## CÂU 3/Difference Between HDLC and PPP

Definition

HDLC is a bit-oriented code-transparent synchronous data link layer protocol developed by ISO. But, PPP is a data link layer communication protocol used to establish a direct connection between two nodes.

Stands for

HDLC stands for High-Level Data Link Control, while PPP stands for Point to Point.

Protocol Type

HDLC is a bit-oriented protocol, whereas PPP is a byte-oriented protocol. Thus, this is the main difference between HDLC and PPP.

Communication

Moreover, another important difference between HDLC and PPP is that HDLC supports point to point and multipoint links, whereas PPP only supports point to point links.

Authentication

Furthermore, there is no authentication mechanism in HDLC, while there is authentication in PPP. Hence, this is also a difference between HDLC and PPP.

Compatibility

Besides, HDLC cannot be used with non-cisco devices, while PPP can be used with non-cisco devices.

Conclusion

The main difference between HDLC and PPP is that HDLC is a bit-oriented protocol for communication over a point to point and multipoint links while PPP is a byte-oriented protocol for communication over a point to point links. In brief, HDLC is bit oriented while PPP is byte oriented.

CÂU 4/The round trip propagation delay is:  
  
(375 × 10 6 m) 2tprop = 2 = 2.50 sec 3 × 10 8 m / s  
To allow for continuous transmission, we must use Go-Back-N or Selective Repeat. Go-Back-N: If N = 7 à  
  
7n f 1.5Mbps 127n f  
  
= 2.5s à nf = 535 715 bits  
  
If N = 127   
  
1.5Mbps  
  
= 2.5s nf = 29 528 bits  
  
Selective Repeat: If N = 4 à  
  
4n f 1.5Mbps 64n f  
  
= 2.5s à nf = 973 500 bits  
  
If N = 64   
  
1.5Mbps  
  
= 2.5s à nf = 58 594 bits

CÂU 1/Solution: In an acknowledged connectionless network, reliable delivery can be achieved through the use of ACK and NAK transmissions. Such protocols are suited for communication over networks in which higher layers are sensitive to loss and the underlying network is inherently unreliable with a significant probability of loss or error. Unacknowledged networks provide simpler and faster communication for networks that are inherently reliable or provide service to higher layers that can tolerate information loss

CÂU 2 / Solution: The use of acknowledgments can provide reliable transfer over networks that are prone to error and loss. In connection oriented networks, every packet in a data flow travels on the same path through the network and the proper ordering of packets is guaranteed. In such networks, if a packet arrives out of order, the receiver immediately knows that a packet has been lost. In a connectionless network, the service needs a mechanism for dealing with unordered delivery of information. This is especially important for real-time or delay-sensitive traffic, which may require immediate retransmission and may not be able to use buffering to correct unordered packet arrivals

CÂU 5/Solution: R = 1.5 Mbps, and nf =250 bytes or 2000 bits (250 x 8). The distance that the information must travel is the earth-to-satellite distance, or d ≈ 36,000 km. The speed of light c is 3 x 108. We can calculate the propagation delay and processing rate as follows: tprop = d/c = 36 x 106 / 3 x 108 = 120 ms tf = nf/R = 2000/1.5 x 106 = 1.33 ms We can use either Go-Back-N or Selective Repeat ARQ. The default window size is N = 7 (with a 3bit sequence number). 0 1 2 3 4 5 6 7 …  
  
ACK0  
  
tprop  
  
tf  
  
tcycle  
  
The maximum information rate is achieved with no error, and hence, no retransmission. tcycle = minimum time to transmit a group of N packets = tf + 2 tprop = 1.33 + 2x2120 = 241.33 ms n = no. of bits transmitted in a cycle = N.nf = 7x2000 = 14,000 bits Rmax = no. of bits sent in a cycle / minimum cycle time = n/tcycle = 58 kbps .If the extended sequence numbering option (7-bit) is used, the maximum send window size would be N = 27 – 1 = 127, and hence, the maximum information rate is: Rmax = N.nf / tcycle = 127x2000/(241.33x10-3) = 1.052 Mbps

Câu 6/Solution: The probability that there are k packet arrivals in a T-second period is given by the binomial distribution with parameters N = 60 and p = 0.1. The average number of arrivals is Np = 6. The average number of arrivals that get transferred to the first line is given by:  
  
 60  k  (0.1) k (0.9) 60− k = 4.59 ∑   k =0  k   
8  
  
The remainder of the packet arrivals are sent to the second line, so the average number sent to line 2 is 6 – 4.59 = 1.41 packets per T-second period

CÂU 7 / follow questions:

a. Discuss which of the following adaptation functions are relevant to meeting the requirements of this transfer: handling of arbitrary message size; reliability and sequencing; pacing and flow control; timing; addressing; and privacy, integrity and authentication.  
  
Message size is important because in real-time signals of voice it is necessary to transfer fixed packet size of that holds no more than 20 ms of speech signal. The handling of arbitrary message size is not as important as long as the desired packet size for voice can be handled.  
  
Sequencing is important because each packet needs to arrive in the same sequence that it was generated. Reliability is moderately important since voice transmission can tolerate a certain level of loss and error. Pacing and flow control are not as important because the synchronous nature of the voice signal implies that the end systems will be matched in speed. Timing, for real-time voice transfer is important because this adaptation function helps to control the jitter in the delivered signal. Addressing is only during the connection setup phase if we assume some for of virtual circuit packet switching method. Privacy, integrity, and authentication have traditionally not been as important as the other issues discussed above.

b. Compare a hop-by-hop approach to an end-to-end approach to meeting the requirements of the voice signal.  
  
If the underlying network is reliable then the end-to-end approach is better because the probability of error is very low so processing at the edge suffices to provide acceptable performance. If the underlying network is unreliable then the hop-by-hop approach may be required. For example if the probability of error is very high, as in a wireless channel, then error recovery at each hop may be necessary to make effective communication possible

CÂU 8/Follow questions:

a. Show that the protocol still operates correctly.  
  
The protocol will operate correctly because there is only one variation from the protocol in the chapter, namely, the sender will retransmit before the time-out.

b. Does the state transition diagram need to be modified to describe the new operation?  
  
The state transition diagram remains the same.

c. What is the main effect of introducing the immediate-retransmission feature?  
  
The main effect is that the expected time for transmission is reduced because when the error is detected a NAK is send and the sender can stop the transmission and initiate the retransmission of the frame. If the error is in the ACK then the sender will not have to wait for the time out. Always when there is an error in the ACK or NAK the last frame sent has to be retransmitted because the sender does not know if the frame was received with or without errors

CÂU 9/ Develop an approach that allows the peer processes to exchange messages of arbitrary size.  
  
To exchange messages of arbitrary size, large messages must be segmented into parts of M-H bytes each in length to be transmitted in multiple PDUs. Small messages must be placed in a single PDU.

The essential control information needs to be exchanged between the peer processes is :   
  
The peer processes need to communicate information that allows for the reassembly of messages at the receiver. For example, the first PDU may contain the message length. The last PDU may contain and end-of-message marker. Sequence numbers may also be useful to detect loss in connection oriented networks and to help in reconstruction of the messages in connectionless networks. Lastly, since variable size PDUs are permitted, the size of the PDU must be transmitted in the PDU header.

suppose that the message transfer service provided by the peer processes is shared by several message source-destination pairs. additional control information is required:

In this case, in addition to all of the header information mentioned in below, each PDU must be labeled with a stream ID, so that the receiver can treat each stream independently when reassembling messages

CÂU 10/a. What is the probability that the entire file is transmitted without errors? Note for n large and p very small, (1 − p)n ≈ e−np.  
  
P[no error in the entire file] = (1 – p)n ≈ e–np , for n >> 1, p << 1 = e-8 = 3.35 x 10-4 We conclude that it is extremely unlikely that the file will arrive error free.

b. The file is broken up into N equal-sized blocks that are transmitted separately. What is the probability that all the blocks arrive correctly without error? Does dividing the file into blocks help?  
  
A subblock of length n/N is received without error with probability: P[no error in subblock] = (1 – p)n/N A block has no errors if all subblocks have no errors, so P[no error in block] = P[no errors in subblock]N =((1 – p)n/N)N = (1 – p)n So simply dividing the blocks does not help.

c. Suppose the propagation delay is negligible, explain how Stop-and-Wait ARQ can help deliver the file in error-free form. On the average how long does it take to deliver the file if the ARQ transmits the entire file each time?  
  
Refer to the following figure for the discussion. tf tACK  
  
ACK/NAK  
  
We assume the following: • • • • • • t0 = basic time to send a frame and receive the ACK/NAK ttotal = total transmission time until success nf = number of bits/frame na = number of bits per ACK nt = number of transmissions Pf = probability of frame transmission error t0 = tf + tACK = nf /R + na /R (tprop ≈ 0).  
  
P[nt = i ] = P[one success after i – 1 failure] = (1 – Pf) Pf i – 1 ttotal | i transmissions = i.t0 E[ttotal i −1 2 ] = ∑ it 0 P[nt = i ] = t0 ( − Pf )∑ iPf 1 = t 0 (1 − Pf ) / (1 − Pf ) =t 0 / (1 − Pf ) ∞ ∞ i =1 i =1  
  
Here, nf = n >> na thus t0 ≈ tf = n/R ; and Pf = 1 – P[ no error] = 1 – e–np E[total] = n/R (1 – Pf) = n/[Re–np] = 8 / (3.35 x 10–4) = 23,847 seconds = 6.62 hours! The file gets through, but only after many retransmissions.

d. Now consider breaking up the file into N blocks. (Neglect the overhead for the header and CRC bits.) On the average how long does it take to deliver the file if the ARQ transmits the blocks one at a time? Evaluate your answer for N = 80, 800, and 8000.  
  
For 1 block Pf = 1 – Pb = 1 – (1 – p) n/N and nf = n/N if tprop ≈ 0 and na << n/N : t0b = nf/R = n/NR Tb = E[ttotalb] = t0b / (1 – Pf) = n(1 – p)–n/N /NR average time to transmit one block T = E[ttotal] = N Tb = n(1 – p)–n/N /R = 8 (1 – p) –n/N = 8 enp/N if n/N >> 1, p << 1 • • • N = 80 ⇒ T ≈ 8 e0.1 = 8.84 sec N = 800 ⇒ T ≈ 8 e0.01 = 8.08 sec N = 8000 ⇒ T ≈ 8 e0.001 = 8.008 sec  
  
Each subblock has a higher probability of arriving without errors, and so requires fewer retransmissions to deliver error free. The overall delay is reduced dramatically.

e. Explain qualitatively what happens to the answer in part (d) when the overhead is taken into account.  
  
As N increases, the effect of overhead becomes more significant because the headers constitute a bigger fraction of each subblock

CÂU 15/follow questions:

a. How does the protocol need to be modified to accommodate this change?  
  
First, the frame header needs to be modified to accommodate the list of frames to receive. It can be a fixed or a variable number of slots. NAK won’t be necessary because the receiver explicitly indicates which frames need to be transmitted.

b. What is the effect of the change on protocol performance?  
  
The performance will increase in cases of multiple errors or in cases where the delay is high. A single frame can ask for the retransmission of several frames. The drawback is the overhead in the header and the increased protocol complexity relative to pure Selective-Repeat ARQ.

CÂU 12/Calendar

Description automatically generated

A picture containing text, whiteboard

Description automatically generated

CÂU 16/a. How long does it take to download the file over a 32 kilobit/second modem? T32k = 8 (1024) (1024) / 32000 = 262.144 seconds

b. How long does it take to take to download the file over a 1 megabit/second modem? T1M = 8 (1024) (1024) bits / 1x106 bits/sec = 8.38 seconds

c. Suppose data compression is applied to the text file. How much do the transmission times in parts (a) and (b) change? If we assume a maximum compression ratio of 1:6, then we have the following times for the 32 kilobit and 1 megabit lines respectively: T32k = 8 (1024) (1024) / (32000 x 6) = 43.69 sec T1M = 8 (1024) (1024) / (1x106 x 6) = 1.4 sec

Timeline

Description automatically generated

CÂU 17/Graphical user interface, application

Description automatically generated

CÂU 18/Ans: (a) 135.46.63.10  
Taking the first 22 bits of 135.46.63.10 as network address, we have 135.46.60.0.  
The bit pattern of 135.46.63.10 is 10000111.00101110.00111111.00001010  
When we perform the bit and operation with 22 leading bit 1s and 10 bit 0s, it is equivalent of making the last 10 bit zero. We get the following network address bit pattern: 10000111.00101110.00111100.00000000. The first two bytes are not changed. The 3rd type changes from 63 to 60 while the 4th byte become zero.  
Match with network address in the routing table. The 2rd row matches. The router will forward the packet to Interface 1.

b) 135.46.57.14  
Taking the first 22 bits of the above IP address as network address, we have 135.45.56.0. It matches the network address of the first row. The packet will be forwarded to Interface 0.

CÂU 19/ you are starting with 198.16.0.0 Now company A requests 4000 IPs. You have to round this number (and any later request too) up to a power of two - 4096.

Now let's think: An IP-address consists of 32 Bit. Some of them are network Bits and some are host Bits (if this is new for you read [this](https://support.microsoft.com/en-us/help/164015/understanding-tcp-ip-addressing-and-subnetting-basics)). How many host Bits do you need, so company A gets 4096 hosts? Right: log²(4096) = 12. So the remaining 20 Bits (32-12) are reserved for the network. Now we already have the start IP-address for company A and also the subnet mask: 198.16.0.0/20 (again if this confuses you read the link above). Now we need to determine the last IP-address for company A. To do so we look at our IP and subnet-mask in binary (consider "|" the boundary between network part and host part of the ip-address):

Start-IP (A):

11000110.00010000.0000|0000.00000000

Subnet-Mask(/20):

11111111.11111111.1111|0000.00000000

As you can see, now you have 12 Bits for your hosts, all of them are free to change so you have all in all 2^12 possibilities = 4096.. The last available IP would be the one, where all hosts are equal to 1:

End-IP: 11000110.00010000.0000|1111.11111111 In dezimal this would look like: 198.16.15.255

Now let's look at company B: it requests 2000 - we round it up to the power of two: 2048

log²(2048)= 11 host-bits = 21 network-bits = /21

Now remember the last IP we assigned to A was:

End-IP: 11000110.00010000.00001111.11111111

So the next available one should be bigger by at least 1 right?

11000110.00010000.00001111.11111111 +1 = 11000110.00010000.00010000.00000000 = 198.16.16.0  We should be able to use this one as the start IP for company B

Lets try this out with our netmask of /21:

Start-IP (B):

11000110.00010000.00010|000.00000000

Subnet-Mask(/21):

11111111.11111111.11111|000.00000000

Seems to be working! We have 11 host Bits so we can connect 2^11 hosts =2048. The last usable IP-address however will be the one where all host bits are equal to one:

11000110.00010000.00010|111.11111111 = 198.16.23.255

Now with company C things start to get complicated, please bear with me: The problem here is, that they require more IPs then their predecessor. But we will handle it anyway :)

Company C wants 4000 IPs we round it up to 4096 and calculate the number of hosts bits as before: hosts=20.

Now we will try the same approach as before, so that you can see the problem:

We take the last assigned IP-address and increment it by one:

11000110.00010000.00010111.11111111 + 1=

11000110.00010000.00011000.00000000

Now let us apply our netmask of /20 on this IP:

Start-IP (B):

11000110.00010000.0001|1000.00000000

Subnet-Mask(/21):

11111111.11111111.1111|0000.00000000

Now you should see the problem:

We have have a 1 in our host-bit section. Therefore we can not use all 12 Bits but only 11 which would give us only 2^11 =2018 hosts.

So you might wonder: What if we just turn the 1 in the host part to a 0? Well in that case you would completely collide with the IP-addresses you provided to company B. So this is not an option. The only way is to "push" the 1 in to the network-bit section by applying the increment to the network section only, so that instead of:

11000110.00010000.0001|1000.00000000

you get (incrementing network part only: 11000110.00010000.0001 + 1)

11000110.00010000.0010|0000.00000000

which is equal to 198.16.32.0

the last ip would be again, where all host-bits are equal to 1 so

198.16.47.255

Now you can do the same thing with company D :)

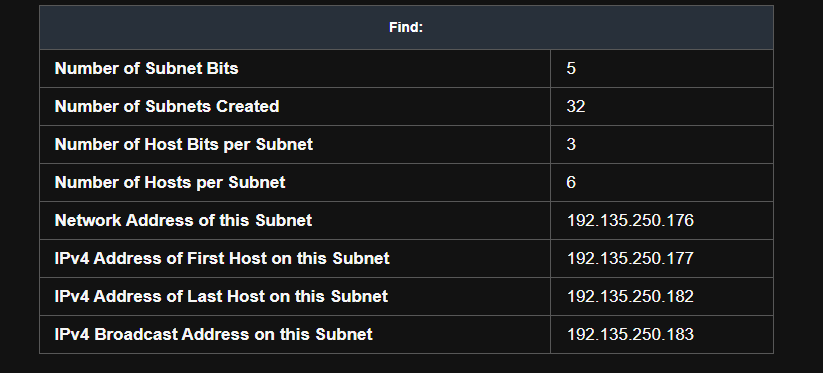
The results should be:

A: 198.16.0.0 – 198.16.15.255 written as 198.16.0.0/20

B: 198.16.16.0 – 198.16.23.255 written as 198.16.16.0/21

C: 198.16.32.0 – 198.16.47.255 written as 198.16.32.0/20

D: 198.16.64.0 – 198.16.95.255 written as 198.16.64.0/19

CÂU 20/

CÂU 14/How many host addresses are needed in the largest required subnet?  
***50***  
  
What is the minimum number of subnets required?  
***The requirements stated above specify two company networks plus two additional networks for future expansion. So, the answer is a minimum of four networks.***  
  
The network that you are tasked to subnet is 192.168.0.0/24. What is the /24 subnet mask in binary?  
***1111111.11111111.11111111.00000000***

he subnet mask is made up of two portions, the network portion, and the host portion. This is represented in the binary by the ones and the zeros in the subnet mask.  
  
In the network mask, what do the ones represent?  
**The ones represent the network portion.**  
  
In the network mask, what do the zeros represent?  
**The zeroes represent the host portion.**

A screenshot of a computer

Description automatically generated with medium confidence

CÂU 11/Based on the topology, how many subnets are needed?  
***5 Four for the LANs, and one for the link between the routers.***

1. How many bits must be borrowed to support the number of subnets in the topology table?  
   ***3***

1. How many subnets does this create?  
   ***8***

1. How many usable hosts does this create per subnet?  
   ***30***

CÂU 13/Dijkstras Algorithm:

Starting vertex is 4.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 |  |
|  | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |  |
| {4} | 5 | 1 | 2 | 0 | 3 | ∞ |  |
| {4,2} | 4 | 1 | 2 | 0 | 3 | ∞ |  |
| {4,2,3} | 4 | 1 | 2 | 0 | 3 | 3 |  |
| {4,2,3,5} | 4 | 1 | 2 | 0 | 3 | 3 |  |
| {4,2,3,5,6} | 4 | 1 | 2 | 0 | 3 | 3 |  |
| {4,2,3,5,6,1} | 4 | 1 | 2 | 0 | 3 | 3 |  |

So the last row value is the shortest path from node 4 to all other nodes.

Each time it updates the path using relaxation:

if(dist(u)+cost(u,v)<dist(v)

   then update the dist(v) =dist(u)+cost(u,v)

dist(u) is distance of vertex u from starting vertex

ccost(u,v) is cost/weight of the edge uv

dist(v) is distance of vertex u from starting vertex