|  |  |
| --- | --- |
|  | CS 542 - Assignment 2 |
|  |  |
|  | Sukanta Sharma (A20472623), Vidya Sudharshana (A20472468), Vidhi Kakini (A20473969)  CS 542 – Computer Networks – I | Spring 2021 | Illinois Institute of Technology  5/14/21 |

# **Instructions:**

* Please submit soft copies on the blackboard.
* Team submissions are accepted. A team of 1 – 3 is accepted.
* For team submissions, one submission is sufficient. Anyone from the team can submit—no need for everyone in the group to submit.
* All should submit typewritten documents. The handwritten ones are not accepted. Zero points will be awarded if the submission is not a typewritten one.
* Please contact **Viswatej Kasapu (vkasapu@hawk.iit.edu)** if something is not clear. After submission, please do not say any excuses like "we understood differently." If you doubt any questions, please email me but do not expect me to give ideas or hints to the solution.
* **The due date is Friday, May 14, 2021, at 11:59 PM (midnight) Central Time.**
* Submissions after the due date are not accepted. This is the hard deadline. I must submit the grades to the University before the deadline. So, I cannot provide extensions to you. **Please submit before the due date without** **fail. Otherwise, zero points will be given.**

# **Instructions:**

* Please submit soft copies on the blackboard.
* For every question which has a calculation, you should show steps clearly. No points will be given for direct answers. Explanation and justification are mandatory.
* In all questions, provide answers in the decimal system (not binary, hexadecimal, or 256 bases).

1. A host with IP address 130.23.43.20 and physical address B2:34:55:10:22:10 has a packet to send to another host on another network with IP address 141.23.56.21 and physical address A4:6E:F4:59:83:AB. The next-hop (router) for this destination in the sender's routing table is Router R1 with IP address 130.23.43.25 and physical address B2:53:45:01:33:10. Give the ARP request packet format from the sender and its corresponding reply packet format filled with all necessary fields. Consider the Ethernet as hardware type and IPv4 as protocol type. **(10 points)**

Ans:

The structure of ARP Packet is given as below:



Figure : ARP Packet

## ARP Request Packet:

The ARP Request Packet is given as follows:

|  |  |  |
| --- | --- | --- |
| 0x0001 | | 0x0800 |
| 0x06 | 0x04 | 0x0001 |
| 0xB23455102210 | | |
| 0x82172B14 | | |
| 0x000000000000 | | |
| 0x82172B19 | | |

## ARP Reply Packet:

The ARP Request Packet is given as follows:

|  |  |  |
| --- | --- | --- |
| 0x0001 | | 0x0800 |
| 0x06 | 0x04 | 0x0002 |
| 0xB25345013310 | | |
| 0x82172B19 | | |
| 0xB23455102210 | | |
| 0x82172B14 | | |

1. Consider the updated ARP cache table at time t. The maximum number of attempts is 10, and the time-out value is 600 seconds. After 120 seconds, the input module receives two ARP packets, and the output module receives one IP packet from IP software. These are the only three packets host received in the last 120 seconds. Consider cache table is updated every 60 seconds. Give the updated cache table at times t+60 seconds and t+120 seconds, respectively. **(10 points)**

Packets Received:

* An ARP reply from the host with IP address 114.5.7.89 and physical address 457342ACAE32
* An ARP reply from the host with IP address 201.11.56.7 and physical address A46EF45983BC
* An IP packet that has to be forwarded to the next hop with IP address 188.11.8.71

| **State** | **Queue** | **Attempt** | **Time-out** | **Protocol Address** | **Hardware Address** |
| --- | --- | --- | --- | --- | --- |
| R | 5 |  | 500 | 180.3.6.1 | ACAE32457342 |
| P | 2 | 2 |  | 129.34.4.8 |  |
| P | 14 | 7 |  | 201.11.56.7 |  |
| R | 8 |  | 60 | 114.5.7.89 | 457342ACAE32 |
| F |  |  |  |  |  |

Ans:

## Cache Table at time t+60 seconds:

For every entry in the cache table

1. For the first entry the state is RESOLVED, so it will decrease the value of the timeout by 60 seconds (i.e., 500 - 60 = 440 seconds).
2. For the second entry the state is PENDING, so it will increase the value of the attempt by 1 (i.e., 2 + 1 = 3) and send an ARP request.
3. For the third entry the state is PENDING, so it will increase the value of the attempt by 1 (i.e., 7 + 1 = 8) and send an ARP request.
4. For the fourth entry the state is RESOLVED, so it will decrease the value of the timeout by 60 seconds (i.e., 60 – 60 = 0 seconds). The updated timeout value is equal to zero, so the state of the entry is changed to FREE, and the corresponding queue is destroyed.
5. For the fifth entry the state is FREE, so it will continue.

The final updated cache table after t + 60 seconds is given below:

| **State** | **Queue** | **Attempt** | **Time-out** | **Protocol Address** | **Hardware Address** |
| --- | --- | --- | --- | --- | --- |
| R | 5 |  | 440 | 180.3.6.1 | ACAE32457342 |
| P | 2 | 3 |  | 129.34.4.8 |  |
| P | 14 | 8 |  | 201.11.56.7 |  |
| F |  |  |  |  |  |
| F |  |  |  |  |  |

## Cache Table at time t+120 seconds:

* **Cache Control Module:**

For every entry in the cache table:

1. For the first entry the state is RESOLVED, so it will decrease the value of the timeout by 60 seconds (i.e., 440 – 60 = 380 seconds).
2. For the second entry the state is PENDING, so it will increase the value of the attempt by 1 (i.e., 3 + 1 = 4) and send an ARP request.
3. For the third entry the state is PENDING, so it will increase the value of the attempt by 1 (i.e., 8 + 1 = 9) and send an ARP request.
4. For the fourth entry the state is FREE, so it will continue.
5. For the fifth entry the state is FREE, so it will continue.

* **Input Module:**

1. The input module receives an ARP packet with target protocol (IP) address 114.5.7.89 and physical address 457342ACAE32. The module checks the table and finds this address. The module checks the table and does not find this address. It will create a cache entry with state set to RESOLVED and timeout value set to 600 seconds and add the entry to the table with the received protocol and hardware address.
2. The input module receives an ARP packet with target protocol (IP) address 201.11.56.7 and physical address A46EF45983BC. The module checks the table and finds this address. It changes the state of the corresponding entry to RESOLVED, attempt value is deleted, and sets the timeout value to 600 seconds. The module then adds the target hardware address (i.e., A46EF45983BC) to the entry. Now it accesses queue 14 and send all the packets in this queue, one by one, to the data link layer.

* **Output Module:**

The output module receives and IP packet with the next hop address 188.11.8.71. It checks the cache table and does not find this address in the table. The module adds an entry to the table with the state PENDING and the attempt value 1. It creates a new queue number 15 for this destination and enqueues the packet. It then sends and ARP request to the data link layer for this destination.

The final updated cache table after t + 120 seconds is given below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **State** | **Queue** | **Attempt** | **Time-out** | **Protocol Address** | **Hardware Address** |
| R | 5 |  | 380 | 180.3.6.1 | ACAE32457342 |
| P | 2 | 4 |  | 129.34.4.8 |  |
| R | 14 |  | 600 | 201.11.56.7 | A46EF45983BC |
| R |  |  | 600 | 114.5.7.89 | 457342ACAE32 |
| P | 15 | 1 |  | 188.11.8.71 |  |

1. For each one, mention whether it is a valid or invalid value for the HLEN field in the IP datagram header. Give your supporting reasons. **(4 points)**
2. 1011

Ans:

The header length should be between 20 and 60 bytes. HLEN field contains the value in 4-byte word format in binary. So, the header length is . So, the given value for the HLEN field is valid.

1. 1201

Ans:

The header length should be between 20 and 60 bytes. HLEN field contains the value in 4-byte word format in binary. The given value for the HLEN field is invalid, because the given value is not in binary format.

1. 0011

Ans:

The header length should be between 20 and 60 bytes. HLEN field contains the value in 4-byte word format in binary. So, the header length is . So, the given value for the HLEN field is invalid.

1. 0101

Ans:

The header length should be between 20 and 60 bytes. HLEN field contains the value in 4-byte word format in binary. So, the header length is . So, the given value for the HLEN field is valid.

1. In an IP packet, the value in the HLEN field is 1100, and the value of the total length is 111111000. How many bytes of data is the packet carrying? Are there any options? If so, what is the length of the options? **(3 points)**

Ans:

HLEN value =

Header Length =

Total Length =

Data Length =

* 456 bytes of data is the packet carrying.
* Minimum header length is 20 bytes, and the given header length is greater than that. So, yes, there are options present, and the length of the options is

1. The total IP datagram length is 70 bytes, out of which data length is 34 bytes. Is this example a valid IP datagram or not? Give your supporting reasons. **(2 points)**

Ans:

IP datagram total length = 70 bytes

Data length = 34 bytes

Header length = , which is between 20 and 60 bytes, so this is a valid example of IP datagram.

1. An IP datagram is divided into three fragments. All fragments are equal in size and have a base header of 20 bytes. The size of data in each fragment is 800 bytes. The first and last fragments can be divided further, but the second cannot be fragmented further. Give D, M, and fragmentation offset values of each fragment. (4 points)

Ans:

The size of data in each fragment is 800 bytes, so the data bytes send in the first fragment will be from 000 to 799, in the second fragment will be from 800 to 1599, and in the third fragment will be from 1600 to 2399. If D value is 1, then the datagram must not fragment further and if D value is 0, then the datagram can be fragment further. If M value is 1, then the datagram is not the last datagram and if M value is 0, then the datagram is the last datagram.

The values for D, M, and fragmentation offset for each fragment is given below:

| **#Fragment** | **D** | **M** | **Fragmentation Offset** |
| --- | --- | --- | --- |
| 1 | 0 | 1 |  |
| 2 | 1 | 1 |  |
| 3 | 0 | 0 |  |

1. A fragment has arrived with the first few hexadecimal digits, as shown below:

4500 003C 0001 8370……

This is the second fragment. How many bytes of data does this fragment contain? What is the offset of the next fragment? **(3 points)**

Ans:

The format of IP datagram is given as below:



Figure :IP Datagram

From the above figure and the received data in hexadecimal digit, we have the following information:

HLEN =

Header Length =

Total Length =

Data Length =

* 40 bytes of data this fragment contains.

The flag and fragmentation offset =

D =

M = , it means this datagram is the last datagram.

The fragmentation offset =

* As we can see, the value of M bit is 0, that means this the last fragment, so there will not be any next fragment.

1. The first 32 bits of an IP datagram are shown below. Is it a valid IP datagram? Explain your answer? **(2 points)**

0001 1010 0000 0000 0000 0000 0001 1110…

Ans:

From the Fig 2 and the given data we can have the following information:

HLEN =

Header Length =

Total Length =

From the above information we can see that the total length is less than the header length, so it is not a valid IP datagram.

1. An IP packet has arrived with the first few hexadecimal digits as shown below:

4600 0040 0001 0000 1217……

The initial 'Time to Live" value in the hexadecimal format is BC. How many hops have this packet already traveled? How many hops can this packet travel before being dropped? **(3 points)**

Ans:

From the Fig 2 and the given data we can have the following information:

Received Time to Live =

Initial Time to Live

* This packet has travelled already.
* This packet can travel more before dropped.

To convert to base-256 number system, we need to convert the integer part and the fraction part separately to base-256 and then we can combine them to obtain the result.

## Integer Part 🡪 :

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 5141 | 20 | 21 |
| 256 | 20 | 0 | 20 |

Therefore, 🡪

## Fraction Part 🡪 :

| **Multiplicand** | **Multiplier** | **Result** | **Integer Part** |
| --- | --- | --- | --- |
| 0.015686035 | 256 | 4.015625 | 4 |
| 0.015625 | 256 | 4 | 4 |

Therefore, 🡪 *[“,” is used as decimal point notation and “.” as a separator of base-256 digits]*

Now, concatenating both the result we can write, 🡪 *[“,” is used as decimal point notation and “.” as a separator of base-256 digits]*

1. What is the value of ? Give results in 256 base system. (Given numbers are in 256 base system) (4 points)

Ans:

## Conversion of to base-256 number system:

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 768 | 3 | 0 |
| 256 | 3 | 0 | 3 |

Therefore, is equivalent to

1. An organization is granted the block . The administrator wants to create 16 subnets.
2. Find the subnet mask (1 point).

Ans:

The block given to us is . The administrator wants to create 16 subnets. So, the number of extra bits that need to be added to the default mask is .

The subnet mask will be, *[in slash notation]* or *[in dotted-decimal notation].*

1. Find the number of addresses in each subnet (1 point)

Ans:

Here the number of bits of prefix *(same as the mask)*

The number of bits of suffix

The number address in each subnet is given by,

Therefore, the number of addresses in each subnet is  **addresses.**

1. Find the subnet address and the direct broadcast address for the first subnet. (2 points)

Ans:

## Subnet Address:

The subnet address is the **first address of the given subnet**. The first address of the first subnet is nothing but the first address of the given network, that is and the mask for this subnet is . So, the subnet address of the first block is **.**

## Direct-Broadcast Address:

The direct-broadcast address is the **last address of the given subnet**. There are addresses in each subnet. So, to find the last address of the given subnet we need to add to the first address of the subnet in the base-256 number system.

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 142 | 200 | 208 | 0 |
| + |  |  |  | 127 |
| **Result** | **142** | **200** | **208** | **127** |

Therefore, the direct-broadcast address is **.**

1. Find the 4th and 99th addresses in the last subnet. (4 points)

Ans:

## 4th address:

There are addresses in each subnet. So, the 4th address of the last subnet (i.e., 16th subnet) is given by,

address

address

address

So, to get the 4th address of the 16th subnet we need to add to the first address of the network in the base-256 number system.

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 1923 | 7 | 131 |
| 256 | 3 | 0 | 7 |

Therefore, 🡪

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 142 | 200 | 208 | 0 |
| + |  |  | 7 | 131 |
| **Result** | **142** | **200** | **215** | **131** |

So, the 4th address of the last subnet is

## 99th address:

There are addresses in each subnet. So, the 99th address of the last subnet (i.e., 16th subnet) is given by,

address

address

address

So, to get the 99th address of the 16th subnet we need to add to the first address of the network in the base-256 number system.

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 1923 | 7 | 226 |
| 256 | 7 | 0 | 226 |

Therefore, 🡪

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 142 | 200 | 208 | 0 |
| + |  |  | 7 | 226 |
| **Result** | **142** | **200** | **215** | **226** |

So, the 99th address of the last subnet is

1. Give the mask in the dotted-decimal notation:
2. For a block of Class-A which results in 128 subnets (1 point)

Ans:

For class A, the default mask is . To have 128 subnets, it requires extra one’s bit. So, the mask, for a block of class A which results in subnets, is *[in slash notation]* or *[in dotted-decimal notation].*

1. Which combines 128 blocks of Class C into a supernet (1 point)

Ans:

For class C, the default mask is . To combine 128 blocks into a supernet, it requires less one’s bit. So, the mask, which combines blocks of class C into a supernet, is *[in slash notation]* or *[in dotted-decimal notation].*

1. Convert an IP address 254.128.64.32 to the binary notation (2 points)

Ans:

To convert the IP address to the binary notation, we need to convert each byte into binary. Then we can concatenate the binary numbers together to get the binary notation of the given IP address.

## Decimal to Binary Conversion:

| **#Byte** | **Value (in Decimal)** | **Value (in Binary)** |
| --- | --- | --- |
| 1 | 254 | 11111110 |
| 2 | 128 | 10000000 |
| 3 | 64 | 01000000 |
| 4 | 32 | 00100000 |

The equivalent IP address of in binary notation is given as,

**11111110 10000000 01000000 00100000.**

1. The 14th address of a block assigned to a specific organization is . The organization needs 120 addresses to give to its 120 users. Find the mask and define this block of addresses. Is there any wastage of the IP addresses? If yes, how many? (Note: The number of router interfaces is 2) (4 points)

Ans:

## Mask:

The organization is given a block of and it needs 120 address to give its 120 users, so the number of suffix bits is . Therefore, the mask for the block is *[in slash notation]* or *[in dotted-decimal notation]*.

## Block Definition:

To define the given block, we need to find the first and the last address of the block.

## First Address:

The 14th address of a block is . To find the first address of the block we need to subtract from the given address in the base-256 number system.

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 65 | 89 | 141 |
| - |  |  |  | 13 |
| **Result** | **120** | **65** | **89** | **128** |

The first address of the block is .

## Last Address:

The mask for the block is , which means the number of prefix bits is and the number of suffix bits is . So, the total number of addresses in the block is . To find the last address we need to add to the first address of the block in the base-256 number system.

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 65 | 89 | 128 |
| + |  |  |  | 127 |
| **Result** | **120** | **65** | **89** | **255** |

Therefore, the last address of the block is **.**

The given block can be defined as a block with  **addresses** and the range from **to** **.**

## IP address wasted:

The total number of addresses in the given block is . The address allocation in the organization is given by,

* **120 addresses** are given to its 120 users.
* **1 address** is used as the network address, which is the first address of the block.
* **1 address** is used as the direct-broadcast address, which is the last address of the block.
* The number of router interfaces is two, so **2 addresses** are used to identify these two interfaces.

The number of IP addresses used is and the number of addresses wasted is **.**

1. A block of addresses granted to an ISP. These addresses are allocated between two groups of customers. The first group has 20 customers, each of which needs 64 addresses, the second group has 20 customers, each of which needs 128 addresses. Show the subblocks and range of addresses for the 10th customer of the first group and the 10th customer of the second group. How many addresses are still available after this allocation? (5 points)

Ans:

From the above information, we can conclude that the address block need to be divided into the first group with subnets with 64 addresses in each subnet and subnets with addresses in each subnet.

## First Group:

There are addresses in each subnet. So, the number of suffix bits is and the prefix bits are . Therefore, the mask for each of these blocks is *[in slash notation]*.

So, the first address of the 10th customer or the 10th subnet is given by,

address

address

address

So, to get the first address of the 10th subnet we need to add to the first address of the block in the base-256 number system.

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 576 | 2 | 64 |
| 256 | 2 | 0 | 2 |

Therefore, 🡪

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 200 | 240 | 0 |
| + |  |  | 2 | 64 |
| **Result** | **120** | **200** | **242** | **64** |

So, the first address of the 10th subnet is

Now, to get the last address of the 10th subnet we need to add to the first address of the subnet in the base-256 number system.

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 200 | 242 | 64 |
| + |  |  |  | 63 |
| **Result** | **120** | **200** | **242** | **127** |

Therefore, the last address of this subnet is

Therefore, the subblock for the 10th customer of the first group has  **addresses** and the range of the subblock is from  **to**

## Second Group:

There are addresses in each subnet. So, the number of suffix bits is and the prefix bits are . Therefore, the mask for each of these blocks is *[in slash notation]*.

So, the first address of the 10th customer of the second group is given by,

addresses

address

address

So, to get the first address of the 10th customer we need to add to the first address of the block in the base-256 number system.

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 2432 | 9 | 128 |
| 256 | 9 | 0 | 9 |

Therefore, 🡪

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 200 | 240 | 0 |
| + |  |  | 9 | 128 |
| **Result** | **120** | **200** | **249** | **128** |

So, the first address of the 10th customer in the second group is

Now, to get the last address of the 10th customer in the second group we need to add to the first address of this subnet in the base-256 number system.

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 120 | 200 | 249 | 128 |
| + |  |  |  | 127 |
| **Result** | **120** | **200** | **249** | **255** |

Therefore, the last address of this subnet is

Therefore, the subblock for the 10th customer of the second group has  **addresses** and the range of the subblock is from  **to**

## Available Address:

The ISP granted the IP address with mask . So, the number of suffix bits is .

There is a total of addresses for the granted block and the total number address used is given by,

addresses

addresses

Therefore, there are  **addresses**  still available after this allocation.

1. Find first address, last address, and number of addresses in the block, if one of the addresses in a block is (3 points)

Ans:

## First Address:

To find the first address or network address we need to perform a logical **AND** operation between the given IP address and the mask. As the mask is , so we can perform this **AND** operation with the left 4 bits of the third byte from the left. The third byte is given as or .

Now,

In the network address first 16 bits will be the same as the address given *(i.e., ),* the third byte is and the rest of the bits will be all zeros.

Therefore, the network address or the first address is **.**

## Last Address:

One of the addresses in a block is given as . So, the number of suffix bits is . The number of suffix bits is . So, the total number of addresses is . To find the last address we need to add to the first address of the block in the base-256 number system.

| **Divisor** | **Dividend** | **Quotient** | **Remainder** |
| --- | --- | --- | --- |
| 256 | 4096 | 15 | 255 |
| 256 | 15 | 0 | 15 |

Therefore, 🡪

| **+/-** | **Byte1** | **Byte2** | **Byte3** | **Byte4** |
| --- | --- | --- | --- | --- |
|  | 140 | 240 | 80 | 0 |
| + |  |  | 15 | 255 |
| **Result** | **140** | **240** | **95** | **255** |

So, the last address of the block is

## Number of Addresses:

One of the addresses in a block is given as . So, the number of suffix bits is . The number of suffix bits is . So, the total number of addresses is **.**

1. Consider the following routing table (the next-hop address is omitted):

| **Mask** | **Network Address** | **Interface** |
| --- | --- | --- |
| /27 | 144.56.55.0 | M0 |
| /26 | 123.80.97.0 | M1 |
| /25 | 123.80.97.128 | M2 |
| /24 | 118.114.132.0 | M3 |
| Default | Default | M4 |

Give the interface number for a packet whose destination IP address is:

1. 144.56.55.31 (1 point)

Ans:

To find the network address of the given IP address, we need to apply the masks in descending order. If the resultant network address is found, then the corresponding interface is chosen and if the resultant network address cannot be found in the routing table the default interface is chosen.

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 144.56.55.31 | /27 | 3110 🡪 000101112 | 111000002 | 00000000­2 🡪 010 | 144.56.55.0 |

As the resultant network address for mask is found in the routing table, so the interface number chosen is **M0**.

1. 144.56.56.31 (1 point)

Ans:

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 144.56.56.31 | /27 | 3110 🡪 000101112 | 111000002 | 00000000­2 🡪 010 | 144.56.56.0 |
| 144.56.56.31 | /26 | 3110 🡪 000101112 | 110000002 | 00000000­2 🡪 010 | 144.56.56.0 |
| 144.56.56.31 | /25 | 3110 🡪 000101112 | 100000002 | 00000000­2 🡪 010 | 144.56.56.0 |
| 144.56.56.31 | /24 | 3110 🡪 000101112 | 000000002 | 00000000­2 🡪 010 | 144.56.56.0 |

As the resultant network address for all masks is not found in the routing table, so the interface number chosen is **M4**.

1. 123.80.97.60 (1 point)

Ans:

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 123.80.97.60 | /27 | 6010 🡪 001111002 | 111000002 | 00100000­2 🡪 3210 | 123.80.97.32 |
| 123.80.97.60 | /26 | 6010 🡪 001111002 | 110000002 | 00000000­2 🡪 010 | 123.80.97.0 |

As the resultant network address for mask is found in the routing table, so the interface number chosen is **M1**.

1. 123.80.97.200 (1 point)

Ans:

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 123.80.97.200 | /27 | 20010 🡪 110010002 | 111000002 | 11000000­2 🡪 19210 | 123.80.97.192 |
| 123.80.97.200 | /26 | 20010 🡪 110010002 | 110000002 | 11000000­2 🡪 19210 | 123.80.97.192 |
| 123.80.97.200 | /25 | 20010 🡪 110010002 | 100000002 | 10000000­2 🡪 12810 | 123.80.97.128 |

As the resultant network address for mask is found in the routing table, so the interface number chosen is **M2.**

1. 123.80.97.88 (1 point)

Ans:

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 123.80.97.88 | /27 | 8810 🡪 010110002 | 111000002 | 01000000­2 🡪 6410 | 123.80.97.64 |
| 123.80.97.88 | /26 | 8810 🡪 010110002 | 110000002 | 01000000­2 🡪 6410 | 123.80.97.64 |
| 123.80.97.88 | /25 | 8810 🡪 010110002 | 100000002 | 00000000­2 🡪 010 | 123.80.97.0 |
| 123.80.97.88 | /24 | 8810 🡪 010110002 | 000000002 | 00000000­2 🡪 010 | 123.80.97.0 |

As the resultant network address for all masks is not found in the routing table, so the interface number chosen is **M4**.

1. 118.114.133.1 (1 point)

Ans:

The masks are applied as given below:

| **Destination IP Address** | **Mask** | **IP Address in binary (only in the last byte) [A]** | **Mask in binary (only in the last byte) [B]** | **[A] AND [B]** | **Network Address** |
| --- | --- | --- | --- | --- | --- |
| 118.114.133.1 | /27 | 110 🡪 000000012 | 111000002 | 00000000­2 🡪 010 | 118.114.133.0 |
| 118.114.133.1 | /26 | 110 🡪 000000012 | 110000002 | 00000000­2 🡪 010 | 118.114.133.0 |
| 118.114.133.1 | /25 | 110 🡪 000000012 | 100000002 | 00000000­2 🡪 010 | 118.114.133.0 |
| 118.114.133.1 | /24 | 110 🡪 000000012 | 000000002 | 00000000­2 🡪 010 | 118.114.133.0 |

As the resultant network address for all masks is not found in the routing table, so the interface number chosen is **M4**.

1. The routing table of routers R1, R2, and R3 are given. Draw the possible network configuration with all 3 routers, not separate configurations corresponding to each routing table. Indicate the next-hop addresses in the figure. (10 points)

R1:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 80.70.56.0 | 100.160.32.67 | M2 |
| /24 | 130.135.7.0 | 150.137.45.78 | M1 |
| /16 | 180.170.0.0 | ----------------- | M0 |
| /16 | 100.160.0.0 | ----------------- | M2 |
| /16 | 150.137.0.0 | ----------------- | M1 |
| Default | Default | 180.170.4.6 | M0 |

R2:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 80.70.56.0 | ----------------- | M0 |
| /16 | 100.160.0.0 | ----------------- | M1 |
| Default | Default | 100.160.56.7 | M1 |

R3:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 130.135.7.0 | ----------------- | M0 |
| /16 | 150.137.0.0 | ----------------- | M1 |
| Default | Default | 150.137.72.48 | M1 |

Ans:

Considering the above routing tables for R1, R2, and R3, the possible network configuration diagram with all 3 routers is shown as below:



Figure : Network Diagram

1. Consider the network configuration below. A packet arrived at the router R3 with the destination address . Show how it is forwarded. (Assume classless addressing and mask of each network is ) Create a routing table for R1 and R3. (10 points)Ans:

The routing table is given as below:

## The routing table for router R1:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 150.14.0.0 | ----------------- | M2 |
| /24 | 133.79.0.0 | ----------------- | M1 |
| /24 | 129.101.0.0 | ----------------- | M0 |
| /24 | 192.180.0.0 | 129.101.17.32 | M0 |
| Default | Default | 129.101.31.18 | M0 |

## The routing table for router R2:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 150.14.0.0 | 129.101.19.20 | M1 |
| /24 | 133.79.0.0 | 129.101.19.20 | M1 |
| /24 | 190.180.7.9 | 129.101.19.20 | M1 |
| /24 | 129.101.0.0 | ----------------- | M1 |
| Default | Default | ----------------- | M0 |

## The routing table for router R3:

| **Mask** | **Network Address** | **Next-Hop Address** | **Interface Number** |
| --- | --- | --- | --- |
| /24 | 190.180.0.0 | ----------------- | M0 |
| /24 | 129.101.0.0 | ----------------- | M1 |
| /24 | 150.14.0.0 | 129.101.19.20 | M1 |
| /24 | 133.79.0.0 | 129.101.19.20 | M1 |
| Default | Default | 129.101.31.18 | M1 |

## Packet Forwarding:

The packet is at router R3 with destination address as *[in dotted-decimal notation]* or *[in binary notation]*. The mask is given as . The network address is extracted by masking off the leftmost bits of the destination address; the result is **.** The resultant network does not match with any entry in the routing table of router R3, so the default entry is chosen. The next-hop address and the interface **M1** are passed to ARP. After this, the packet is forwarded to router **R2.**

Now, the packet is at router R2 with destination address as *[in dotted-decimal notation]* or *[in binary notation]*. The mask is given as . The network address is extracted by masking off the leftmost bits of the destination address; the result is **.** The resultant network does not match with any entry in the routing table of router R3, so the default entry is chosen. The interface **M0** is passed to ARP. After this, the packet is forwarded to the **Rest of the Internet.**