**Assignment 13**

**P12**

When the route goes through each autonomous system, it adds number of this AS paths to ASN. That’s why , when each BGP peer gets this route, it notices AS path. If they find their AS number inside of it, it means this route in a loop, because this route came the same BGP peer again.

**P16**

ISP C can force ISP B to hand over the traffic through East Coast for going to ISP D. After this configuration ISP B will use only East Coast to reach to D.

**P20**

X and Y has a peering agreement. That means they can advertise their all routes to each other and give the responsibility to transit these routes. On the other hand, Y and Z has a peering agreement. That means they can also send their all routes to each other and give responsibility to transit these routes. Therefore, if Z receive some traffic from Y that actually comes from X, it should to transmit this traffic because of the peering between Y. There is no policy implementation that BGP allows Z not transfer this data or traffic.

**P22**

The designers of SNMP chose UDP over TCP because UDP is a lightweight, low-overhead, connection-less protocol, making it faster and more efficient for the short, typically one-way SNMP messages. UDP avoids the overhead of establishing and maintaining connections required by TCP as well as TCP’s congestion, allowing SNMP to operate effectively even when parts of the network are down and don’t have slow traffic. While UDP is unreliable, SNMP can tolerate occasional message losses, making UDP a suitable choice.

**Assignment 12**

**P3**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Step** | **N’** | **D(t),p(t)** | **D(u),p(u)** | **D(v),p(v)** | **D(w),p(w)** | **D(y),p(y)** | **D(z),p(z)** |
| **0** | **x** | **infinity** | **infinity** | **3,x** | **6,x** | **6,x** | **8,x** |
| **1** | **xv** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |
| **2** | **xvu** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |
| **3** | **xvuw** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |
| **4** | **xvuwy** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |
| **5** | **xvuwyt** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |
| **6** | **xvuwytz** | **7,v** | **6,v** | **3,x** | **6,x** | **6,x** | **8,x** |

**P5**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **u** | **v** | **x** | **y** | **z** |
| **u** | **0** | **1** | **4** | **2** | **6** |
| **v** | **1** | **0** | **3** | **3** | **5** |
| **x** | **4** | **3** | **0** | **3** | **2** |
| **y** | **2** | **3** | **3** | **0** | **5** |
| **z** | **6** | **5** | **2** | **5** | **0** |

**P9**

**No, because only increasing the cost of a link can cause count-to-infinity. The count-to-infinity problem in distance vector routing will not occur if we decrease the cost of a link because decreasing the cost leads to immediate and efficient updates without causing routing loops or gradual cost increments. Similarly, the count-to-infinity problem will not occur if we connect two nodes that did not previously have a link, as this simply adds a new path with a specific cost, which is straightforwardly updated in the routing tables without creating the conditions for the problem.**

**Assignment 11**

**P11**

In the form a.b.c.d/x, meaning it has the range from 223.1.17.0 - 223.1.17.255.

Subnet 1: the power of 2 that’s greater than 60 is 64

Subnet 2: the power of 2 that’s greater than 90 is 128

Subnet 3: the power of 2 that’s greater than 12 is 16

Therefore, starting from the largest subnet,

Subnet 2 ranging from: 223.1.17.0 - 223.1.17.127

Subnet 1 ranging from: 223.1.17.128 - 223.1.17.191

Subnet 3 ranging from: 223.1.17.192 - 223.1.17.207

**P17**

Each datagram contains 20 bytes of header.

1500-20 = 1480 bytes --> amount of data in each datagram

5000000/1480 = 3378.37837838

Therefore, about 3379 datagrams are needed to transfer an MP3 from source Host A to destination Host B.

**Assignment 10**

**P1**

1. Firstly, if we want each packet (which goes to Host H3) to forward from interface 3, we should configure headers of all these packets according to interface 3. It means that when router checks the longest prefix of these packets’ headers, all of them should match with interface 3. For example, if the prefix of one packet is 1000110 and it should be transmitted to Host 3, that packet should be matched with interface 3 in the forwarding table.

|  |  |  |  |
| --- | --- | --- | --- |
| Forwarding table of router A | | | |
| Prefix | Network Address | Hop | Interface |
| 1000110 | Network Address of H3 | 2 | 3 |

1. No, because there is no interface 4. There is no matching according to source address.

**P4**

1. For finding the minimal number of time slots we should take the datagrams from queues and crossing them from switching fabric to the output ports in the optimal way. When we notice the fronts of input ports we can see there are two Y*Y* datagrams in different input ports. That's why, we should take one of them for the first slot for forwarding output ports, then we can take another one for the next slot. Also, we get X*X* from the first input port. The first slotfirst slot is like this: {X(from the first input),Y (from the second input)}{X(from the first input),Y (from the second input)}. Then, we look at the first datagrams of input ports again. We get X*X* datagram from the second input and Y*Y* from the third input port. The second slotsecond slot is like this: {X(from the second input),Y (from the third input)}{X(from the second input),Y (from the third input)}. At the last stage there is only one Z*Z* datagram in input lines and we get it for the third slotthird slot and send to output port. That's why,the minimal number of time slots is 3.
2. For finding the maximal number of time slots ,for example, we can get YY from not second but third input port and XX from the first input port as well. Then, the first slotfirst slot is like this: {X(from the first input),Y (from the third input)}{X(from the first input),Y (from the third input)}. After the first slot transmits, we can only choose YY from the second input line and ZZ datagram from the third input. The second slotsecond slot is like this: {Y(from the second input),Z (from the third input)}{Y(from the second input),Z (from the third input)}. Finally, there is only XX datagram in the second input port and it will be in the third slot for transmission to output ports. In this situation, there are also 33 slots for transferring datagrams in the crossbar switch fabric

**P6**



|  |  |  |  |
| --- | --- | --- | --- |
| Package | Arrival Time | Departure Time | Delay |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 |
| 3 | 1 | 2 | 1 |
| 4 | 1 | 3 | 2 |
| 5 | 3 | 4 | 2 |
| 6 | 2 | 5 | 2 |
| 7 | 3 | 6 | 3 |
| 8 | 5 | 7 | 2 |
| 9 | 5 | 8 | 3 |
| 10 | 7 | 9 | 2 |
| 11 | 8 | 10 | 2 |
| 12 | 8 | 11 | 3 |
| Average delay: 1.91 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Package | Arrival Time | Departure Time | Delay |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 2 | 2 |
| 3 | 1 | 1 | 0 |
| 4 | 1 | 6 | 5 |
| 5 | 3 | 4 | 1 |
| 6 | 2 | 7 | 5 |
| 7 | 3 | 3 | 0 |
| 8 | 5 | 9 | 4 |
| 9 | 5 | 5 | 0 |
| 10 | 7 | 10 | 3 |
| 11 | 8 | 8 | 0 |
| 12 | 8 | 11 | 3 |
| Average delay: 1.91 | | | |

C)

|  |  |  |  |
| --- | --- | --- | --- |
| Package | Arrival Time | Departure Time | Delay |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 2 | 2 |
| 3 | 1 | 4 | 3 |
| 4 | 1 | 1 | 0 |
| 5 | 3 | 3 | 0 |
| 6 | 2 | 6 | 4 |
| 7 | 3 | 5 | 2 |
| 8 | 5 | 7 | 2 |
| 9 | 5 | 9 | 4 |
| 10 | 7 | 11 | 4 |
| 11 | 8 | 8 | 0 |
| 12 | 8 | 10 | 2 |
| Average delay: 1.91 | | | |

D)

|  |  |  |  |
| --- | --- | --- | --- |
| Package | Arrival Time | Departure Time | Delay |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 2 | 2 |
| 3 | 1 | 1 | 0 |
| 4 | 1 | 5 | 4 |
| 5 | 3 | 3 | 0 |
| 6 | 2 | 7 | 5 |
| 7 | 3 | 4 | 1 |
| 8 | 5 | 9 | 4 |
| 9 | 5 | 6 | 1 |
| 10 | 7 | 10 | 3 |
| 11 | 8 | 8 | 0 |
| 12 | 8 | 11 | 3 |
| Average delay: 1.91 | | | |

**P9**

|  |  |
| --- | --- |
| Destination Address Range | Link Interface |
| 00000000 to 00111111 | 0 |
| 01000000 to 01011111 | 1 |
| 01100000 to 01111111 | 2 |
| 10000000 to 10111111 | 2 |
| 11000000 to 11111111 | 3 |

**No. of addresses for Interface 0: 2^6 = 64**

**No. of addresses for Interface 1: 2^5 = 32**

**No. of addresses for Interface 2: 2^5 + 2^6 = 96**

**No. of addresses for Interface 3: 2^6 = 64**

**Assignment 9**

**P40**

1. From the figure, we can spot the starting curve at the time interval between 1-6 and 23-26
2. Congestion control occurs when window size reaches a certain value the protocol decrease the size to avoid congestion and it occurs at round 16 and round 22.
3. As we can observe from the figure that the window size did not drop to zero and did not experience any slow start. That information lets us know about the nature of loss detection which is the detection of triple ACK.
4. As we can observe from the figure that the window size did drop to zero and did experience a slow start contrary to the first case. That information lets us know about the nature of loss detection which is the timeout.
5. The starting curve stops at window size 33, which is the initial value of sshthresh.
6. Sshthresh is calculated by cwnd/2. cwnd at 18th transmission round is 42, therefore, 42/2 is about 21
7. As we can see from the previous case we can see the congestion window is 29 packets for 24th transmission and using previously defined formula we can assume that the sshtresh is 15 for this problem
8. The number of packets are increasing in each round.
   1. 1st round = 1 total =1
   2. 2nd round = 2 total = 3
   3. 3rd round = 4 total = 7
   4. 4th round = 7 total = 14
   5. 5th round = 16 total = 30
   6. 6th round = 32 total = 62
   7. 7th round = 33 total = 95 Therefore, the 70th segment is within the 7th round.
9. Then, the congestion window size will be 8 and the ssthresh will 8/2 = 4.
10. In TCP Tahoe main idea is a slow start which means that the window size will go immediately to '0'. Therefore, the congestion window size will be 2 and ssthresh will be 42/2 = 21.
11. .
    1. 17th round = 1 packet
    2. 18th round = 2 packets
    3. 19th round = 4 packets
    4. 20th round = 8 packets
    5. 21th round = 16 packets
    6. 22th round = 21 packets
    7. Total: 52 packets

**P44**

1. it takes 6 Round Trip Time to increase from 6 MSS to 12 MSS.
2. Sum of MSS / Sum of RTT:
   1. Sum of MSS = 7+8+9+10+11+12 = 57
   2. Sum of RTT = 1+1+1+1+1+1 = 6
   3. 57 / 6 = 9.5

**Assignment 8**

**P26**

1. We need to find a maximum size of L (length of file) so the sequence numbers can be used efficiently, for finding this value we need to calculate the sequence number limitation. Since we know the size of the sequence number is 4 bytes, we can calculate a max value for it:

4 bytes = 2^32 = 4,294,967,296

Therefore, there are 4,294,967,296 different sequence numbers.

1. First, we need to find the numbers of segments:

2^32/536 = 812999 segments

66bytes are added to each segment.

812999 x 66 = 528857913 bytes are added.

528857913 + 2^32 = 4.8 X 10^9 bytes

To find the transmission time:

4.8x10^9 / 115 Mbps = 250 seconds.

**P27**

1. Sequence number of the second segment can be calculated by adding the size of the first segment to the sequence number of the first segment because each byte corresponds to a sequence number. Since they are using the same channel source and destination ports will not change.

Sequence number: 127 + 80 = 207

Source port number: 302

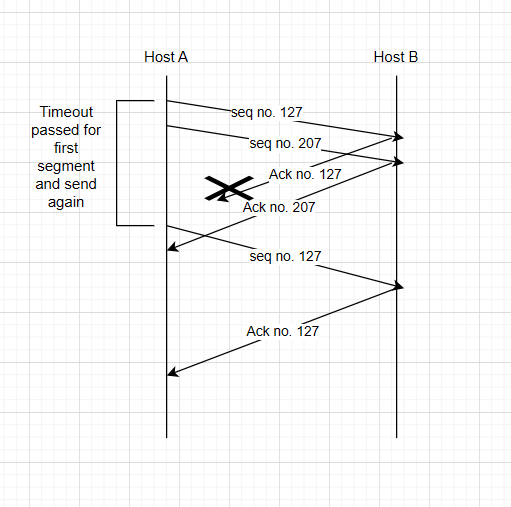
Destination port number: 80

1. The acknowledgement messages are sent over the same channel in the reverse direction of the message which will change the source and destination port addresses, ACK number will be the same as the sequence number of the first segment.

Acknowledge number: 127

Source port number: 80

Destination port number: 302

1. The acknowledgment number in this acknowledgment will be the next expected byte, which is 127.
2. 

**Assignment 7**

**Chapter 3**

**P3**

**Sum of the three 8bits bytes:** 01010011

01100110

01110100

1 00101101

Since we have 1 extra bit we need to wrap it around and we get: 00101110

- **In the second step,** we swap the bits with their complements and we get**: 11010001**

- UDP uses the 1s complement of the sum for its checksum calculation to reliably detect errors in data transmission while maintaining computational simplicity and efficiency.

-The receiver after getting the check sum adds up all the bytes if all 1s acquired then there is not an error in the sum. On the other hand if there are 1 or more 0s then there is an error.

- Using checksum method we can detect 1 bit error, however some cases 2 bit errors pass through undetected.

**P4**

1. Adding 01011100 and 01100101 will get 11000001. Therefore, the 1s complement of sum is 00111110.
2. Adding 11011010 and 01100101 will get 01000000. Therefore, the 1s complement of sum is 10111111.
3. Adding 01011101 and 01100100 will get 11000001.

**Assignment 6**

**P26**

1. Yes it is possible, as long as there is a peer who is willing to send the required file to Bob and Bob has enough space to receive the file.
2. It is possible. He must make sure that all the computers have distinct IP address and act as an independent participant in the swarm.

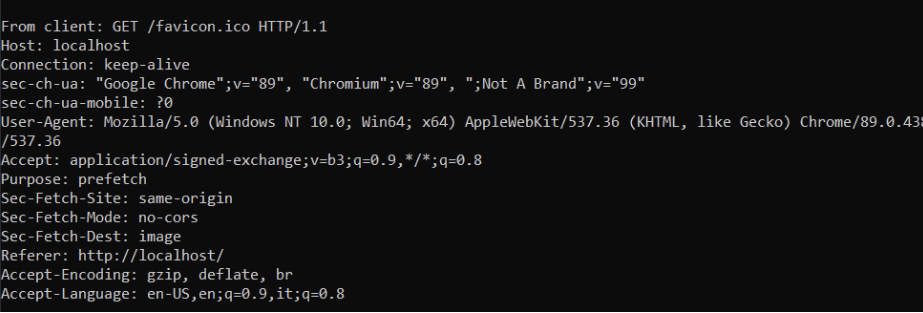
**P28**

1. There will be no connection. This is because client will search for the server which doesn’t exist. Therefore, TCP connection will not be established.
2. Unlike TCP, UDP connection will be established. UDP Client doesn’t need to activate the server before client.
3. No connection will occur due to port number mismatch. An error will occur in client side for TCP connection, and the server will not receive the request from the client side for UDP connection.

**Assignment 5**

**Chapter 2**

**P12**



**P15**

Mail From: is for indicating the sending and return path of the message, while From: is a header field that identifies the author or sender of the mail.

**P20**

We can use the command ipconfig/displaydns that will print out all the web servers requested by the users in the department. Then, we can analyze from that.

**Assignment 2**

**Problem 8:**

1. 10Mbps / 200kbps = 50
2. 1/50 x 0.1 = 0.002
3. P[n users are transmitting at the same time]=1−Sum[ Binomial(120, i)×0.1^i×0.9^120-i ,i=0,n]
4. Using the above binomial formula

120∑n=51 P[n users are transmitting at the same time]

**Problem 25:**

1. 20000000m/2.5 x 10^8 m/sec = 0.08 sec

R x d = 5 x 0.08 = 400000 bits

1. 800000 bits / 5Mbps = 0.16sec

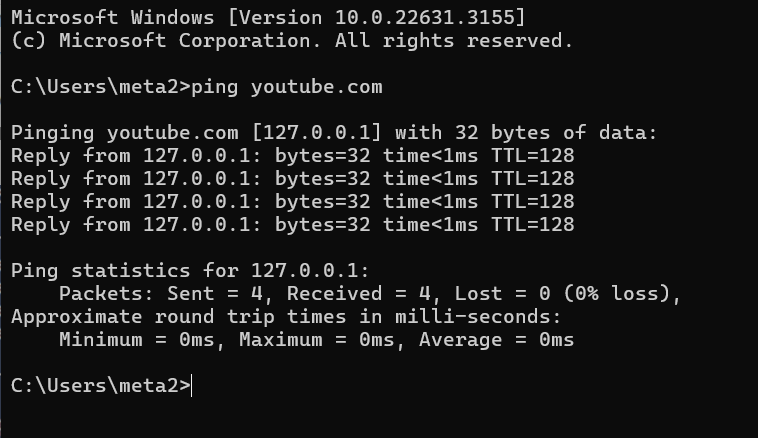
dtrans > dprops, therefore, 400000 bits will be in the link at any given time.

1. Bandwidth-delay product shows us the bit capacity of the link.
2. 20000km / 400000 bits = 50 m , No it is not longer.
3. (m x s)/R

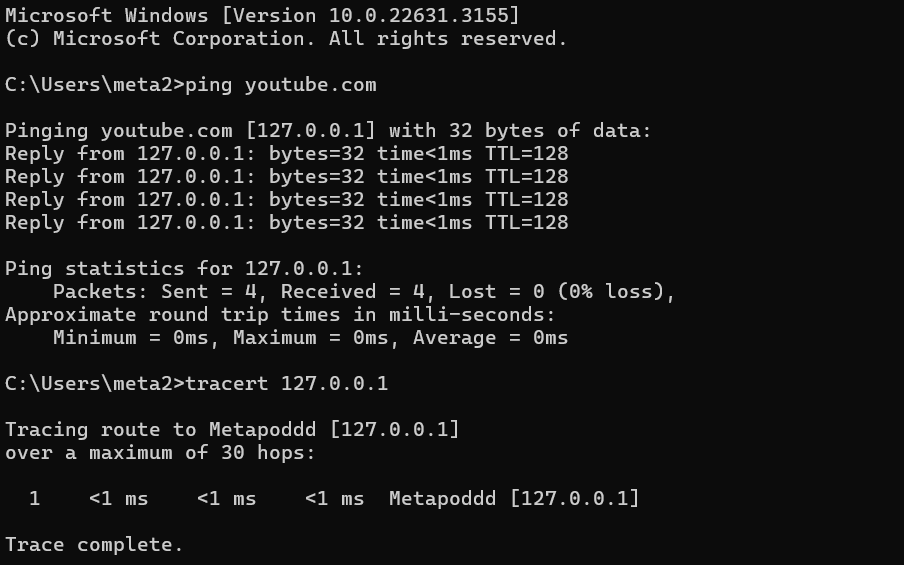
**What is the difference between Abstraction and Encapsulation?**

Abstraction refers to the concept of hiding the underlying complexities of the network infrastructure and presenting a simplified view to the users or higher-level applications. Encapsulation, on the other hand, involves packaging data and protocols into units (such as packets) for transmission over the network, providing a protective layer and ensuring reliable delivery.

1. Ping another computer.



1. Tracert a server.



1. Download lab resource.