CS3543 Lab Routing Assignment

Member 1: SUSHMA NALARI (CS19BTECH11006)
Member 2: MALLA SRI HARI (CS19BTECH11039)
Member 3: LIKITHA LANGARU (CS19BTECH11008)
Member 4: LINGAREDDY HARSHA VARDHAN (CS19BTECH11023)

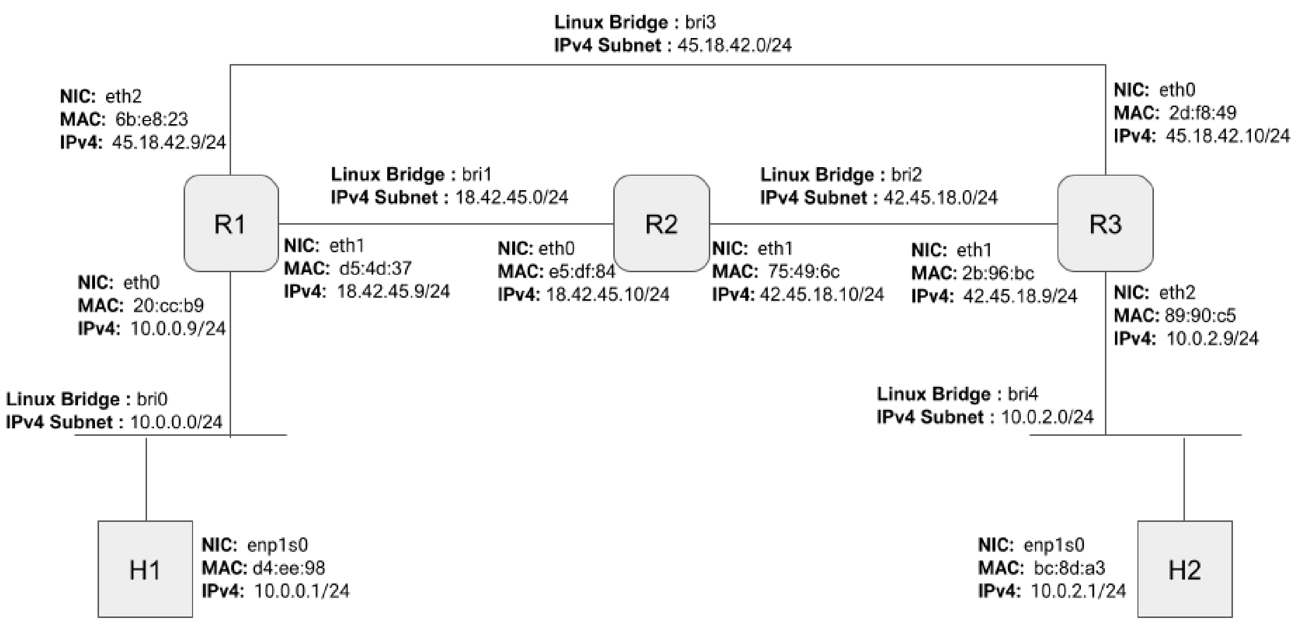
Question 1.
Fill the blanks in Table 1 to clarify NIC and IPv4 to belong to Subnets 1 to 4. If there is no corresponding NIC belonging to a subnet, mention "N/A". All the prefixes must be planned by yourself.

	Linux Bridge	H1	H2	R1	R2	R3
Subnet 1	bri0	enp1s0: 10.0.0.1/24 MAC address: 52:54:00:d4:ee:98	N/A	eth0: 10.0.0.9/24 MAC address: 52:54:00:20:cc:b9	N/A	N/A
Subnet 2	bri1	N/A	N/A	eth1: 18.42.45.9/24 MAC address: 52:54:00:d5:4d:37	eth0: 18.42.45.10/24 MAC address: 52:54:00:e5:df:84	N/A
Subnet A	bri2	N/A	N/A	N/A	eth1: 42.45.18.10/24 MAC address: 52:54:00:75:49:6c	eth1: 42.45.18.9/24 MAC address: 52:54:00:2b:96:bc
Subnet B	bri3	N/A	N/A	eth2: 45.18.42.9/24 MAC address:52:54:00:6b:e8:23	N/A	eth0: 45.18.42.10/24 MAC address: 52:54:00:2d:f8:49

Subnet C	bri4	N/A	enps10: 10.0.2.1/24 MAC address: 52:54:00:bc:8d:a3	N/A	N/A	eth2: 10.0.2.9/24 MAC address: 52:54:00:89:90:c5

Question 2.
Illustrate the network diagram that appropriately contains the information given in Table 1. The full mark will be given if the network diagram fully covers the information given in the table. (Linux Bridge and other information must also be mentioned for the sake of explainability of answers to the following questions). The original presentation file can be locally copied to your Google Drive and used to work on this assignment.

NETWORK DIAGRAM:



NIC configuration and Static Routing Instruction

1. Configure all NICs of the hosts and routers (H1, H2, R1, R2 and R3) as planned in Table 1 and the network diagram.
2. Manually configure the routing table of all the routers so that 1) the path from H1 to H2 is always {H1 -> R1 -> R3 -> H2} and 2) the path from H2 to H1 is always {H2 -> R3 -> R2 -> R1 -> H1}.
3. Make sure that H1 and H2 can ping with each other.

Question 3.1
Paste the screen capture of the ping command from H1 and H2 to show that the static routing configuration is working to allow H1 and H2 to communicate with each other.

PING H2 from H1:

```
vm1@vm1:~$ ping 10.0.2.1 -c 6
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=61 time=3.14 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=61 time=1.78 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=61 time=1.37 ms
64 bytes from 10.0.2.1: icmp_seq=4 ttl=61 time=2.78 ms
64 bytes from 10.0.2.1: icmp_seq=5 ttl=61 time=2.64 ms
64 bytes from 10.0.2.1: icmp_seq=6 ttl=61 time=3.43 ms

--- 10.0.2.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 1.373/2.524/3.431/0.724 ms
vm1@vm1:~$ _
```

PING H1 from H2:

```
vm2@vm2:~$ ping 10.0.0.1 -c 6
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=62 time=1.08 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=62 time=3.50 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=62 time=3.32 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=62 time=2.86 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=62 time=3.27 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=62 time=3.77 ms

--- 10.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 1.078/2.965/3.769/0.886 ms
vm2@vm2:~$ ~_
```

Question 3.b
Paste the screen capture of the routing table of R1.

ROUTING TABLE OF R1:

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

C>* 10.0.0.0/24 is directly connected, eth0, 02:29:23
S>* 10.0.2.0/24 [1/0] via 45.18.42.10, eth2, weight 1, 01:29:35
C>* 18.42.45.0/24 is directly connected, eth1, 02:29:23
C>* 45.18.42.0/24 is directly connected, eth2, 02:29:22
vyos@vyos:~$ _
```

Question 3.c
Perform traceroute from H1 to H2 so that the path is following the instruction. Paste the screen capture of the traceroute result of H1.

TRACEROUTE FROM H1 TO H2:

```
vm1@vm1:~$ traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.0.9 (10.0.0.9)  0.870 ms  0.626 ms  0.536 ms
 2  45.18.42.10 (45.18.42.10)  1.210 ms  1.138 ms  1.063 ms
 3  10.0.2.1 (10.0.2.1)  1.188 ms  1.098 ms  1.033 ms
vm1@vm1:~$
```

Question 3.d
Perform traceroute from H2 to H1 so that the path is following the instruction. Paste the screen capture of the traceroute result of H2.

TRACEROUTE FROM H2 TO H1:

```
vm2@vm2:~$ traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1), 30 hops max, 60 byte packets
 1  10.0.2.9 (10.0.2.9)  1.031 ms  0.922 ms  0.881 ms
 2  * * *
 3  18.42.45.9 (18.42.45.9)  1.999 ms  1.908 ms  1.810 ms
 4  10.0.0.1 (10.0.0.1)  2.842 ms  2.583 ms  2.956 ms
vm2@vm2:~$
```

Question 3.e

When a packet from H1 to H2 is transmitted by R1, whose MAC address is set as the destination address in the Ethernet header? Answer the names of node and NIC respectively.

EXPLANATION:

As mentioned packets are sent from H1 to H2 using the ping command (H2 is pinged from H1). The packets are transmitted through R1 and using tcpdump on R1 and '-e' the ethernet header is checked.

To transmit the packets through R1 (H1->R1->R3->H2), R3's eth0 MAC address is set as the destination address because it is the interface between R1 and R3.

DESTINATION ADDRESS: 52:54:00:2d:f8:49
NODE: R3
NIC: eth0

```
vyos@vyos:~$ sudo tcpdump -i eth2 -e -p icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
16:32:12.128414 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 1, length 64
16:32:13.130195 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 2, length 64
16:32:14.131434 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 3, length 64
16:32:15.132933 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 4, length 64
16:32:16.134652 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 5, length 64
16:32:17.135965 52:54:00:6b:e8:23 (oui Unknown) > 52:54:00:2d:f8:49 (oui Unknown
), ethertype IPv4 (0x0800), length 98: 10.0.0.1 > 10.0.2.1: ICMP echo request, i
d 9, seq 6, length 64
```

Question 3.f

Explain what kind of mistake can create a routing loop with an actual example which can be implemented in the network used in this network (3 marks). Actually implement the routing loop and show the result of traceroute exhibiting the routing loop (2 marks).

ROUTING LOOP:

A routing loop is created when the packets keep transmitted in a loop between the routers without reaching the destination. This is possible when the destination address/subnet of a router or an interface is wrongly assigned.

For example, here the path that is followed is:

H1→ R1 → R3 → H2 and H1→ R1→ R2 → R3 → H2

If the destination subnet for the routers is changed to bri4 (R3 <--> H2) and the direct connection between R3 and H2 is broken by deleting the interface between them, then a loopm= gets created. The packets sent from H1 do not reach H2 and keep transmitting between R1, R2, and R3.

This can be observed in the screen capture of the traceroute of the packets from H1 to H2.

The route *"18.42.45.9 (18.42.45.9) * * *"* keeps repeating indicating that the packets are moving through the same router over and over again. If there is no loop the packets pass through that point only once and the transmission completes in 3 or 4 hops.

```
vm1@vm1:~$ traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.0.9 (10.0.0.9)  0.585 ms  0.540 ms  0.554 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  18.42.45.9 (18.42.45.9)  3.087 ms * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  18.42.45.9 (18.42.45.9)  6.904 ms  6.776 ms  12.585 ms
20  * * *
21  * * *
22  18.42.45.9 (18.42.45.9)  11.963 ms  11.841 ms *
23  * * *
24  * * *
25  * * 18.42.45.9 (18.42.45.9)  8.633 ms
26  * * *
27  * * *
28  18.42.45.9 (18.42.45.9)  11.475 ms  11.415 ms *
29  * * *
30  * * *
vm1@vm1:~$
```

#
Dynamic Routing Instruction using OSPF
#

1. Flush the static routing configuration from all the routers.
2. Enable tcpdump on both of R1's NICs, on which OSPF is enabled,and save (write) the captured packets. The packet capture files will be used to answer a question.
3. Enable OSPF. You can configure all the NICs of routers to belong to Area 0.
4. Make sure that H1 and H2 can ping with each other.

Question 4.a.
Perform traceroute from H1 to H2 as well as from H2 to H1. 1) Explain the path of both directions, 2) paste the screen captures of traceroute for both directions.

PATH:

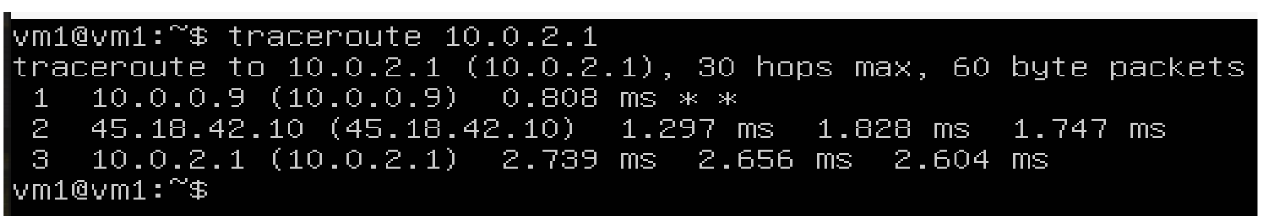
OSPF(Open Shortest Path First) Protocol chooses the path with the shortest cost using Dijkstra's algorithm. The possible paths and their costs are (considering the default cost of each interface as 10):

Path1: H1→ R1 → R3 → H2 Cost: 10 (1 interface)
Path2: H1→ R1 → R2 → R3 → H2 Cost: 10+10 = 20 (2 interfaces)

From the costs we can conclude why the path H1 → R1 → R3 → H2 is chosen. SImilar is the case with the path selection from H2 to H1.

H1 TO H2	H2 TO H1
H1→ R1 → R3 → H2	**H2→ R3 → R1 → H1**

TRACEROUTE H1 TO H2:



```
vm1@vm1:~$ traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.0.9 (10.0.0.9)  0.808 ms * *
 2  45.18.42.10 (45.18.42.10)  1.297 ms  1.828 ms  1.747 ms
 3  10.0.2.1 (10.0.2.1)  2.739 ms  2.656 ms  2.604 ms
vm1@vm1:~$
```

TRACEROUTE H2 TO H1:

```
vm2@vm2:~$ traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1), 30 hops max, 60 byte packets
 1  10.0.2.9 (10.0.2.9)  0.895 ms  0.706 ms  0.503 ms
 2  45.18.42.9 (45.18.42.9)  1.475 ms  1.356 ms  1.249 ms
 3  10.0.0.1 (10.0.0.1)  3.093 ms  2.955 ms  2.807 ms
vm2@vm2:~$
```

Question 4.b

Paste screen captures of 1) the routing table of R1, and 2) the list of OSPF neighbors.

ROUTING TABLE OF R1:

```
vyos@vyos:~$ show ip route ospf
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

O   10.0.0.0/24 [110/1] is directly connected, eth0, weight 1, 00:24:34
O>* 10.0.2.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:00:45
O   18.42.45.0/24 [110/1] is directly connected, eth1, weight 1, 00:01:40
O>* 42.45.18.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:00:45
O   45.18.42.0/24 [110/1] is directly connected, eth2, weight 1, 00:00:45
vyos@vyos:~$
```

LIST OF OSPF NEIGHBORS OF R1:

```
vyos@vyos:~$ show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
               RXmtL RqstL DBsmL
18.42.45.10       1 Full/DR           35.478s 18.42.45.10     eth1:18.42.45.9
                    0     0     0
10.0.2.9          1 Full/Backup       39.907s 45.18.42.10     eth2:45.18.42.9
                    0     0     0

vyos@vyos:~$ _
```

LIST OF OSPF NEIGHBORS OF R2:

```
vyos@vyos:~$ show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
               RXmtL RqstL DBsmL
10.0.0.9          1 Full/Backup       31.836s 18.42.45.9      eth0:18.42.45.10
                    0     0     0
10.0.2.9          1 Full/Backup       34.312s 42.45.18.9      eth1:42.45.18.10
                    0     0     0

vyos@vyos:~$ _
```

LIST OF OSPF NEIGHBORS OF R3:

```
vyos@vyos:~$ show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
               RXmtL RqstL DBsmL
10.0.0.9          1 Full/DR           38.505s 45.18.42.9      eth0:45.18.42.10
                   0     0     0
18.42.45.10       1 Full/DR           36.463s 42.45.18.10     eth1:42.45.18.9
                   0     0     0

vyos@vyos:~$ _
```

Question 4.c
Revise your OSPF configuration of each router so that the traffic between H1 and H2 always goes through the path {H1 <---> R1 <---> R2 <---> R3 <---> H2}. 1) Explain what kind of revision you made on which router. Also, 2) paste the screen capture of traceroute between H1 and H2 to show that the above mentioned path is successfully implemented.

EXPLANATION:

From the default costs of each interface, the path selected will always be H1 <---> R1 <---> R3 <---> H2. To make sure that H1 <---> R1 <---> R2 <---> R3 <---> H2 is always chosen, we must make it the shortest path (lowest cost).

This can be done by increasing the weight of the interfaces between R1 and R2 or decreasing the weight of the interfaces between R1 and R2 and(or) R2 and R3. (We chose the first method here).

The weight of the interface between R1 and R2 is increased to 50.

From 4.a:
Path1: H1→ R1 → R3 → H2 Cost: 50 (1 interface with modified weight/cost)
Path2: H1→ R1 → R2 → R3 → H2 Cost: 10+10 = 20 (2 interfaces)

From the costs we can conclude the path H1 → R1 → R2 → R3 → H2 is always chosen as the shortest path. SImilar is the case with the path selection from H2 to H1.

TRACEROUTE H1 TO H2:

```
vm1@vm1:~$ traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.0.9 (10.0.0.9)  0.914 ms * *
 2  18.42.45.10 (18.42.45.10)  1.507 ms  1.384 ms  1.282 ms
 3  42.45.18.9 (42.45.18.9)  3.206 ms  3.077 ms  2.976 ms
 4  10.0.2.1 (10.0.2.1)  3.077 ms  2.964 ms  2.856 ms
vm1@vm1:~$ _
```

PATH: H1→ R1 → R2 → R3 → H2

TRACEROUTE H2 TO H1:

```
vm2@vm2:~$ traceroute 10.0.0.1
traceroute to 10.0.0.1 (10.0.0.1), 30 hops max, 60 byte packets
 1  10.0.2.9 (10.0.2.9)  0.451 ms  0.410 ms  0.395 ms
 2  42.45.18.10 (42.45.18.10)  0.833 ms  0.821 ms  0.792 ms
 3  18.42.45.9 (18.42.45.9)  1.133 ms  1.114 ms  1.100 ms
 4  10.0.0.1 (10.0.0.1)  1.313 ms  1.301 ms  1.316 ms
vm2@vm2:~$ _
```

PATH: H2→ R3 → R2 → R1 → H1

Question 4.d
Shutdown R2, and explain what happens to the routing table of R1 after R2 becomes down.

NEW OSPF ROUTES IN R1 (AFTER R2 IS DOWN):

As R2 is down, no acknowledgement messages will be received from it. The path followed will be the only path available i.e, transmits all the packets it receives to R3.

In the attached screen capture, this can be observed.
Packets transfer is done *“via 45.18.42.10, eth2”.*

This can be clearly observed in the ip route screen capture attached.

```
vyos@vyos:~$ show ip route ospf
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

O   10.0.0.0/24 [110/1] is directly connected, eth0, weight 1, 00:24:34
O>* 10.0.2.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:00:45
O   18.42.45.0/24 [110/1] is directly connected, eth1, weight 1, 00:01:40
O>* 42.45.18.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:00:45
O   45.18.42.0/24 [110/1] is directly connected, eth2, weight 1, 00:00:45
vyos@vyos:~$
```

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

O   10.0.0.0/24 [110/1] is directly connected, eth0, weight 1, 00:25:06
C>* 10.0.0.0/24 is directly connected, eth0, 00:31:23
O>* 10.0.2.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:01:17
O   18.42.45.0/24 [110/1] is directly connected, eth1, weight 1, 00:02:12
C>* 18.42.45.0/24 is directly connected, eth1, 00:31:23
O>* 42.45.18.0/24 [110/2] via 45.18.42.10, eth2, weight 1, 00:01:17
O   45.18.42.0/24 [110/1] is directly connected, eth2, weight 1, 00:01:17
C>* 45.18.42.0/24 is directly connected, eth2, 00:31:23
vyos@vyos:~$ _
```

Question 4.e
Observing the packet capture data at R1, explain what kind of OSPF messages flew from/to R1 after R2 became down.

EXPLANATION:

Here, R2 is initially shut down, R2 is pinged from R1, and the packet capture data is observed at R1.

By default, a “hello” packet is sent to the multicast address 224.0.0.5 in intervals of 10 seconds. If this packet is received then the OSPF router sends back this packet. This router keeps track of the packets sent from a link. If the router does not receive any packets for 4 and above intervals continuously then the link is treated as disconnected.

Here if we compare the packet transfer from R1 to R3 and R1 to R2, this can be observed.

In R1 to R3, we know that none of the links are dead/disconnected, the packets sent from both the routers(R1 and R3) are captured.

IP 45.18.42.9 > 224.0.0.5: ……………..
IP 45.18.42.10 > 224.0.0.5: ……………..

Whereas in the case of R1 to R2, as we shut down R2, the link(Sublink A) between them is disconnected. So, the packets are sent from R1 but none of them are received.

IP 18.42.45.9 > 224.0.0.5: ……………..
IP 18.42.45.9 > 224.0.0.5: ……………..

TCPDUMP OF R1 AT INTERFACE CONNECTED TO R3 (AFTER R2 IS DOWN):

```
vyos@vyos:~$ sudo tcpdump -i eth2 -c 10
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
19:34:21.599623 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:23.892271 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:31.599680 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:33.892156 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:41.599817 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:43.892333 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:51.600039 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:34:53.892819 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:35:01.600141 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:35:03.892838 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
10 packets captured
10 packets received by filter
0 packets dropped by kernel
vyos@vyos:~$
```

TCPDUMP OF R1 AT INTERFACE CONNECTED TO R2 (AFTER R2 IS DOWN):

```
vyos@vyos:~$ sudo tcpdump -i eth1 -c 10
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
19:37:41.508393 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:37:51.508452 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:01.508515 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:11.509010 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:21.508671 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:31.508773 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:41.508846 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:38:51.509270 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:39:01.509332 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
19:39:11.509392 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 44
10 packets captured
10 packets received by filter
0 packets dropped by kernel
vyos@vyos:~$ _
```

Here initially R2 is powered on, R2 is pinged from R1, the packet capture data is observed at R1, R2 is shut down, and the packet capture data is observed at R1.

If we observe, the packets are transmitted from both the routers (R1 and R3) to 224.0.0.5 when R2 is running.

IP 18.42.45.9 > 224.0.0.5: ……………..
IP 18.42.45.10 > 224.0.0.5: ……………..

While R2 is shutting down, a IGMP "Leave Group" message is sent. This indicates that the multicast transmissions are no longer required at that address.

Then after the shut down of R2, the transmissions are the same as observed above (TCPDUMP OF R1 AT INTERFACE CONNECTED TO R2 (AFTER R2 IS DOWN)).

This can be observed in the attached screen capture.

TCPDUMP OF R1 AT INTERFACE CONNECTED TO R2 (DURING SHUTDOWN OF R2):

```
vyos@vyos:~$ sudo tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
19:40:51.510458 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:40:55.113502 IP 18.42.45.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:01.510478 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:05.113543 IP 18.42.45.10 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:08.084099 IP 18.42.45.10 > 224.0.0.5: OSPFv2, LS-Update, length 76
19:41:08.086861 IP 18.42.45.10 > 224.0.0.22: igmp v3 report, 2 group record(s)
19:41:08.163903 IP 18.42.45.9 > 224.0.0.5: OSPFv2, LS-Ack, length 44
19:41:08.164936 IP 18.42.45.10 > 224.0.0.22: igmp v3 report, 2 group record(s)
19:41:11.510878 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:21.510944 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:31.511000 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
19:41:41.511057 IP 18.42.45.9 > 224.0.0.5: OSPFv2, Hello, length 48
^C
12 packets captured
12 packets received by filter
0 packets dropped by kernel
vyos@vyos:~$
```

One more thing that can be observed from the packet transmissions after R2 is down is that link state updates and link state acknowledgements are sent through the multicast address.

This can be observed in the attached screen capture.

LINK STATUS UPDATES AND ACKNOWLEDGEMENT (AFTER R2 IS DOWN):

```
20:02:13.916675 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:21.617173 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:23.916928 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:31.617339 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:33.917234 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:41.388499 IP 45.18.42.9 > 224.0.0.5: OSPFv2, LS-Update, length 60
20:02:41.456179 IP 45.18.42.10 > 224.0.0.5: OSPFv2, LS-Ack, length 44
20:02:41.617401 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:43.917352 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:51.617349 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:02:53.917482 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:01.617482 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:03.917549 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:11.617542 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:13.917722 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:21.617485 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:23.917786 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:31.617638 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:33.917819 IP 45.18.42.10 > 224.0.0.5: OSPFv2, Hello, length 48
20:03:41.618174 IP 45.18.42.9 > 224.0.0.5: OSPFv2, Hello, length 48
^C
163 packets captured
163 packets received by filter
0 packets dropped by kernel
vyos@vyos:~$
```