[WR] Workshop 3

Topic	
Book chapter	
Covered in	
Date	
Main topic	Network layer
Source	
URL	

Question 1 - IP subnets

❓ Are the two IP addresses 129.127.8.8 and 129.127.104.8 in the same subnet? Explain your answer.

Are the two IP addresses 129.127.8.8/24 and 129.127.104.8/24 in the same subnet? Explain your answer.

1. **129.127.8.8 and 129.127.104.8**

The first pair of IP addresses is using class-full addressing. Because they starts with number 129, they are using class B addressing, meaning the first 16 bits (first 2 numbers in decimal) are used for network part.

Because both of the IP addresses start with 129.127 in the network part of the IP address, they are from the same subnet.

2. **129.127.8.8/24 and 129.127.104.8/24**

These IP addresses are using Classless InterDomain Routing or (CIDR). The first 24 bits are allocated for the network part.

Converting to binary:

Decimal IP	Binary IP
129.127.8.8	**10000001 01111111 00001000** 00001000
129.127.104.8	**10000001 01111111 01101000** 00001000

By comparing the first 24 bits of the 2 IP addresses (highlighted in orange above), we realised that they don't match. Therefore, these IP addresses are not from the same subnet

Question 2 - IP fragmentation (covered only in the workshops)

❓ See Section 4.3.2 in Seventh Edition (You can access the ebook via the Course Readings on the Course Page - left hand menu item)

A IPv4 router that has an incoming datagram of 1,600 bytes to send into an outgoing link that has an MTU of 500 bytes. The in-coming datagram is stamped with the identification number 291.

How many fragments will be generated?
What are the values in the various fields in the IP datagram(s) generated by the router?

Considering that 1,600 / 500 = 3.2, rounding up is 4. Hence, we will need to break the 1,600 bytes fragment to 4 smaller fragments to fit the outgoing link.

Fields that will be generated by the router:

- identification: same as the original fragment which is 291

- flag: last fragment of the original datagram has flag set to 0, other fragments have flag set to 1
- offset: to indicate where the fragment fits within the original datagram

Question 3 - Forwarding

❓ Consider an IP network using 32-bit host addresses. Suppose a router has four links, numbered 0 to 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

Write out a forwarding table that has five entries (rows), uses longest prefix matching, and forwards packets to the correct link interfaces.

Describe how your forwarding table determines the appropriate link interrace for the IP datagrams with destination addresses:
11111000 10010001 01010001 01010101
11100000 00000000 11000011 00111100
11100001 10000000 00010001 01110111

Forwarding table

Header value	Output link
11100000 00000000	0
11100000 00000001	1
11100000 00000010	2
11100001	2
otherwise	3

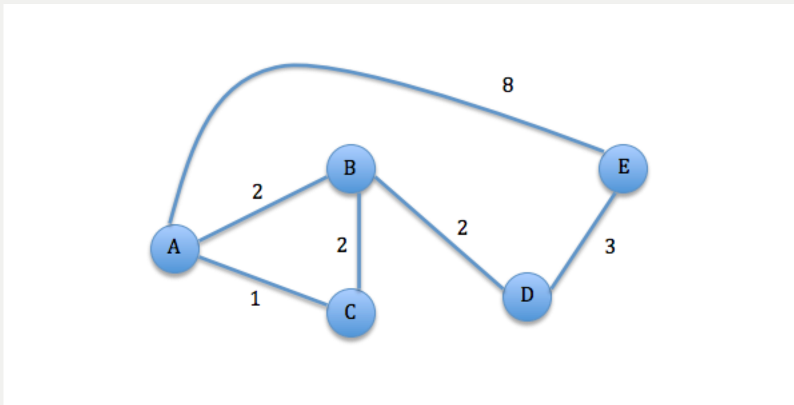
How the forwarding is used to route each packet:

Packet	How it is processed	Output link
11111000 10010001 01010001 01010101	Looking at each bit from left to right 11111000 → does not match with any header value for output links 0, 1, 2 → route the packet to link 3	3
11100000 00000000 11000011 00111100	11100000 00000000 → matches with output link 0 headers	0

Packet	How it is processed	Output link
11100001 10000000 00010001 01110111	11100001 1 → matches with output link 2 headers	2

Question 4 - Routing Algorithms

❓ Using node E as your base, show how node E builds it's routing table, using Dijkstra's algorithm and then using Distance Vector.



Using Dijkstra’s algorithm:

Applying Dijkstra’s algorithm, we form a table like below after each iteration:

Step	N’ (list of visited nodes)	D(A), p(A)	D(B), p(B)	D(C), p(C)	D(D), p(D)
0	E	8, E	∞	∞	3, E
1	ED	8,E	5,D	∞	
2	EDB	7,B		7,B	
3	EDBA			7,B	
4	EDBAC				

Final routing table for node E as a result of this algorithm:

Destination	Link	Distance
A	(E,A)	8
B	(E,D)	5
C	(E,D)	7
D	(E,D)	3

Using Distance Vector’s algorithm:

Time t=0:

		cost to				
		A	**B**	**C**	**D**	**E**
from	A	0	2	1	∞	8
	B	2	0	2	2	∞
	C	1	2	0	∞	∞
	D	∞	2	∞	0	3

	E	8	∞	∞	3	0

Time t=1:

- A receives forwarding tables of B, C, E
- B receives forwarding tables of A, C, D
- C receives forwarding tables of A, B
- D receives forwarding tables of B, E
- E receives forwarding tables of A, D

Updated info using Bellman-Ford equation:

			cost to				
		neighbours	**A**	**B**	**C**	**D**	**E**
from	A	B, C, E	0	min (2, 0+2, 1+2, 8+∞) = 2 → no change in path	min (1, 2+2, 0+1, 8+∞) = 1 → no change in path	min(∞,2+2,1+∞, 8+3) = 4 → path changed to A > B >	min (8, 2+∞, 1∞+, 8+0)=8 → no change in path
	B	A, C, D	min(2, 2+0, 2+1, 2+∞) = 2 → no change in path	0	min(2, 2+1, 2+0, 2+∞) = 2 → no change in path	min(2, 2+∞, 2+∞, 2+0) = 2 → no change in path	min(∞, 2+8, 2+∞, 2+3) = 5 path changed B > D > E
	C	A, B	min(1, 1+0, 2+2)=1 → no change in path	min(2, 1+2, 2+0)=2 → no change in path	0	min(∞, 1+∞, 2+2)=4 → path changed to C > B > D	min(∞, 1+8, 2+∞)=9 → pat changed to C>A>E
	D	B, E	min(∞, 2+2, 3+8)=4 → path changed to D > B > A	min(2, 2+0, 3+∞)=2 → no change in path	min(∞, 2+2, 3+∞)=4 → path changed to D > B > C	0	min(3, 2+∞, 3+0)=3 → no change in path
	E	A, D	min(8, 8+0, 3+∞)=8 → no change in path	min(∞, 8+2, 3+2)=5 → path changed to E > A >B	min(∞, 8+1, 3+∞)=9 → path changed to E > A > C	min(3, 8+∞, 3+0)=3 → no change in path	0

Simply the table:

			cost to				
		neighbours	**A**	**B**	**C**	**D**	**E**
from	A	B, C, E	0	2	1	4	8
	B	A, C, D	2	0	2	2	5
	C	A, B	1	2	0	4	9
	D	B, E	4	2	4	0	3
	E	A, D	8	5	9	3	0

Time t=2:

			cost to				
		neighbours	**A**	**B**	**C**	**D**	**E**
from	A	B, C, E	0	min (2, 2+0, 1+2, 8+5)=2 → no change	min (1, 2+2, 1+0, 8+9)=1 → no change	min(4, 2+2, 1+4, 8+3)=4 → no change	min (8, 2+5, 1+9, 8+0)=7 → path changed
	B	A, C, D	min (2, 2+0, 2+1, 2+4)=2 → no change	0	min(2, 2+1, 2+0, 2+4)=2 → no change	min (2, 2+4, 2+4, 2+3)=2 → no change	min (5, 2+8, 2+9, 2+3)=5 → no change

	C	A, B	min (1, 1+0, 2+2)=1 → no change	min (2, 1+2, 2+0)=2 → no change	0	min (4, 1+4, 2+2)=4 → no change	min (9, 1+8, 2+5)=7 → patl changed
	D	B, E	min (4, 2+2, 3+8)=4 → no change	min (2, 2+0, 3+5)=2 → no change	min(4, 2+2, 3+9)=4 → no change	0	min (3, 2+5, 3+0)=3 → no change
	E	A, D	min (8, 8+0, 3+4)=7 → path changed to E > D > A	min (5, 8+2, 3+2)=5 → no change	min(9, 8+1, 3+4)=7 → path changed	min (3, 8+4, 3+0)=3 → no change	0

Simply the table:

			cost to				
		neighbours	**A**	**B**	**C**	**D**	**E**
from	A	B, C, E	0	2	1	4	7
	B	A, C, D	2	0	2	2	5
	C	A, B	1	2	0	4	7
	D	B, E	4	2	4	0	3
	E	A, D	7	5	7	3	0

Time t=3:

			cost to				
		neighbours	**A**	**B**	**C**	**D**	**E**
from	A	B, C, E	0	min(2, 2+0, 1+2, 7+5)=2 → no change	min(1, 2+2, 1+0, 7+7)=1 → no change	min(4, 2+4, 1+4, 7+3)=4 → no change	min(7, 2+7, 1+ 7+0)=7 → no change
	B	A, C, D	min(2, 2+0, 2+1, 2+4)=2 → no change	0	min(2, 2+1, 2+0, 2+4)=2 → no change	min(2, 2+4, 2+, 2+0)=2 → no change	min(5, 2+7, 2+ 2+3)=5 → no change
	C	A, B	min(1, 1+0, 2+2)=1 → no change	min(2, 1+2, 2+0)=2 → no change	0	min(4, 1+4, 2+2)=4 → no change	min(7, 1+7, 2+5)=7 → no change
	D	B, E	min(4, 2+2, 3+7)=4 → no change	min(2, 2+0, 3+5)=2 → no change	min(4, 2+2, 3+7)=4 → no change	0	min(3, 2+5, 3+0)=3 → no change
	E	A, D	min(7, 7+0, 3+4)=7 → no change	min(5, 7+2, 3+2)=5 → no change	min(7, 7+1, 3+4)=7 → no change	min(3, 7+4, 3+0)=3 → no change	0

→ no path change in the whole network → done with DV algorithm

Final forwarding table of E:

Destination	Link	Distance
A	(E,A)	8
B	(E,D)	5
C	(E,D)	7
D	(E,D)	3

Question 5

❓ What topics or examples would you like to cover in this workshop session?

I would like to discuss on assignment 3 if possible.