
2. **Trudy-to-Bob: "I am Alice"** Commentary: Trudy starts to authenticate herself as Alice.
3. **Bob-to-Trudy: "R"** Commentary: Bob responds to step 2 by sending a nonce in reply. Trudy does not yet know $K_{A-b}(R)$ so he can not yet reply.
4. **Trudy-to-Bob: "R"** Commentary: Trudy responds to step 1 now continuing Bob's authentication, picking as the nonce for Bob to encrypt, *the exact same value that Bob sent her to encrypt in step 3.*
5. **Bob-to-Trudy: "K_{A-B}(R)"** Bob completes his own authentication of himself to the other side by encrypting the nonce he has sent in step 4. Trudy now has K_{A-B}(R).

KR. P17 (Chap 8)

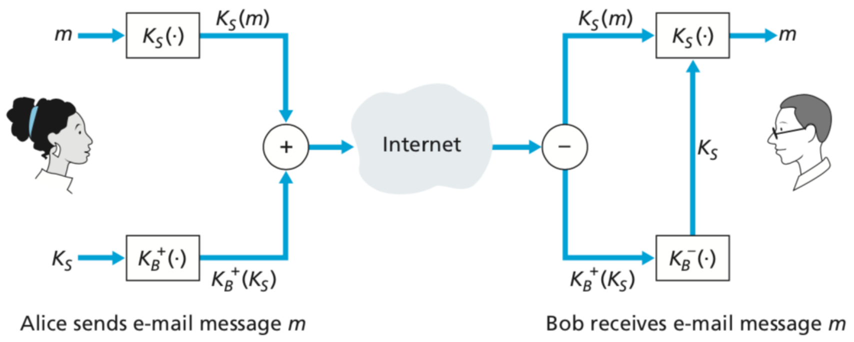


Figure 8.19 ♦ Alice used a symmetric session key, K_S, to send a secret e-mail to Bob

Figure 8.19 shows the operations that Alice must perform with PGP to provide confidentiality, authentication, and integrity. Diagram the corresponding operations that Bob must perform on the package received from Alice.

Answer