

## ***LAB5***

Q1)

```
ip 10.0.1.11 255.255.255.0 10.0.1.254
```

Checking for duplicate address...

```
PC1 : 10.0.1.11 255.255.255.0 gateway 10.0.1.254
```

a)

```
show arp
```

arp table is empty

```
PC2> ping 10.0.1.11
```

```
84 bytes from 10.0.1.11 icmp_seq=1 ttl=64 time=0.395 ms
```

```
84 bytes from 10.0.1.11 icmp_seq=2 ttl=64 time=0.555 ms
```

```
84 bytes from 10.0.1.11 icmp_seq=3 ttl=64 time=0.493 ms
```

```
84 bytes from 10.0.1.11 icmp_seq=4 ttl=64 time=0.508 ms
```

```
84 bytes from 10.0.1.11 icmp_seq=5 ttl=64 time=0.255 ms
```

```
^C
```

```
PC1> show arp
```

```
00:50:79:66:68:01 10.0.1.12 expires in 111 seconds
```

b)

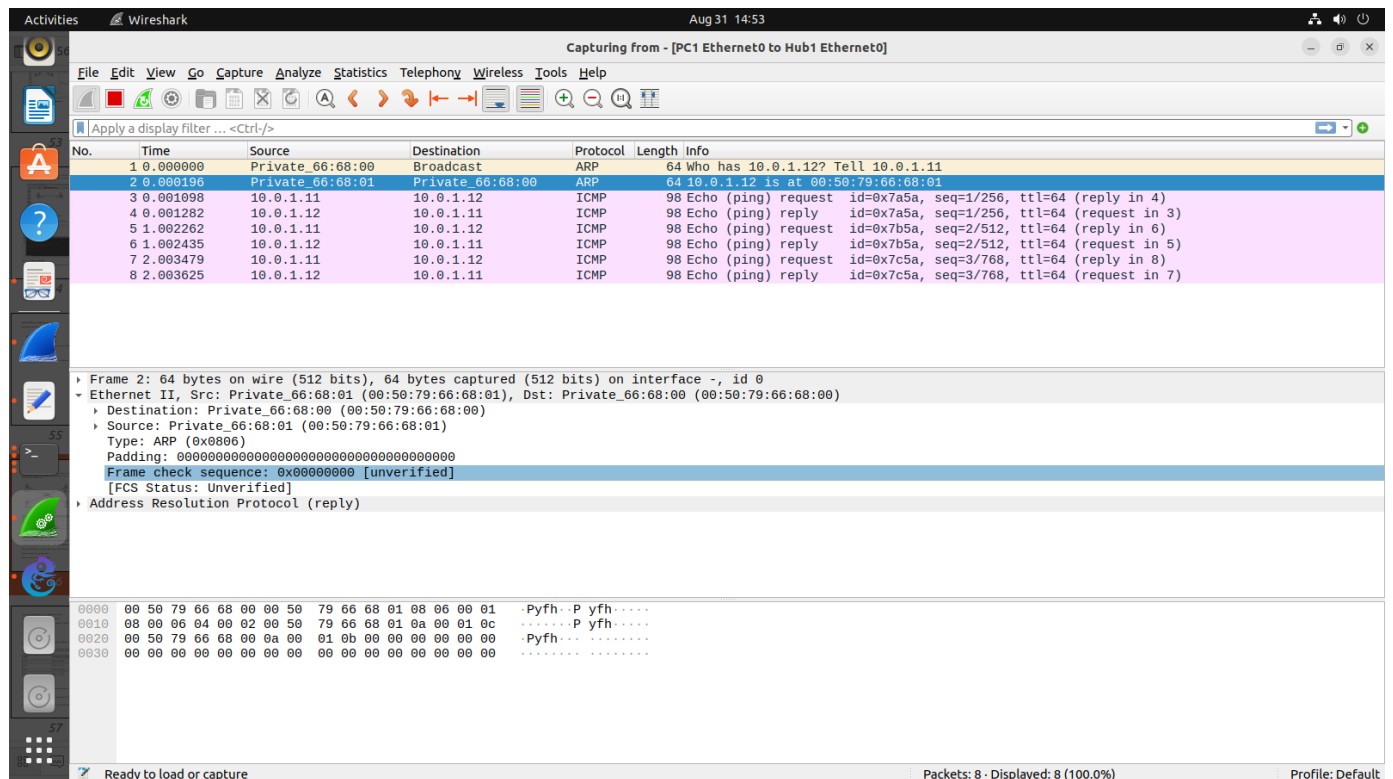
on wireshark filter: ip.addr == 10.0.1.12

c)

Destination: Private\_66:68:00 (00:50:79:66:68:00)

Source: Private\_66:68:01 (00:50:79:66:68:01)

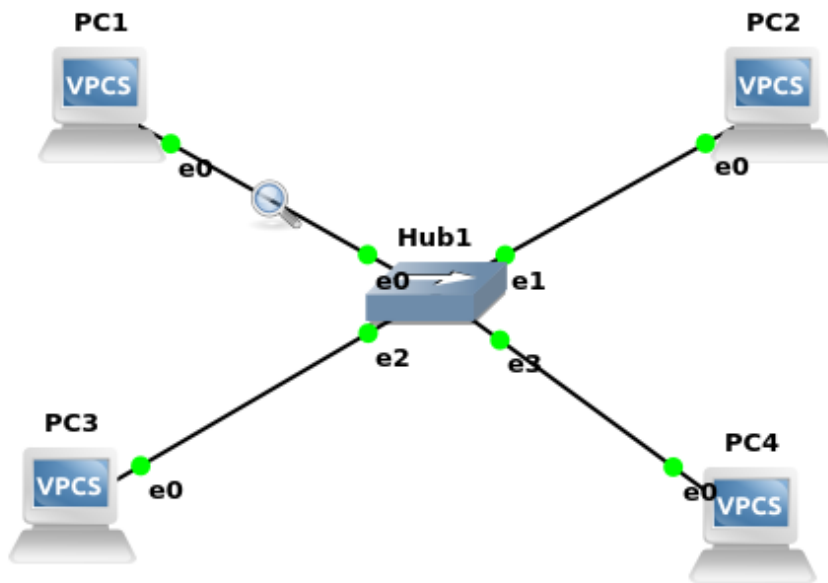
Type: ARP (0x0806)



d)

PC1> show arp

00:50:79:66:68:01 10.0.1.12 expires in 111 seconds



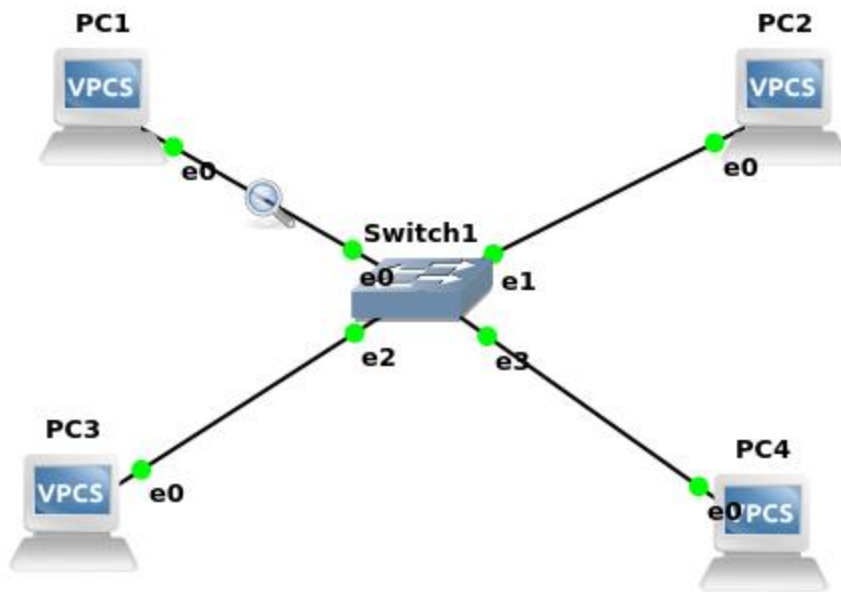
Q3)

PC1> ip 10.0.1.100 / 24

Checking for duplicate address...

PC1 : 10.0.1.100 255.255.255.0

Similarly for all



b)

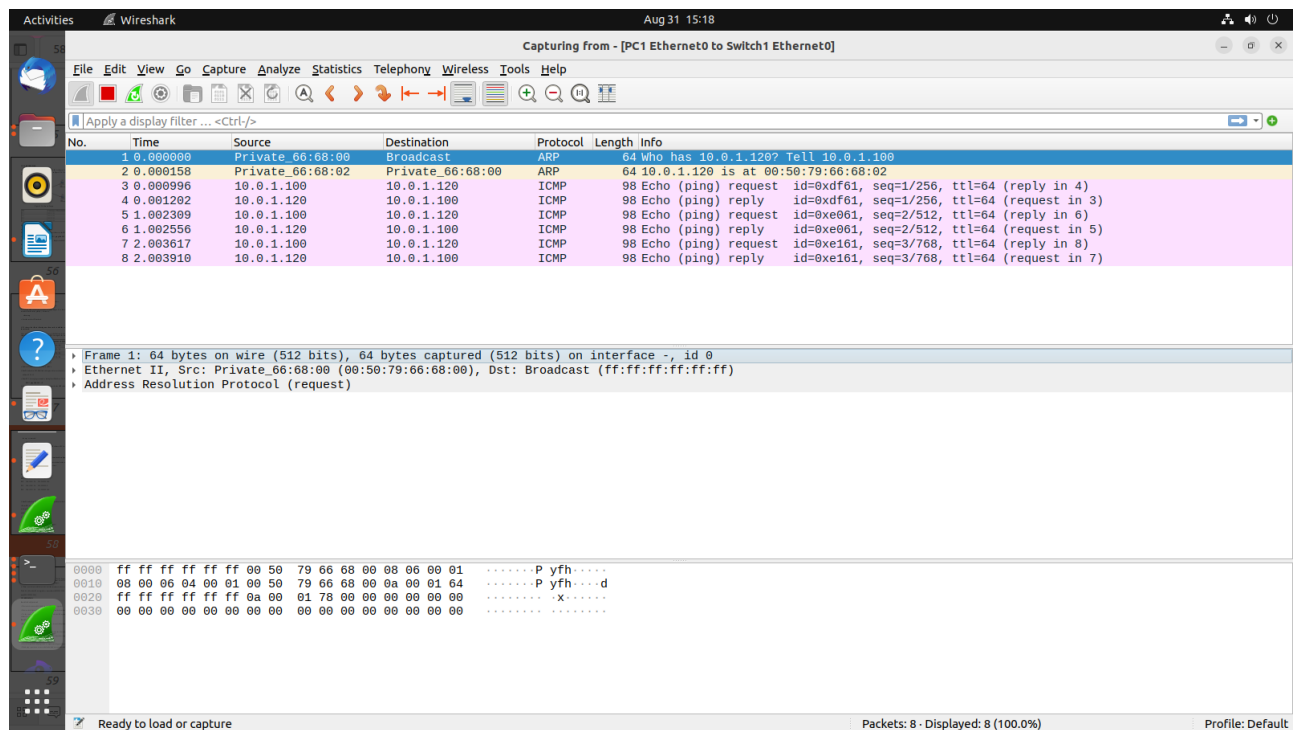
1)

```
PC1> ping 10.0.1.120 / 24 -c 3
```

```
84 bytes from 10.0.1.120 icmp_seq=1 ttl=64 time=0.327 ms
```

```
84 bytes from 10.0.1.120 icmp_seq=2 ttl=64 time=0.493 ms
```

```
84 bytes from 10.0.1.120 icmp_seq=3 ttl=64 time=0.633 ms
```



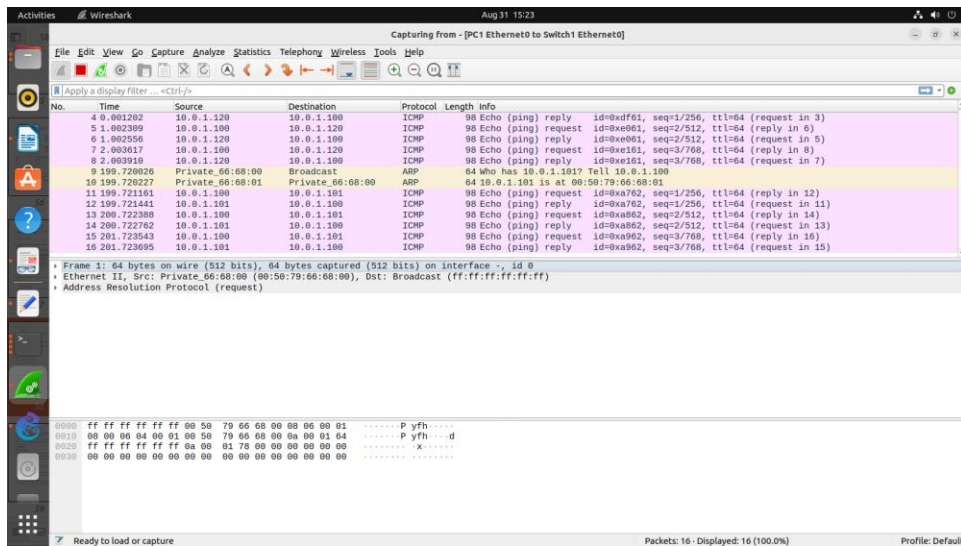
2)

PC1> ping 10.0.1.101 / 28 -c 3

84 bytes from 10.0.1.101 icmp\_seq=1 ttl=64 time=0.553 ms

84 bytes from 10.0.1.101 icmp\_seq=2 ttl=64 time=0.604 ms

84 bytes from 10.0.1.101 icmp\_seq=3 ttl=64 time=0.356 ms



3)

PC1> ping 10.0.1.121 / 28 -c 3

10.0.1.121 icmp\_seq=1 timeout

10.0.1.121 icmp\_seq=2 timeout

10.0.1.121 icmp\_seq=3 timeout

Activities Wireshark Aug 31 15:41

Capturing from - [PC1 Ethernet0 to Switch1 Ethernet0]

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Private_66:68:00	Broadcast	ARP	64	Who has 10.0.1.120? Tell 10.0.1.100
2	0.000158	Private_66:68:02	Private_66:68:00	ARP	64	10.0.1.120 is at 00:50:79:66:68:02
3	0.000996	10.0.1.100	10.0.1.120	ICMP	98	Echo (ping) request id=0xdf61, seq=1/256, ttl=64 (reply in 4)
4	0.001202	10.0.1.120	10.0.1.100	ICMP	98	Echo (ping) reply id=0xdf61, seq=1/256, ttl=64 (request in 3)
5	1.002309	10.0.1.100	10.0.1.120	ICMP	98	Echo (ping) request id=0xe061, seq=2/512, ttl=64 (reply in 6)
6	1.002566	10.0.1.120	10.0.1.100	ICMP	98	Echo (ping) reply id=0xe061, seq=2/512, ttl=64 (request in 5)
7	2.003617	10.0.1.100	10.0.1.120	ICMP	98	Echo (ping) request id=0xe161, seq=3/768, ttl=64 (reply in 8)
8	2.003910	10.0.1.120	10.0.1.100	ICMP	98	Echo (ping) reply id=0xe161, seq=3/768, ttl=64 (request in 7)
9	199.720026	Private_66:68:00	Broadcast	ARP	64	Who has 10.0.1.101? Tell 10.0.1.100
10	199.720227	Private_66:68:01	Private_66:68:00	ARP	64	10.0.1.101 is at 00:50:79:66:68:01
11	199.721161	10.0.1.100	10.0.1.101	ICMP	98	Echo (ping) request id=0xa762, seq=1/256, ttl=64 (reply in 12)
12	199.721441	10.0.1.101	10.0.1.100	ICMP	98	Echo (ping) reply id=0xa762, seq=1/256, ttl=64 (request in 11)
13	200.722388	10.0.1.100	10.0.1.101	ICMP	98	Echo (ping) request id=0xa862, seq=2/512, ttl=64 (reply in 14)
14	200.722762	10.0.1.101	10.0.1.100	ICMP	98	Echo (ping) reply id=0xa862, seq=2/512, ttl=64 (request in 13)
15	201.723543	10.0.1.100	10.0.1.101	ICMP	98	Echo (ping) request id=0xa962, seq=3/768, ttl=64 (reply in 16)
16	201.723695	10.0.1.101	10.0.1.100	ICMP	98	Echo (ping) reply id=0xa962, seq=3/768, ttl=64 (request in 15)
17	442.979924	Private_66:68:00	Broadcast	ARP	64	Who has 10.0.1.121? Tell 10.0.1.100
18	442.980085	Private_66:68:03	Private_66:68:00	ARP	64	10.0.1.121 is at 00:50:79:66:68:03
19	442.980971	10.0.1.100	10.0.1.121	ICMP	98	Echo (ping) request id=0x9a63, seq=1/256, ttl=64 (reply in 24)
20	442.981170	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
21	443.981748	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
22	444.981852	10.0.1.100	10.0.1.121	ICMP	98	Echo (ping) request id=0x9c63, seq=2/512, ttl=64 (reply in 29)
23	444.982041	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
24	445.983172	10.0.1.121	10.0.1.100	ICMP	98	Echo (ping) reply id=0x9a63, seq=1/256, ttl=64 (request in 19)
25	445.983245	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
26	446.983321	10.0.1.100	10.0.1.121	ICMP	98	Echo (ping) request id=0x9e63, seq=3/768, ttl=64 (reply in 33)
27	446.983612	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
28	447.984204	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
29	448.984738	10.0.1.121	10.0.1.100	ICMP	98	Echo (ping) reply id=0x9c63, seq=2/512, ttl=64 (request in 22)
30	448.984778	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
31	449.985697	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
32	450.986732	Private_66:68:03	Broadcast	ARP	64	Who has 0.0.0.0? Tell 10.0.1.121
33	451.987705	10.0.1.121	10.0.1.100	ICMP	98	Echo (ping) reply id=0x9e63, seq=3/768, ttl=64 (request in 26)

Frame 17: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface -, id 0

Ready to load or capture Packets: 33 · Displayed: 33 (100.0%) Profile: Default

4)

PC4> ping 10.0.1.100 / 24 -c 3

No gateway found

5)

PC2> ping 10.0.1.121 / 28 -c 3

No gateway found

6)

PC2> ping 10.0.1.120 / 24 -c 3

No gateway found

## **THEORY QUESTIONS BASED ON ABOVE QUESTIONS**

### **Based on Q1**

1. What is the destination MAC address of an ARP Request packet?

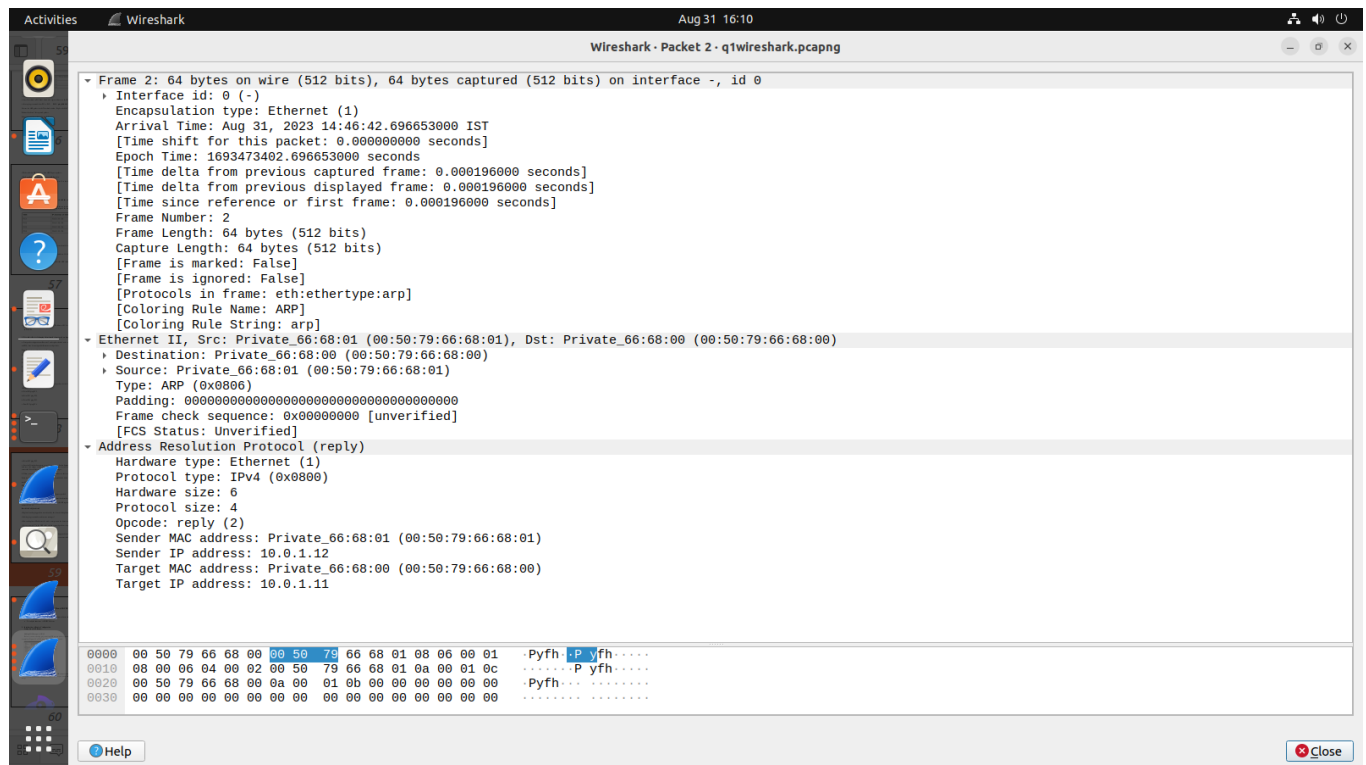
00:50:79:66:68:00

2. What are the different Type Field values in the Ethernet headers that you observed?

Type: ARP (0x0806)

3. Use the captured data to analyze the process in which ARP acquires the MAC address for IP address 10.0.1.12.





## Based on Q3

### Scenarios and Analysis:

PC1 10.0.1.100 / 24 255.255.255.0

PC2 10.0.1.101 / 28 255.255.255.240

PC3 10.0.1.120 / 24 255.255.255.0

PC4 10.0.1.121 / 28 255.255.255.240

let's expand the bits of the given IP addresses:

1. 10.0.1.100:

- IP address: 10.0.1.100
- Binary representation: 00001010.00000000.00000001.01100100

2. 10.0.1.101:

- IP address: 10.0.1.101

- Binary representation: 00001010.00000000.00000001.01100101

3. 10.0.1.120:

- IP address: 10.0.1.120

- Binary representation: 00001010.00000000.00000001.01111000

4. 10.0.1.121:

- IP address: 10.0.1.121

- Binary representation: 00001010.00000000.00000001.01111001

i. From PC1 ping PC3: Since PC1(10.0.1.100/24), and PC3(10.0.1.120/24) have IP addresses in the same /24 subnet they are on the same network. The ping operation between PC1 and PC3 will be successful because they can directly communicate within the same subnet. The first 24 bits of both the IP addresses match completely and the ping is successful.

ii. From PC1 ping PC2: PC1 and PC2 are in different subnets. PC1's IP address is in the subnet 10.0.1.100/24, and PC2's IP address is in the subnet 10.0.1.101/28. Even if they are within different subnets, the ping is successful. \*\*\*\*\*why? Even though they have different subnet masks, their IP addresses are within the same range of IP addresses. The ping operation will be successful because they can directly communicate within the same subnet. When PC1 finds PC2, it looks for only the first 24 bits to match, as the match occurs, it requests successfully. When PC2 needs to reply to PC1, it looks for the first 28 bits of the IP address to match. Coincidentally, since both IP addresses are the same for the first 28 bits, the reply is also sent successfully and the ping operation is successful. This is a completely coincidental case.

iii. From PC1 ping PC4: PC1 and PC4 are in different subnets due to their distinct subnet masks. PC1's IP address is in the subnet 10.0.1.100/24, and PC4's IP address is in the subnet 10.0.1.121/28. The ping operation will be unsuccessful because they are not on the same subnet. Moreover, a timeout occurs. \*\*\*\*\*why? The timeout occurs because when PC1 looks for a match of the first 24 bits of the IP address, and since the first 24 bits are the same in the first IP address, PC1 assumes that it has found the host and requests. But PC4 looks for a match of the first 28 bits of the IP addresses and thus does not reply to PC1. Therefore a timeout occurs as PC1 is sending a request without a reply from PC4.

iv. From PC4 ping PC1: PC1 and PC4 are in different subnets due to their distinct subnet masks. PC1's IP address is in the subnet 10.0.1.100/24, and PC4's IP address is in the subnet 10.0.1.121/28. The ping operation will be unsuccessful because they are not on the same subnet. No gateway message is seen on the terminal. \*\*\*\*\*why? When PC4 is looking for

the first 28 bits of the ip address to match,it never finds a match and a no gateway message appears on the terminal due to this.

v. From PC2 ping PC4: PC2 and PC4 are in different subnets. PC2's IP address is in the subnet 10.0.1.101 / 28, and PC4's IP address is in the subnet 10.0.1.121 / 28. Similar to earlier scenarios, the ping operation was unsuccessful because they are not on the same subnet.No gateway message is seen on the terminal.\*\*\*\*why?When PC2 is looking for the first 28 bits of the ip address to match,it never finds a match and a no gateway message appears on the terminal due to this.

vi. From PC2 ping PC3: PC2(10.0.1.101/28) and PC3(10.0.1.120/24) do not have IP addresses in the same /24 subnet.The ping is unsuccessful.No gateway message is seen on the terminal.\*\*\*\*why?When PC2 is looking for the first 28 bits of the ip address to match,it never finds a match and a no gateway message appears on the terminal due to this.