CAT-2

Computer Networks

Unit 3

Network Layer

* 3-3 DHCP

DHCP = Dynamic Host Configuration Protocol

→ A large organization or an ISP can receive a block of addresses directly from ICANN, and a small organization can receive a block of addresses from an ISP.

→ The network administrator manually assigns addresses to the individual hosts or routers.

→ However, address assignment in an organization can be done automatically using the Dynamic Host Configuration Protocol (DHCP).

→ DHCP is an application-layer program, using the client-server paradigm, that helps the TCP/IP at the network layer

→ called a plug-and-play protocol.

Usage of DHCP

(i) to assign permanent IP addresses to host & routers
(ii) to provide temporary, on demand addresses to hosts & routers. (eg. connect laptop to hotel wifi)

→ DHCP provides the following information to hosts:

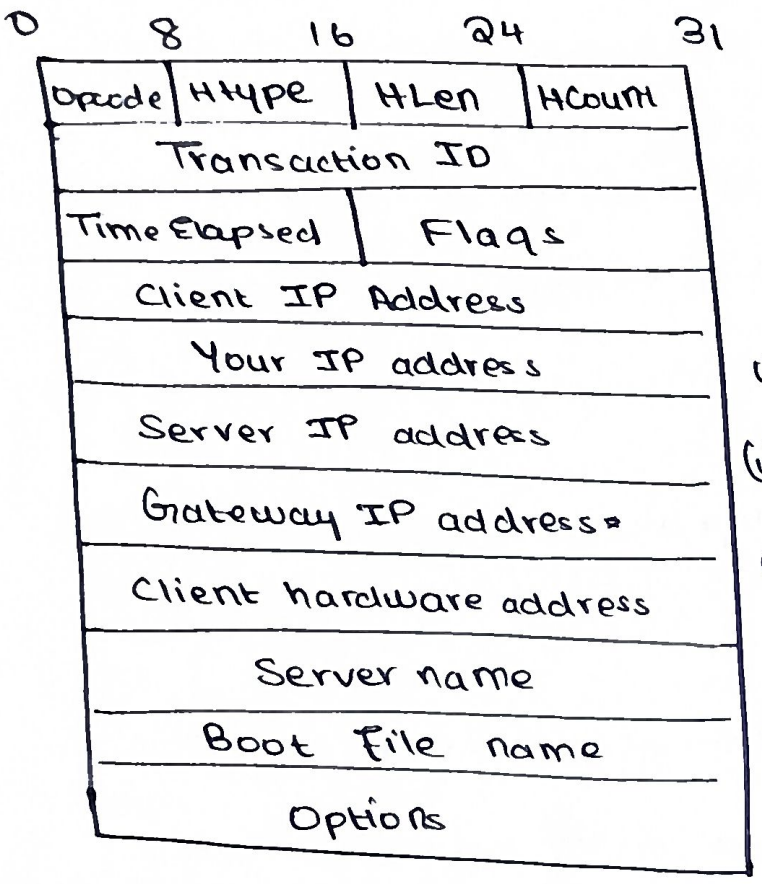
(i) IP address

(ii) prefix

(iii) address of a router

(iv) IP address of a name server

* DHCP Message Format



Field Description

(i) opcode = operation code

request = 1

reply = 2

(ii) Htype: hardware type (Ethernet..)

(iii) Hlen: length of hardware address

(iv) Hcount: max. no. of hops the packet can travel

(v) Transaction ID: an integer set by the client and repeated by the server.

(vi) Time elapsed: no. of seconds since the client started to boot

(vii) Flags: First bit defines

unicast = 0

multicast = 1

other 15 bits not used

(viii) Client IP address - set to 0 if client does not know it

(ix) Your IP address - client IP address sent by server

(x) Server IP address - a broadcast IP if the client does not know it.

(xi) Gateway IP address - address of default router

(xii) Server name - 64 byte domain name of server

(xiii) Boot file name - a 128 byte file name holding extra info.

(xiv) Options - has a dual purpose. Can either

(i) carry additional information

(ii) specific vendor information

→ The server uses a number called the magic cookie, in the form of an IP address with the value 99.130.83.99.

→ When the client finishes reading the message, it looks for this magic cookie. If it is present, the next 60 bytes are options.

→ An option has 3 fields:

53	1	
Tag	Length	Value

1 DHCPDISCOVER
2 DHCPOFFER
3 DHCPREQUEST
4 DHCPDECLINE
5 DHCPACK
6 DHCPNACK
7 DHCPRELEASE
8 DHCPINFORM

* Operation of DHCP

A. Joining Host creates a DHCPDISCOVER message

(i) set transaction ID to a random no.

(ii) cannot set any other field

(iii) encapsulate message in a UDP datagram

(iv) set source port = 68, dest port = 67

(v) encapsulate user datagram in an IP datagram

(source addr = 0.0.0.0, dest addr: 255.255.255.255)
broadcast

B. DHCP server or servers responds with a DHCPOFFER message.

(i) The 'your address' field defines the offered IP addres for the joining host.

(ii) also has server IP address of the server.

(iii) has the lease time for which the host can keep the IP address.

(iv) encapsulate message in a user datagram (in reverse order)

(v) encapsulate user datagram with the server address as the source IP, but the destination address is a broadcast address. This means that the server can allow other DHCP servers to receive the offer & give a better offer if they can.

c. Joining host receives one or more offers and chooses the best of them.

(i) Joining host then sends a DHCPREQUEST message to the server that has given the best offer.

(ii) Fields with known values are set.

(iii) encapsulate message in a user datagram, and then in an IP datagram (the source address is set to the new client address, but the destination address is set to the broadcast address, to let the other servers know that their offer wasn't accepted)

D. The selected server responds with a DHCPACK message to the client.

(i) Message is also broadcast to let other servers know that the request is accepted or rejected.

* Why does the DHCP use two well-known ports?

→ The well-known port 68 is used because the response from the server to the client is broadcast.

→ If there are 2 clients - say a DHCP & a DAYTIME client & the same temporary port no. is used, then while the DHCP client can process the message, the other can't.

Note: If 2 DHCP clients are running at the same time (after power failure/restoration), messages are distinguished by the transaction ID.

* Usage of FTP in DHCP

→ The server does not send all the information that a client may need for joining the network.

→ In the DHCP ACK message, the server defines the pathnames of a file in which the client can find the complete information such as the address of the DNS server.

→ The client then uses FTP to obtain the rest of the needed information.

* Error Control in DHCP

→ DHCP uses UDP, which is unreliable.

→ The 2 strategies used by DHCP for error control are:

(i) It requires that UDP uses the checksum

(ii) The DHCP client uses timers and a retransmission policy if it does not receive the DHCP reply to a request (However, to prevent a traffic jam when several hosts need to retransmit a request), DHCP forces the client to use a random number to set its timers).

* DHCP Transition States

1. When a DHCP client first starts it is in the INIT state.

2. The client broadcasts a DISCOVER message.

3. When it receives an offer, the client goes into the SELECTING state

4. After it selects one offer out of many, it goes into the REQUESTING state.

5. If an ACK arrives while it is in this state, it goes to the BOUND state, otherwise goes back to INIT and requests another IP address.

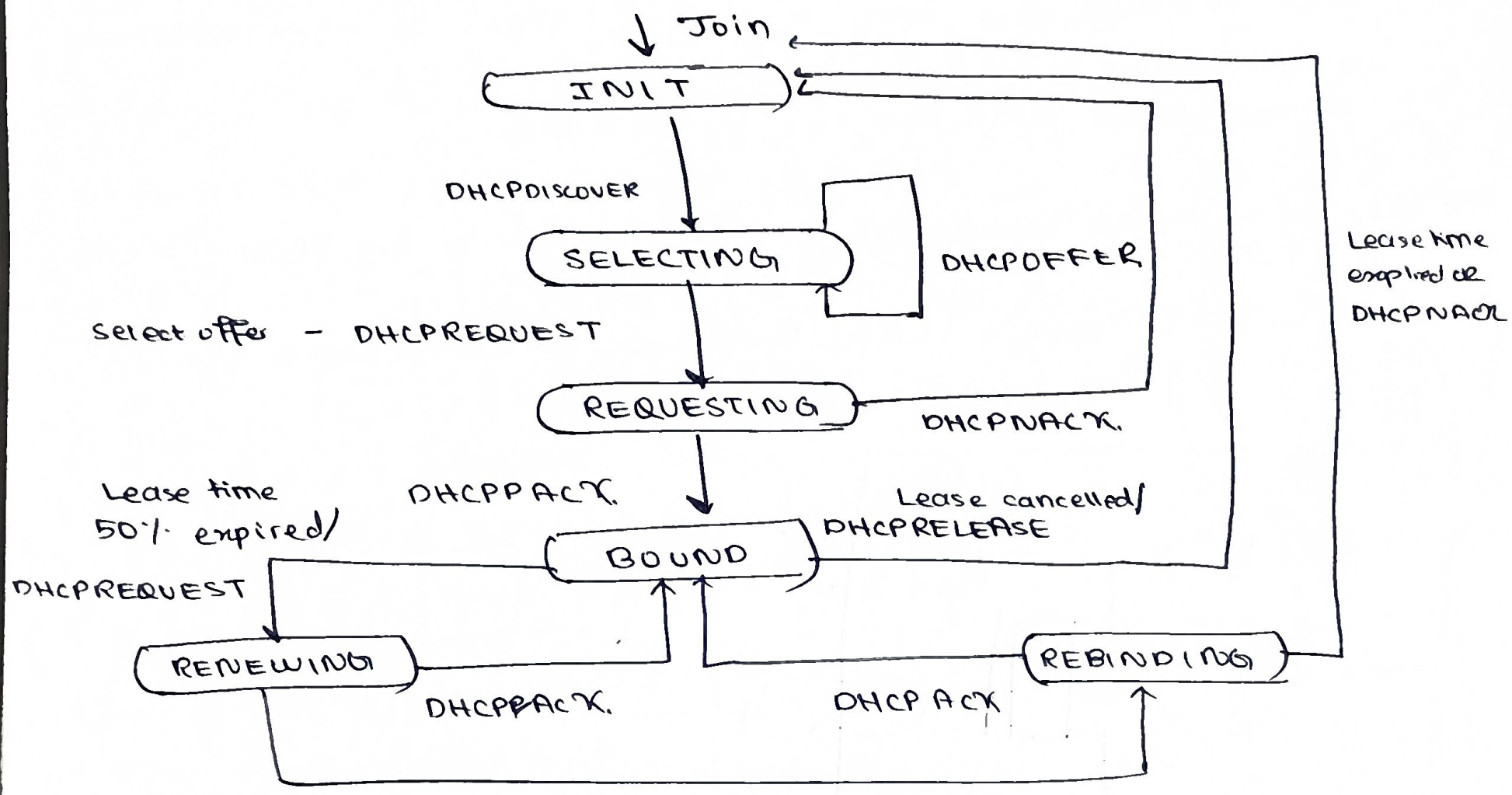
6. The client can use the IP address only when it is in the BOUND, RENEWING or REBINDING state.

7. For this, it uses 3 timers:

(i) renewal timer - 50% of lease time

(ii) rebinding timer - 75% " "

(iii) expiration timer - set to the lease time.



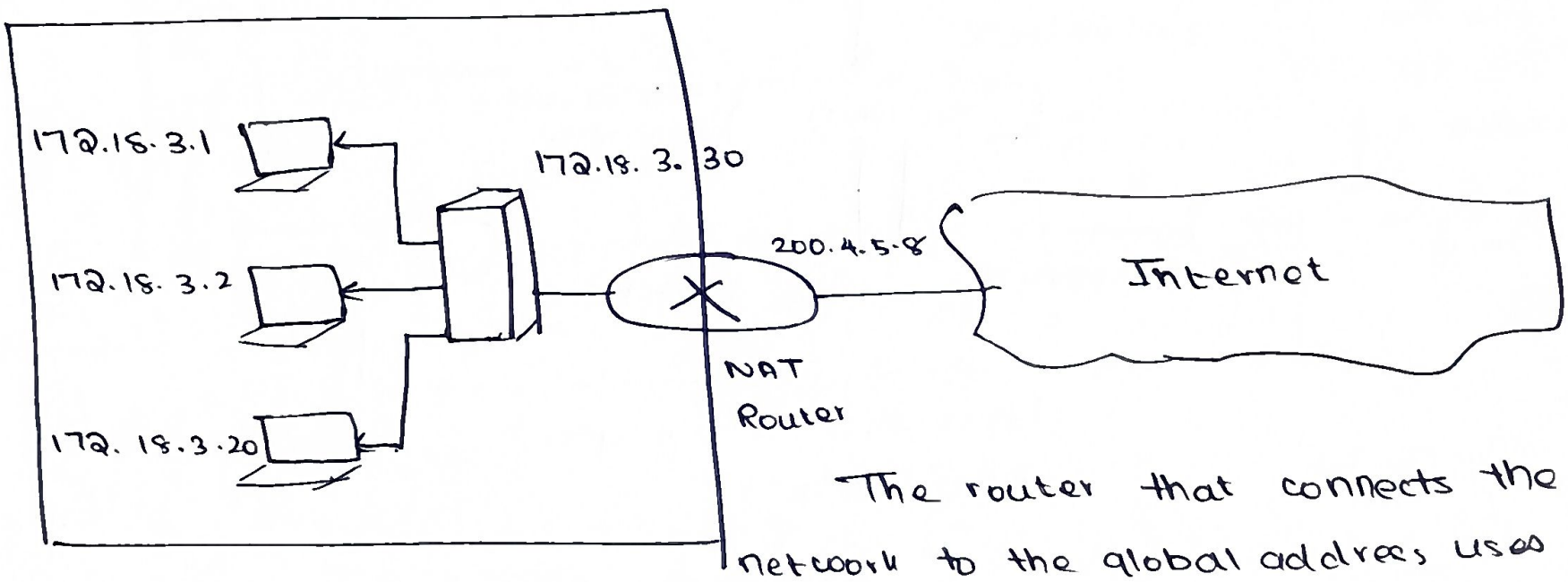
* NAT - Network Address Translation Resolution

(For context only: If a small business has been granted a range of addresses, if the business expands - new addresses may not be able to be given). Note that, in reality only a small fraction of the total computers may be using the internet, so that many addresses are not needed. The business can use multiple private addresses for internal communication & fewer addresses for universal comm. which is assigned by the ISP).

→ NAT is a technology that can provide a mapping between private and universal addresses, and also support virtual private networks.

→ This allows a site to use a set of private addresses for internal communication, and a set of global internet addresses for communication with the rest of the world.

→ The site must have only one connection to the global Internet through a NAT-capable router that runs NAT software.



The router that connects the network to the global address uses one private address & one global address. The private network is invisible to the rest of the Internet.

* Address Translation with NAT

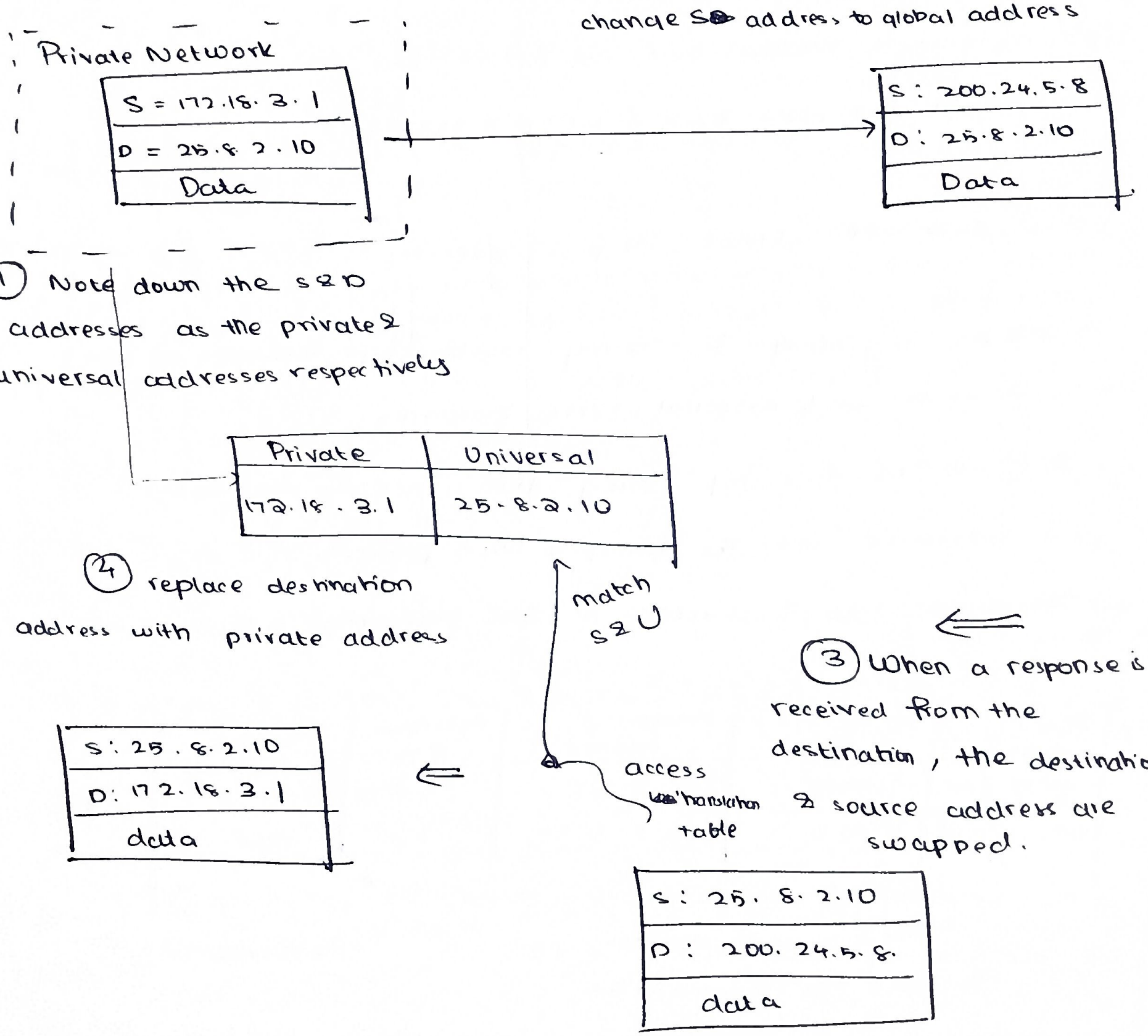
→ All of the outgoing packets go through the NAT router, which replaces the source address in the packet with the global NAT addressed.

Usage of a Translation Table

(1) Using one IP address

→ The simplest form of a translation table has only 2 columns - the private address and the external address (dest. address)

→ Consider the following example of sending a packet to an address & receiving an acknowledgement.



(2) Using a pool of IP addresses

→ If there is only one global address, then only one private network host can access a given external host.

→ To remove this restriction, the NAT router can use a pool of global addresses.

Drawbacks

(i) limit on address pool

(ii) no private network host can access two external server programs at the same time (HTTP & TELNET)

(3) Using both IP addresses & port addresses

→ allows a many-to-many relationship between private network hosts and external server programs.

→ This can be done by using the source & destination port addresses and the transport layer protocol.

→ There are now 5 columns in the translation table.

Private address	Private port	External address	External port	Transport protocol
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP

These must be unique to send the response to the right address.

* Forwarding of IP Packets

Forwarding: To deliver the packet to its next hop - which can be the final destination or the immediate connecting device.

→ Forwarding of IP packets can happen in 2 ways:

(i) IP as a connectionless protocol - forwarding is based on the destination address of the IP datagram

(ii) IP as a connection-oriented protocol - forwarding is based on the label attached to the IP datagram.

* (1) Forwarding Based on the Destination Address

→ Forwarding requires a host or router to have a forwarding table.

→ When a host has a packet to send / a router has a packet to be forwarded - look at the forwarding table.

→ called as classless address - since the address space is one entity, there are no classes.

→ There is one row for each of the blocks.

→ A classless forwarding table has the following pieces of info:

(i) network address
(ii) mask
(iii) next hop IP address
(iv) interface

Searching the ~~network~~ forwarding table

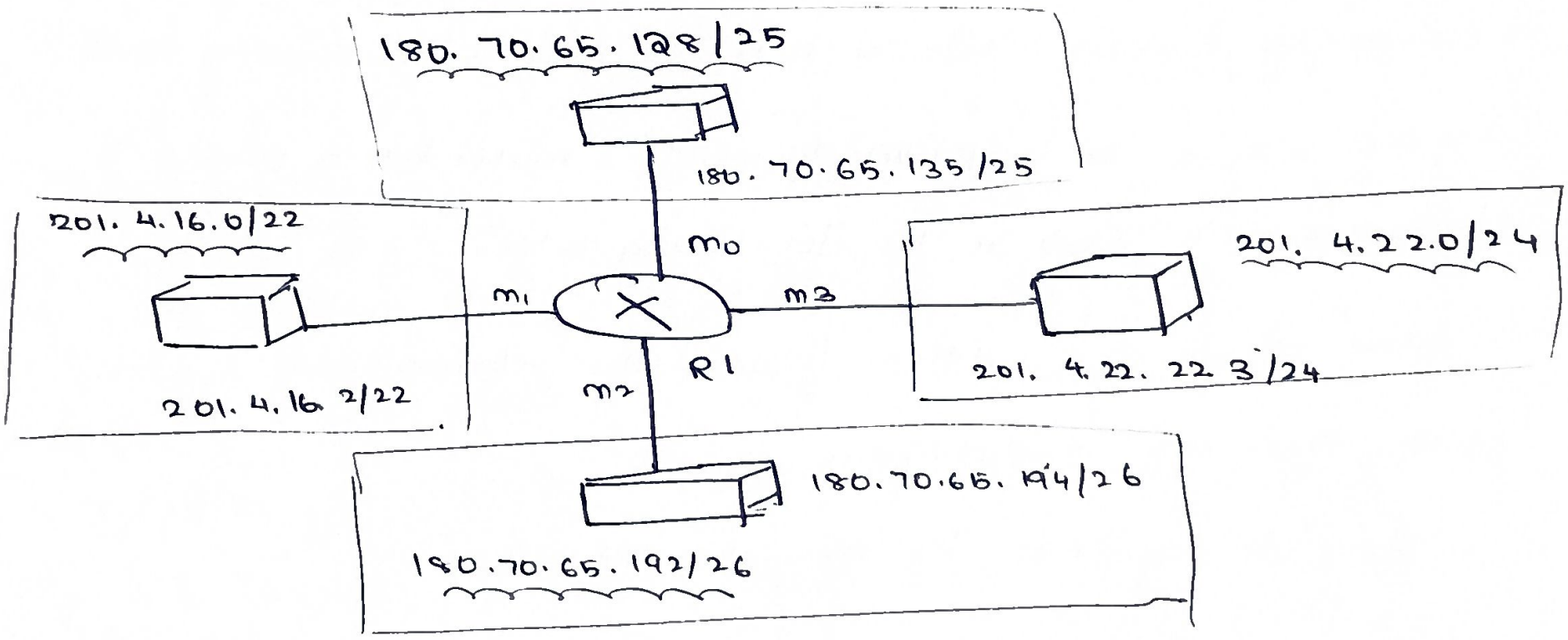
→ Search the table row by row

→ In each row extract the n leftmost bits (n is from the mask)

→ The resulting address is called the network address.

→ If the network address matches, extract the info from the next two columns. (perform AND w/ destination address)

Example A. Make a forwarding table for R1 for the given configuration. Make a table with the prefix bits as well



Solution:

Network Address/ Mask	Next Hop	Interface
180. 70. 65. 192/26	—	m2
180. 70. 65. 128/25	—	m0
201. 4. 22. 0/24	—	m3
201. 4. 16. 0/22	—	m1
Default	180. 70. 65. 200	m2

Network Address/ Mask	Next Hop	Interface
10110100 01000110 01000001 11	—	m2
10110100 01000110 01000001 1	—	m0
11001001 00000100 00011100	—	m3
11001001 000 00100 000100	—	m1
Default	180. 70. 65. 200	m2

B. Show the forwarding process if a packet arrives at R1 with the destination address 180.70.65.140

Ans:

180.70.65.140 in binary is:

10110100 0100 0110 0100 0001 1000 1100

Comparison with the first row

check first 26 bits

dest: 10
from table: 11
} does not match

Comparison with the second row

dest: 11
from table: 11
} matches - route through m0

result = 180.70.65.128

Address Aggregation

→ with classless address, it is very likely that the no. of forwarding table entries is large. ⇒ search time increases

→ As a solution, use address Aggregation

→ Consider R_1 connected to 4 organizations. R_2 is far away from R_1. R_1 has a longer forwarding table because each packet must be correctly routed to the appropriate organization. R_2 can however have a small forwarding table. For R_2, any packet with a destination in a particular range can be

sent out from the same interface.

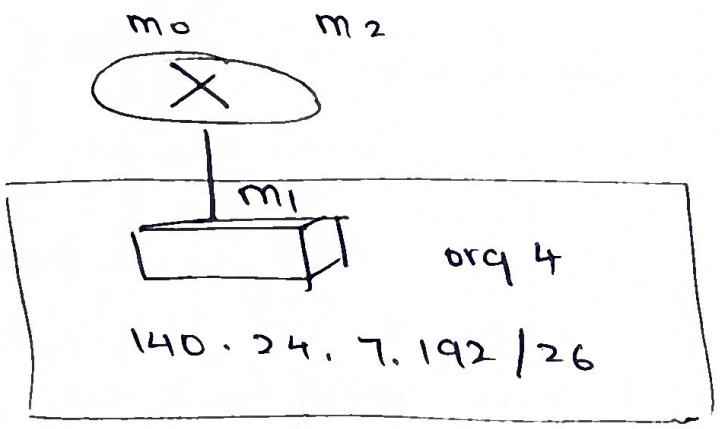
→ This is called address aggregation because the blocks of addresses for four organizations are aggregated into one larger block.

Longest Mask Matching?

→ Address aggregation can be used even if one of the organizations is not geographically close to the others.

→ This is done with longest mask matching. This principle states that the forwarding table is sorted from the longest mask to the shortest mask.

Example Show the routing of a packet for org 4. with dest addr. 140.24.7.200.



Network addr/ mask	Next-hop address	Interface
140.24.7.192/26	—	m1
140.24.7.0/24	R1	m0
0.0.0.0/0	default	m2

140.24.7.200

1000 1100 0001 1000 0000 0111 1100 1000

140 24 7 200

row1: 1000 1100 0001 1000 0000 0111 1100 0000

Route through interface m1

If the masks were not sorted the /24 mask would be applied, and the packet would be ~~fo~~warded to R1 (not in pic see other pic

Hierarchical Routing

→ create a hierarchy of forwarding tables.

→ A local ISP can be assigned a single, but large, block of addresses with a certain prefix length.

→ The local ISP can divide this block into smaller blocks of different sizes, and assign these to individual users & orgs.

→ For eg. if the block assigned to the local ISP starts with a.b.c.d/n, the ISP can create blocks like e.f.g.h/m, where m varies for each customer and m > n.

→ This reduces the size of the forwarding table since the rest of the Internet does not have to be aware of this division. All customers are defined as a.b.c.d/n to the rest of the Internet.

→ Inside the local ISP, the router must recognize the sub blocks and route the packet to the dest. customer.

Geographical Routing

→ Extend hierarchical routing to include geographical routing

→ Assign a block to America, Europe, Asia etc.

→ Routers outside America will have only one entry for packets to America.

Forwarding Table Search Algorithms

→ use longest prefix match

→ divide forwarding table into buckets, one for each prefix.

The router first tries the longest prefix. If the dest. address is found in this bucket, the search is complete. Otherwise move to the next bucket.

→ This type of search takes a long time.

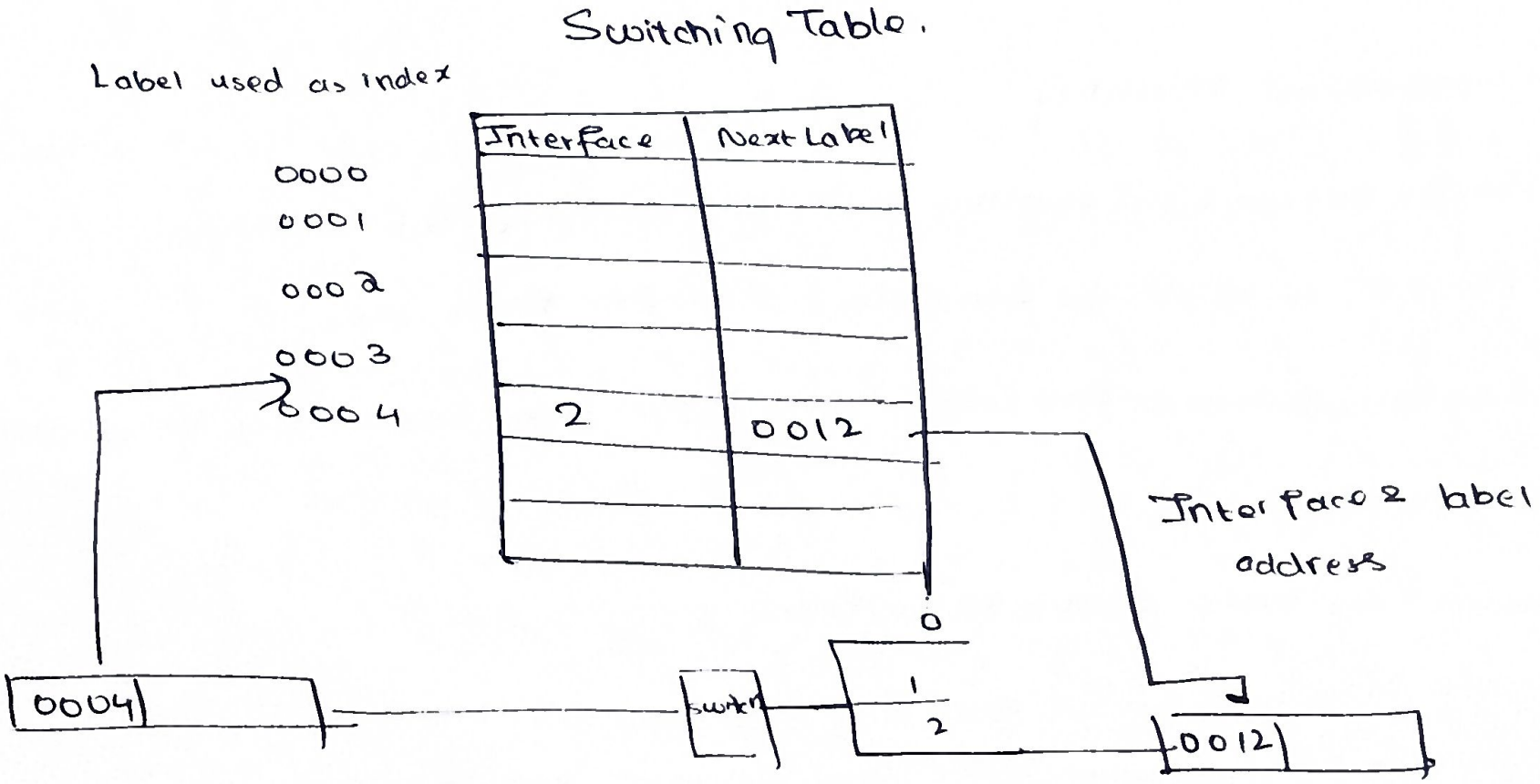
→ Can change data structure used for searching (use a trie/ tree/ binary tree instead of a list)

* B. Forwarding based on Label

→ Used when IP behaves like a connection-oriented protocol in which the routing is replaced by switching.

→ In a connection-oriented network, a switch forwards a packet based on a label attached to the packet.

→ Switching can be done by accessing a table using an index



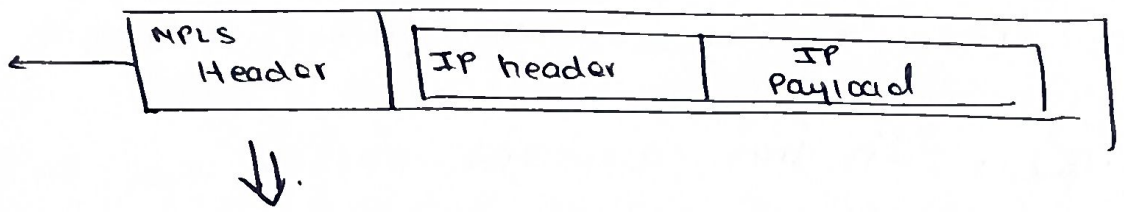
* Multi-Protocol Label Switching

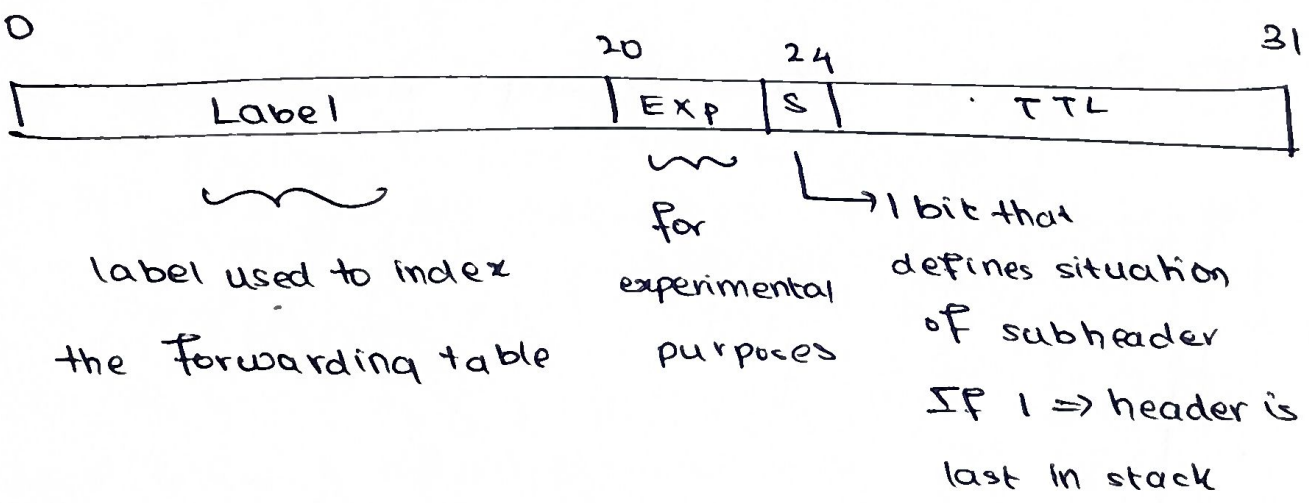
→ In MPLS, some conventional routers can be replaced by MPLS routers, which behave like a router & a switch

→ When behaving like a router → forward packet based on the destination label

When behaving like a switch → forward packet based on the label.

→ The IPv4 packet is encapsulated in an MPLS packet





Example. From the diagram on page 18, can R_1 receive a packet from 140.24.7.194. What will happen to the packet?

140.24.7.194 ⇒ 1000 1100 0001 1000 0000 0111 1100 0010

compare w/ row 1 → 00

row 2 → 01

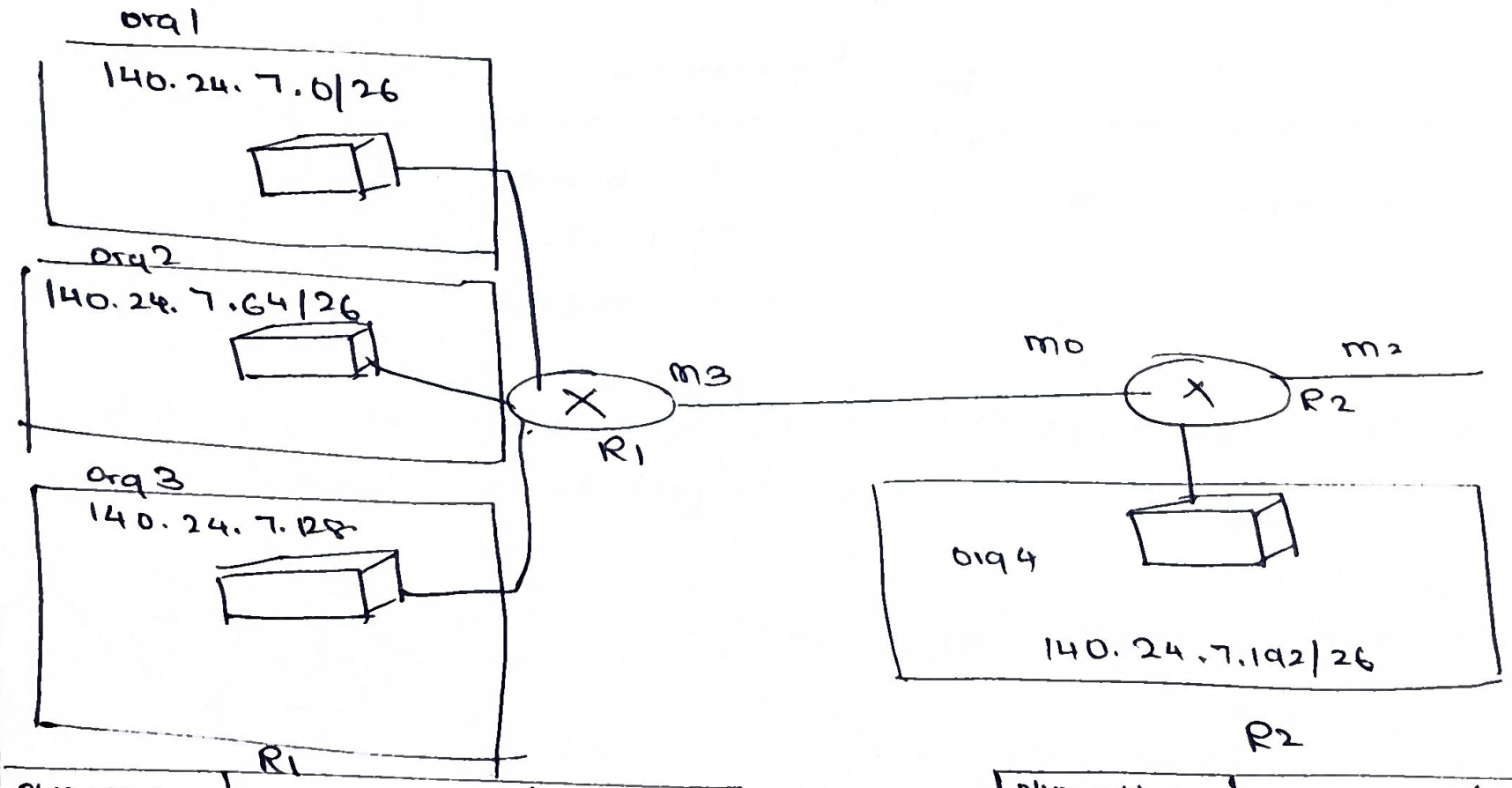
row 3 → 10

doesn't match w/ any address, use the default interface, m3

B. Considering the diagram on page 11 and this routing table how does R_2 send a packet with destination address 140.24.7.42?

Destination Network address / mask	Next hop	interface
140.24.7.192 /26	-	m_1
140.24.7.0 /24	-	m_0
0.0.0.0 / 0	default	m_2

Ans On checking, it can be seen that there is a match with the 140.24.7.0/24 network. R_2 forwards the packet to R_1 via the interface m0. When R_1 receives the packet, it uses its forwarding table (pg.18). In this case, the packet goes to a dest w/ IP 140.24.7.0, it goes to org1 via interface m0.



R1

n/w addr + mask	next hop	interface
140.24.7.0/26	-	m0
140.24.7.64/26	-	m_1
140.24.7.128/26	-	m_2
0.0.0.0/0	default router	m3

R2

n/w addr + mask	next hop	interface
140.24.7.192/26	-	m_1
140.24.7.0/24	R1	m0
0.0.0.0	default	m_2

3.3 * IPv4

→ An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the internet.

→ The IP address is the address of the connection, not the host or router, because if the device is moved to another network, the IP address may be changed.

* IPv4 Address Space

→ Address Space - total number of addresses used by the protocol

→ If the protocol uses b bits to define an address, the address space is 2^b.

→ IPv4 uses 32-bit addresses ⇒ the address space is 2^{32}.

* IPv4 Notation

→ Three common notations are used:

(i) binary - displayed as 32 bits. Insert a space every 8 bits
1 octet = a byte

(ii) decimal - more readable, separate by a (.).

(iii) hexadecimal - equivalent to 4 bits, there are 8 hexadecimal digits, used in network programming

* IPv4 Addressing Hierarchy

→ address divided into 2 parts (totally 32 bits)

(i) prefix - defines the network

(ii) suffix - defines the node

→ The prefix can be fixed or variable length.

fixed - length prefix ⇒ classful addressing

variable - length prefix ⇒ classless addressing

* Classful Addressing

→ Prefix length is fixed.

→ To accommodate both small and large networks, 3 fixed-length prefixes were used

→ The whole address space was divided into 5 classes (A, B, C, D, E)

Class	prefix length	no. of bits defining class	n/w identifier length
A	8	1	7
B	16	2	14
C	24	3	21

→ Class D is not divided into prefix and suffix- used for multicast address.

→ Class E also not divided, used as a reserve, addresses start with 1111.

* Address Depletion

→ The reason why classful addressing became obsolete.

→ The addresses were not distributed properly- no more addresses were available for organizations.

→ For eg: Class A could be assigned to only 128 orgs.

→ Class A & B mostly unused

→ Class C had too few addresses in a block.

→ Class D & E almost never used.

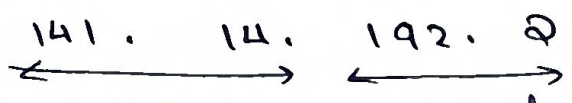
Solution - use subnetting and supernetting?

* Subnetting & Supernetting?

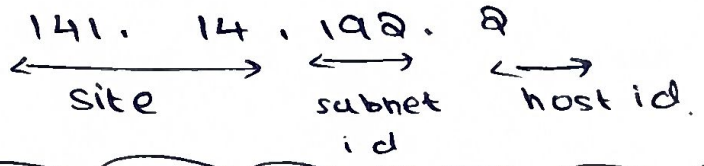
A. Subnetting - the practice of dividing a network into 2 or more smaller networks (divide class A/B to several subnets)

→ increases routing efficiency, enhances network security & reduces the size of the broadcast domain

w/o subnetting:



w/ subnetting



Network Address = IP AND Default Mask

B. Supernetting

→ Combine several blocks (Class C) into a larger block

→ makes routing of packets more difficult

* Advantage of Classful Addressing

→ Given an address, we can easily find the class of the address

→ Since the prefix length is fixed, prefix length can be found.

Example What is the subnetwork address if the destination address is 130.45.34.56 and the subnet mask is 255.255.240.0

Address: 1000 0010 . 0010 1101 . 0010 0010 . 0010 0011 1000

Subnet Mask: 1111 1111 . 1111 1111 . 1111 0000 . 0000 0000 0000

Subnetwork Address: 1000 0010 . 0010 1101 . 0010 0000 . 0000 0000

130 . 45 . 32 . 0

* Classless Addressing

→ The whole address space is divided into variable length blocks.

→ Prefix: block (network)

Suffix: node (device)

→ Blocks can have $2^0, 2^1, \ldots 2^{32}$ addresses. The no. of addresses in a block needs to be a power of 2.

Note - The size of the network is inversely proportional to the length of the prefix.

* Prefix Notation in Classless Addressing

→ Since the prefix length is not inherent in the address, the length must be separately given.

→ In slash notation, the prefix length, n, is added to the address

→ This notation is also called the Classless Interdomain Routing (CIDR)

byte	.	byte	.	byte	.	byte	/	n

* Extracting Information from an Address

Method1

1. no. of address $\Rightarrow N = 2^{32-n}$

2. First address $\Rightarrow$ keep the n leftmost bits, and set the remaining bits to 0.

3. last address $\Rightarrow$ keep the n leftmost bits - and set the remaining bits all to 1s.

Method2 - using the address mask.

(can easily be used by a computer program w/ the NOT, AND, OR bit-wise operations)

1. no. of addresses - $N \Rightarrow$ NOT(mask)+1

2. First address $\Rightarrow$ any address in block AND (mask)

3. last address $\Rightarrow$ any address OR (NOT(mask))

Example The classless address is 167. 199. 170. 82 / 27

Find the no. of addresses, first & last address in both methods

Solution : Method1:

no. of Addresses : $N = 2^{32-n}$

$N = 2^{32-27} = 2^5 = 32$

First address :	10100111	11000111	10101010	01010010
change last 5 bits to 0	10100111	11000111	10101010	01000000
		= 167. 199. 170. 64/27		

last addres: change last 5 bits to 1

(or) add size

167. 199. 170. 64
31

.95

last address = 167. 199. 170. 95

Method 2: Find the mask address

/27 ⇒ 27 1's and 5 0s

= 11111111 11111111 11111111 11100000

= 255. 255. 256. 224

? ← add 1

og is 255

N = NOT(mask) + 1 = 0. 0. 0. 31 + 1 = 32

First = address AND mask = 167. 199. 170. 82

Last = address OR (NOT mask) = 167. 199. 170. 95

* Network address in Classless addressing

→ used to route packet to destination

→ the router needs to know which interface to send the packet from

→ after identifying the network address - consult the forwarding table for finding out the interface. (see diagram on pg. 12)

* Block Allocation

→ done by a global authority called ICANN - International Corporation for Assigned Names and Numbers (ICANN).

Method:

1. no. of requested addresses N - should be a power of 2

2. First address must be divisible by the no. of addresses in the block.

First address = (prefix in decimal) × 2^{32-n} = (prefix in decimal) × N

for eg - if 1000 addresses are requested, a block of 1024 addresses is granted.

* Subnetting in Classless Addressing

→ An ISP that is granted a range of addresses may divide the range into several subranges, and assign each subrange to a subnetwork

Steps in Designing subnets

1. no. of addresses in each subnetwork should be a power of 2.

2. find the prefix length as: $n_{sub} = 32 - \log_2 N_{sub}$

3. starting address should be div. by no. of addresses in the block

Always start by assigning addresses to the larger subnetworks

Example An org. is granted a block of addresses w/ the starting address: 14.24.74.0/24. It needs 3 subblocks. The subblock sizes are: 60, 10 & 120. Design the subblocks.

Solution: Total no. of address = $2^{32-24} = 2^8 = 256$ addresses

Group 1: 120 addresses

assign 128 addresses

subnet mask : $n_1 = 32 - \log_2 N_{sub}$

$= 32 - \log 128$

$= 32 - 7 = 25$

First addr = 14.24.74.0/25

last addr = 14.24.74.127/25

Group 2: 60 addresses

assign 64 address

subnet mask : $n_2 = 32 - \log_2 64$

$= 32 - 6 = 26$

First addr = 14.24.74.128/26

last addr = 14.24.74.191/26

Group 3: 10 addresses

=> assign 16 addresses

$n_3 = 32 - \log_2 16$

$= 28$

First addr = 14.24.74.192/28

last addr = 14.24.74.207/28.

Example An ISP is granted a block of addresses starting 190.100.0.0/16. The ISP needs to distribute these addresses to 3 groups.

a. Group 1 - 64 customers - each needs 256 addresses

b. Group 2 - 128 customers - each needs 128 addresses

c. Group 3 - 128 customers - each needs 64 addresses

Design the subblocks and find out how many addresses are still available after these allocations.

Ans: Group 1: 256 addresses

Subnet mask => $n_1 = 32 - \log_2 256$

$= 24$

1st addr = 190.100.0.0/24 - 190.100.0.255/24

64th addr = 190.100.63.0/24 - 190.100.63.255/24

(63 → division of group; 255 → for each individual cust)

Total = 64 × 256

= 16,384

Group 2: 128 addresses

Bits to define host = $\log_2 128 = 7$

prefix length : 32 - 7 = 25

1st address = 190.100.64.0/25 - 190.100.64.127/25

64th address = 190.100.127.128/25 - 190.100.127.255/25

Total = 128 × 128 = 16,384

Group 3: bits for each host : $\log_2 64 = 6$

Prefix length : $32 - 6 = 26$

1st addr : 190.100.128.0/26 - 190.100.128.63/26

2nd addr = 190.100.128.64/26 - 190.100.128.127/26

128th addr = 190.100.159.192/26 - 190.100.159.255/26

Total = $128 \times 64 = 8192$

Remaining Addresses:

no. of granted addresses = 65,536

no. of allocated address = 40,960

no. of available addresses = 24,576

* Address Aggregation

→ Blocks of addresses are combined to create a larger block, routing can be done based on prefix of the larger block.

* Special Addresses

(1) This-host address

→ 0.0.0.0/32

→ used when the host needs to send an IP datagram but does not know its own address to use as the source address.

(2) limited-broadcast address

→ 255.255.255.255/32

→ when a router/host needs to send a datagram to all devices in a network.

→ packet cannot travel outside network

(3) Loopback Address

→ 127.0.00 /8

→ packet never leaves host

→ used mostly for client-server testing

(4) Private address

→ 4 specific addresses - 10.0.0.0/8
172.16.0.0 /12
192.168.0.0/16
169.254.0.0/16

(5) Multicast Address

→ the block 224.0.0.0 /4

* Problems

(1) Find the number of addresses in a range if the first address is 146.102.29.0 and the last address is 146.102.32.255

Ans: subtract: 146.102.32.255
146.102.29.0
0.0.03.255

convert from base 256 to base 10

$(0 \times 256^3 + 0 \times 256^2 + 3 \times 256 + 255 \times 256^0) + 1$

$= 1024$

(2) Find the class of each of the following addresses:

a. 0000 0001 00001011 00001011 11101111

↳ starts with 0 ⇒ class A

b. 1100 0001 1000 0011 0001 1011 1111 1111

↳ starts with 110 ⇒ class C

0 → class A
10 → B
110 → C
1110 → D
1111 → E

c. 10100111 11011011 1000 1011 01101111

↳ class B

a. 1111 0011 1001 1011 1111 1011 00001111

↳ class E

(3) Find the class of each address.

a. 227.12.14.87 = D

b. 193.14.56.22 = C

c. 14.23.120.8 = A

d. 252.5.15.111 = E

0 - 127 → A

128 → 191 → B

192 → 223 → C

224 → 239 → D

240 - 255 → E

(4) An address in a block is given as 180.8.17.9. Find the no. of address, the first & last address.

Ans no. of address: 2^{32-n}

** 180 lies between 128 & 191 ⇒ class B address

⇒ n = 16

$N = 2^{32-16} = 2^{16} = 65,536$

First address = keep the 16 left most bits, and discard the rest

= 180.8.0.0

last address - keep the 16 left most bits & set the rest to 1

= 180.8.255.255

(5) One of the addresses in a block is 167.199.170.82/27.

Find the no. of addresses, the first and the last address.

no. of address a : $N = 2^{32-27} = 2^5 = 32$

First address:

1010 0111 1100 0111 1010 1010 0101 0010

mask 1111 1111 1111 1111 1111 1111 1110 0000 AND

010 00000

= 167.199.170.64/27

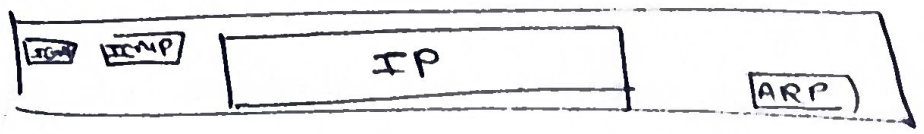
do the same, but set last bits to 1, for last address.

3.4 IP - ICMP

* IP - responsible for packetizing, forwarding and delivery of a packet at the network layer.

Note that the V4 network layer can be considered to be one main protocol and three auxiliary ones

Main - IP

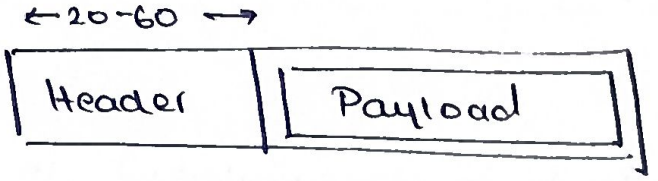


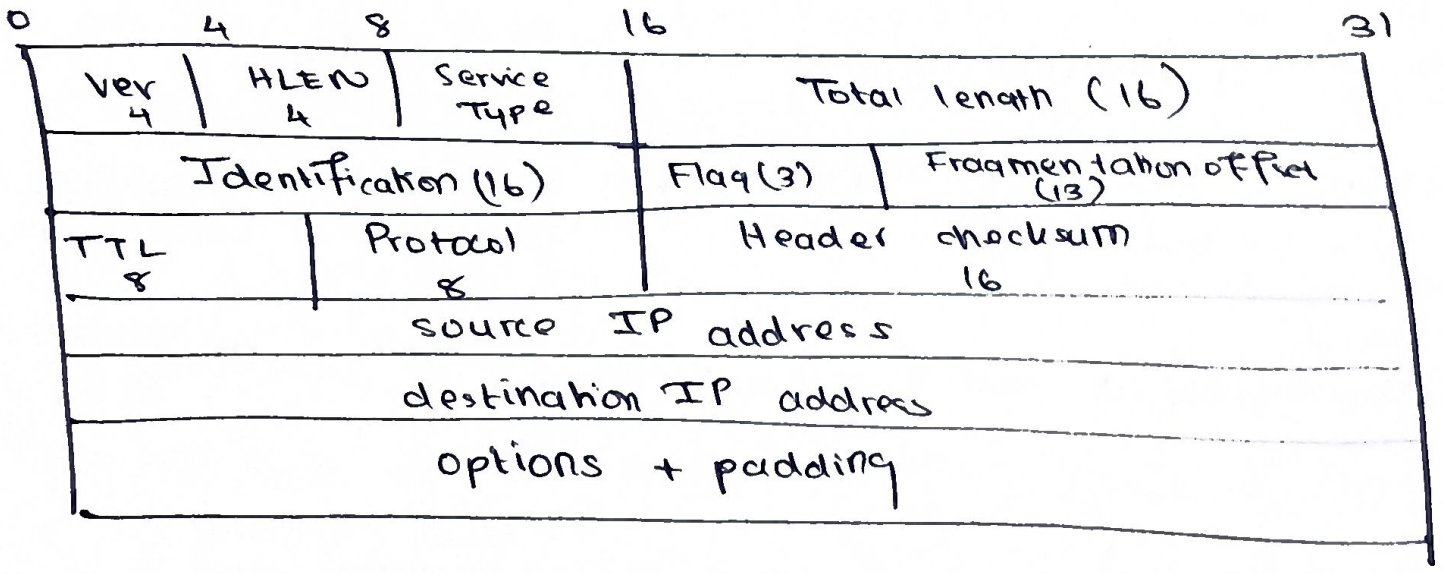
Auxiliary - (i) ICMPv4 - handle errors

(ii) IGMP - multicasting

(iii) ARP - used in address mapping

* IP Datagram Format





(i) version number - defines version of IPv4 protocol

(ii) header length - defines total length of datagram header since IPv4 datagram has a variable header length

** Find header length by multiplying by 4.

(iii) service type - to provide differentiated services

(iv) total length - total length of IP datagram

*** length of data = total length - (HLEN) ×4

(v) identification, flags, fragmentation offset - used when size of the IP datagram is larger than what the underlying network can carry.

(vi) TTL - control maximum no. of hops visited

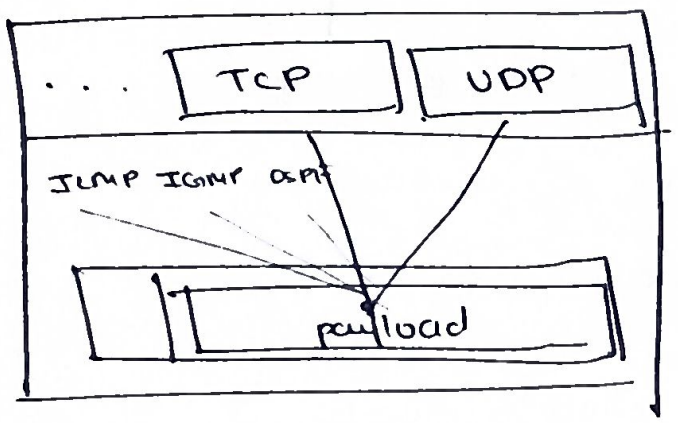
(vii) protocol - datagram can carry packets from other protocols, specified as an 8bit no.

*** Multiplexing and Demultiplexing using the protocol field value.

→ When the payload is encapsulated in a datagram at the source IP, the corresponding protocol no. is inserted in this field.

→ At the destination, this field helps define to which protocol the payload should be delivered.

(i.e multiplexing at source, demultiplexing at destination)



Protocol values

ICMP	01
IGMP	02
TCP	06
UDP	17
OSPF	89

(viii) header checksum - self explanatory

(ix) src, dest addr

(x) options - upto 40 bytes

(xi) payload = content

Example1 . An IPv4 packet arrives with the first 8 bits as $(01000010)_2$. The receiver discards the packet. Why?

Ans. 1st 4 bits = ~~0000~~ 0100 = 4 = correct

next 4 bits = 0010 = 2 ⇒ HLEN = 2 x 4 = 8

min HLEN = 20

⇒ corrupt packet

Example2 In an IPv4 packet, the value of HLEN is $(1000)_2$. How many bytes of options are being carried by this packet?

HLEN = 8 x 4 = 32

⇒ first 20 bytes = base header

12 bytes = options

Example3 In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is $(0028)_{16}$. How many bytes of data is being carried?

Ans : $(0028)_{16}$ = 40

header length = 5 x 4 = 20

data = 40 - 20

= 20 bytes

Example4 : An IPv4 packet has arrived with the first few hexadecimal digits as follows:

$$(45\,00\,00\,28\;\;0001\;\;0000\;\;0102\ldots\ldots)_{16}$$

(9th byte, 10th byte)

How many hops can this packet travel before being dropped?

What upper-layer protocol does the data belong to?

Ans: TTL ⇒ skip 8 bytes (9th byte)

$= (01)_{16}$

⇒ can travel only one hop

upper layer protocol $(02)_{16}$ = IGMP

*Checksum Calculation in IPv4

Consider the following

4	5	0	28
49.153		0	0
4	17	0	
10.12.14.5			
12.6.7.9			

4, 5, 0 → 4 5 0 0

28 → 0 0 1 C

1

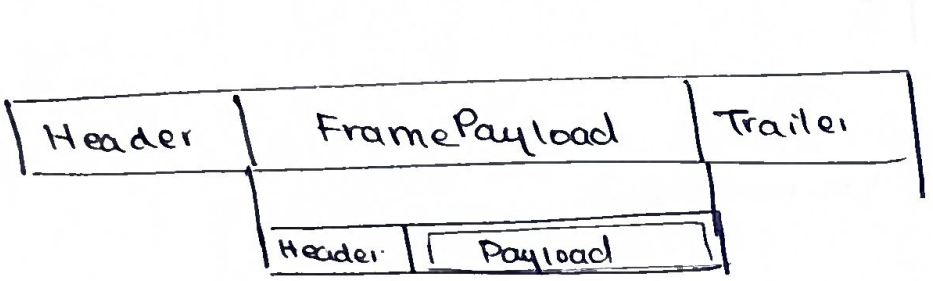
* Fragmentation

→ A datagram can travel through different networks. (Encapsulation and decapsulation happens at each router).

→ The format and size of the frame received depend on the protocol used.

Maximum Transfer Unit (MTU)

→ Each link layer protocol has its own frame format. Each has a maximum size of the payload that can be encapsulated. This is defined by the Maximum Transfer Unit (MTU)



Protocol	MTU
Hypermannel	65,535
Token ring (16Mbps)	17,914
Token ring (4Mbps)	4,464
FDDI	4352
Ethernet	1500
X 25	576
PPP	296

Fragmentation - For some physical networks, we must divide the datagram to make it possible for it to be under the MTU. This is called fragmentation. When a datagram is fragmented, each fragment has its own header.

→ A datagram can be fragmented by the source host or any router in the path. The reassembly of the datagram, however, is done only by the destination host.

Changing Fields during Fragmentation

(i) identification field - the combination of the id field & IP address must uniquely define the datagram as it leaves the source host

(ii) Flags field - 3 bit field

leftmost bit = reserved

second bit (D bit) = do not fragment (1 => don't fragment)

third bit (M bit) = more fragment bit, if it is 1,

it means the datagram is not the last fragment.

(iii) fragmentation offset - 13 bits

shows the offset of the data in the original datagram

measured in units of 8 bytes.

offset = starting byte / 8

* Creation and Reassembly of Fragments

a. The first fragment has an offset value of 0.

b. Divide the length of the 1st fragment by. The second fragment has an offset equal to that result.

c. Divide the total length of the first and second fragment by 8. The third fragment has an offset value equal to that result.

d. Continue the process. The last fragment has its M bit set to 0.

Example : A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment or a middle fragment? Do we know if the packet was fragmented?

Ans: M = 0 ⇒ no more fragments

cannot say if packet was fragmented

Example : A packet arrives with an M bit value of 1. Is this the first, middle or last fragment? Do we know if the packet was fragmented?

M = 1 ⇒ More fragments present

cannot tell if first, middle or last

Example : A packet has arrived with an M bit of 1 and a fragmentation offset of 0. Is this the first, middle or last fragment?

M = 1 and offset = 0 ⇒ first fragment

Example: A packet has arrived in which the offset value is 100? What is the number of the first byte? Do we know the no. of the last byte?

no. of first byte = 8 × 100 = 800

cannot know last byte without data length

Example : A packet has arrived in which the offset value is 100, the HLEN is 5, and the total length is 100. What are the nos. of the first and last byte?

offset = 100 ⇒ no. of first byte = 800

data length = total - (HLEN × 4) = 100 - (4 × 5) = 80

last byte no = 879

* Options

→ can be a maximum of 40 bytes

→ options not necessary can be used for testing and debugging.

→ Two categories of options

(i) single - byte options

(ii) multiple - byte options

A. Single - byte options

1. no - operation - used as a filler between options
2. end of option - used to pad the end of the options field

B. Multiple - byte options

1. record route - record internet routers that handle the datagram
2. strict source route - used to predetermine route for datagram
   must visit each router specified
3. loose source route - similar to (2), but the datagram can visit other routers as well
4. timestamp - to record the time of datagram processing by a router.

* Security of IPv4 Datagrams

1. Packet sniffing
2. packet modification
3. IP spoofing
4. IPsec - can be used to protect IP packets
   IP sec is connection oriented.

IPsec provides the following services:

(i) defining algorithms & keys

(ii) packet encryption

(iii) data integrity

(iv) origin authentication

* ICMPv4

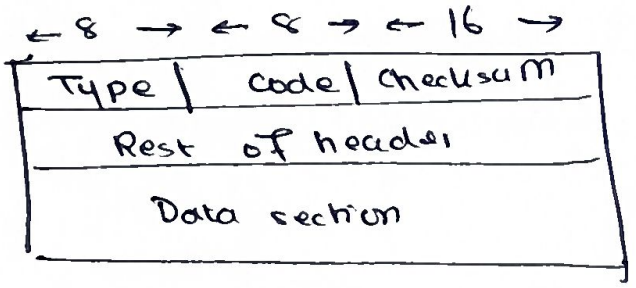
→ The Internet Control Message Protocol 4 (ICMPv4) has been designed to compensate for the foll. deficiencies of IP:

(i) IPv4 has no error-reporting or error-correcting mechanism

(ii) also lacks a mechanism for host and management queries

* ICMP Messages

(i) Error-reporting messages - reports problems that a router a host may encounter when it processes an IP packet.

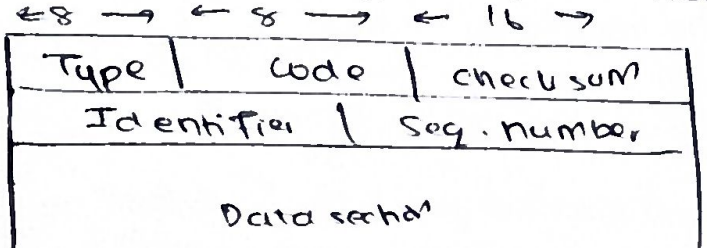


(ii) Query messages - helps host or n/w manager get specific information from a router or another host.

eg. nodes can discover hosts

host - learn about routers on network

routers can help a node redirect its messages



* Debugging Tools

(i) ping - to find if host is alive & responding

- echo requests and echo reply messages are sent, and sequence is incremented by 1.
- sub Arrival time - departure time to get RTT

(ii) traceroute - trace path of packet from source to destination

(iii) ICMP checksum - add & complement

* 3-5 Unicast Routing

* Routing

→ Routing tables can be static or dynamic

Static → manual entries

Dynamic → a table that is updated automatically, when there is a change somewhere in the Internet

* Routing Protocol → a combination of rules and procedures that lets routers in the Internet inform each other of changes.

* Unicast Routing

→ A packet is routed hop by hop from the source to destination with the help of forwarding tables.

→ The source host needs no forwarding table because its packet to the default router in its local network.

→ The destination host needs no forwarding table because it receives the packet from its default router in its local networks

→ Only the routers that glue together networks need forwarding tables.

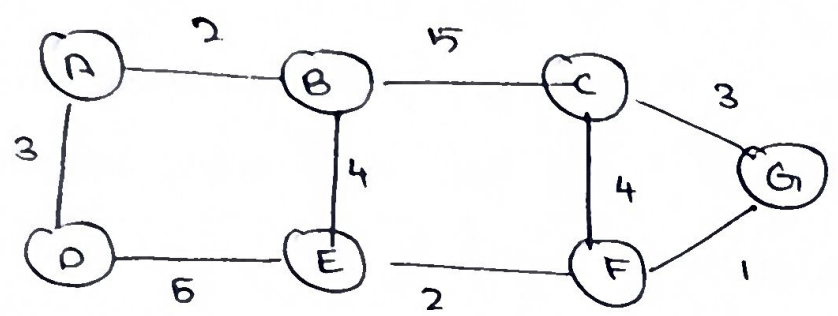
* Least - Cost Routing?

→ Represent the internet as a weighted graph.

→ Find the least cost between source & destination router.

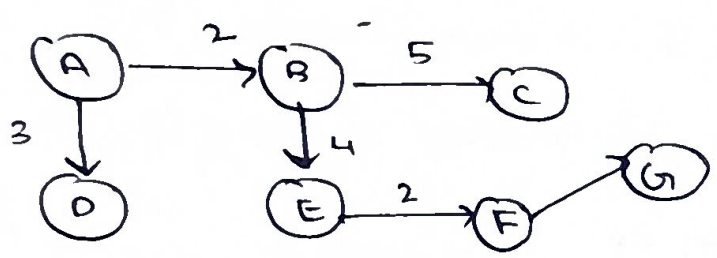
→ From the weighted graph, construct a least-cost tree.

Example:



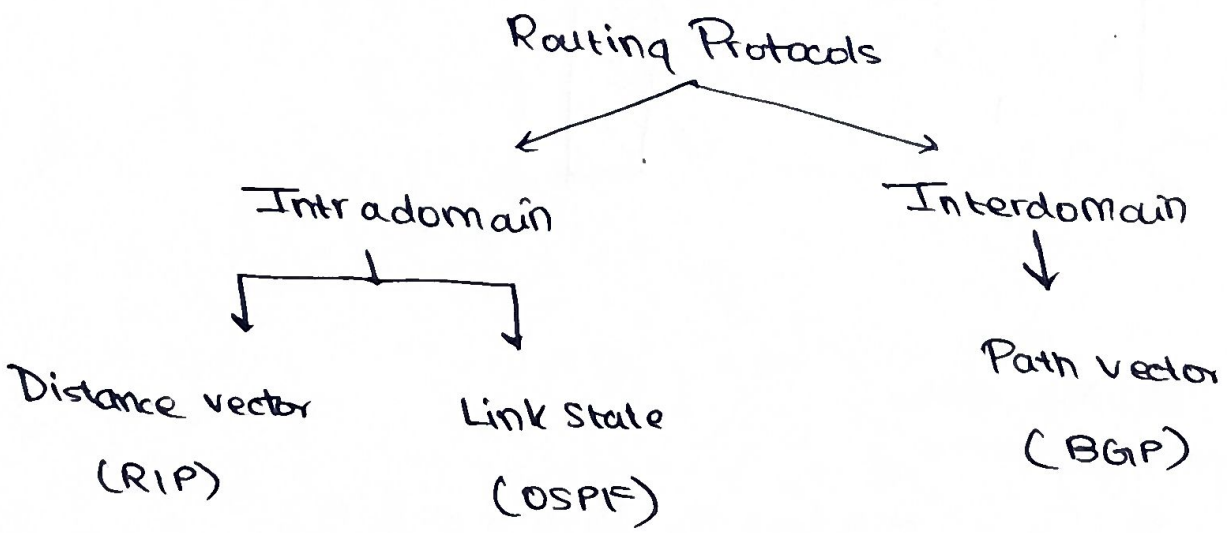
Least cost tree for

Source Node A



till all do for each node

* Routing Algorithms



(1) Distance Vector Routing?

→ Each node creates its own least cost tree, with the info. about its immediate neighbors.

→ The incomplete trees are exchanged between immediate neighbors more & more complete to represent the whole internet.

** Each node shares its routing table with its immediate neighbors periodically & when there is a change.

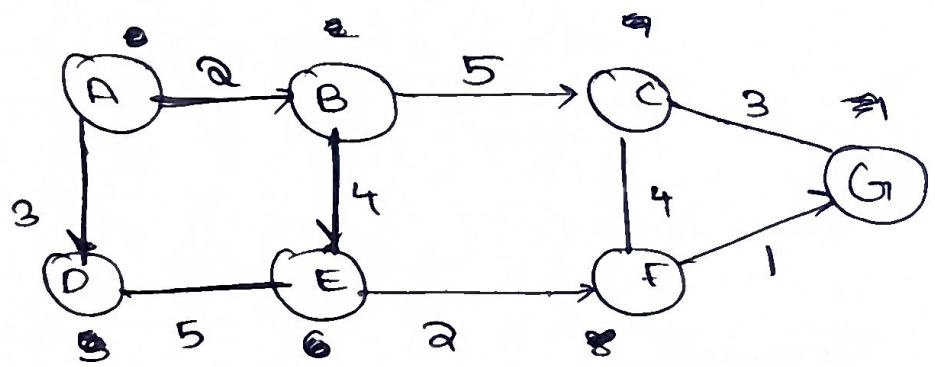
Calculation of least Cost using the Bellman-Ford Algorithm

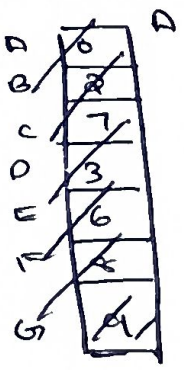
↓

helps find least cost between source (x) and destination (y), through some intermediary node, when the costs between the source and intermediary nodes, and the least costs between the intermediary and destination are given

$$D_{xy} = \min \{ D_{xy}, (c_{xz} + D_{zy}) \}$$

Example: Consider the following tree, and ~~the distance vector~~ for node A. Update the distance vector when B receives a copy of A's vector, and when B receives a copy of E's vector.





Answer

A. B receives a copy of A's vector.

neighbor distance

	New B	Old B	A		min (old B, cost + old A)
A	2	A 2	A	0	min (2, 2+0)
B	0	B 0	B	2	min (0, 2+2)
C	5	C 5	C	3	min (5, 2+3)
D	∞	D ∞	D	∞	min (∞, ∞)
E	4	E 4	E	∞	min (4, ∞+·)
F	∞	F ∞	F	∞	min (∞, ∞)
G	∞	G ∞	G	∞	min (∞ ∞)

B. B receives a copy of vector E

B to E cost = 4

	New B	Old B	E	min (old B, cost+old E)
A	2	2	∞	min (2, 4 + ∞)
B	0	0	4	min (0, 4+4)
C	5	5	∞	min (5, 4+∞)
D	5	~~∞~~ 5	5	min (5, 5+4)
E	4	4	0	min (4, 4+0)
F	6	∞	2	min (∞, 4+2)
G	∞	∞	∞	min (∞, ∞+4)

(~~Find shortest distance to each node~~)

(From previous iteration)

(fill w/ distance of neighbors)

Algorithm for distance-vector routing

```
Distance_vector_routing() {
    D[myself] ← 0
    for y = 1 to N {
        if y is a neighbor
            D[y] = c[myself][y]
        else
            D[y] = ∞
    }
    send vector {D[1], D[2], ... D[n]} to all neighbors
repeat forever
}
wait for a vector Dw from a neigbor / any change in the link
for y = 1 to N {
    D[y] = min(D[y], (c[myself][w] + Dw[y]))
}
if any change in vector
    send vector {D[1], D[2], ... y to all neighbors
```

When should the the new vector be shared?

(i) periodic update - every 30s

(ii) Triggered update

A. A node receives a table from a neighbor resulting in change in its own table after updating.

B. A node detects some failure in the neighboring links which results in a distance change to ∞.

* Issues with Distance-Vector Routing

A. Count-to-Infinity (draw the A, B, 16 diagram for 2 node loop)

→ Each router advertises its routing table to its neighboring routers, and they update their routing table.

→ If there is a link failure, one router (Router A) might not be aware of this change right away. Since Router A is unaware of the failure, it continues to advertises routes to the failed network, with an increasingly high distance (counting up)

→ Other routers, unaware of the problem continue to use these increasingly large distance values, believing each has the shortest path, though the path is no longer valid.

→ Counting goes on infinitely, up to ∞.

B. Two-Node loop (a specific case of count to ∞)

→ Consider 2 nodes A & B, that know how to reach node X.

→ Suddenly the link between A and X fails. Node A changes its table.

→ If B sends its table to A, before receiving A's update, A assumes that B has found a way to reach X, and A updates its table

→ Seeing this update, B also updates

→ The cost of reaching X increases till it reaches ∞. Packets bounce between A & B.

* Solutions to the Count to Infinity Problem

A. Split Horizon

→ Avoid advertisement of routes back to the router from which they were learned.

→ i.e when a router learns a route from one of its neighbors, it refrains from advertising that route back to the same neighbor.

→ This approach breaks the loop that can occur when 2 routers, unaware of a network failure, keep advertising their routes to each other, causing the count to increase infinitely.

Drawback : If there is no news ~~raw~~ about a route after a period, the node deletes the route from its table.

B. Poison Reverse

→ When a router detects that a route is no longer valid, it doesn't just mark the route as unreachable in its own routing table, but also advertises to its neighbors with a metric set to infinity.

→ This poisons the route by flooding the info. that the route is no longer usable.

* Three Node Instability - 2-node instability can be avoided using split horizon & poison reverse. No solution if the instability is between 3 nodes.

② Link State Routing

→ Link state defines the characteristics of a link

→ Cost associated with an edge defines the state of the link.

→ Lower cost links are preferred than those with higher cost.

If the cost of a link is ∞ ⇒ does not exist / is broken

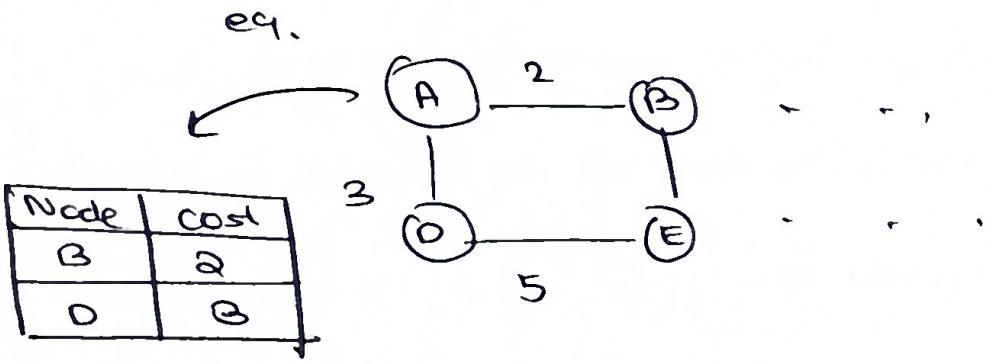
→ In link state routing, each node floods its immediate neighbors info to the whole network.

Make a link state database (LSDB) - (an adjacency matrix)

Building Routing Tables

1. creation of states of the links by each node called the link state packet (LSP).

eg.



Node	Cost
B	2
D	3

2. Dissemination of LSPs to every other router - flooding
3. Form shortest path tree for each node.
4. Calculation of a routing table based on shortest path tree.

Flooding of LSPs

1. Send a copy of the LSP out of each interface.
2. A node that receives an LSP compares it with the copy it may already have. If the new LSP is older than the one it has, discard.

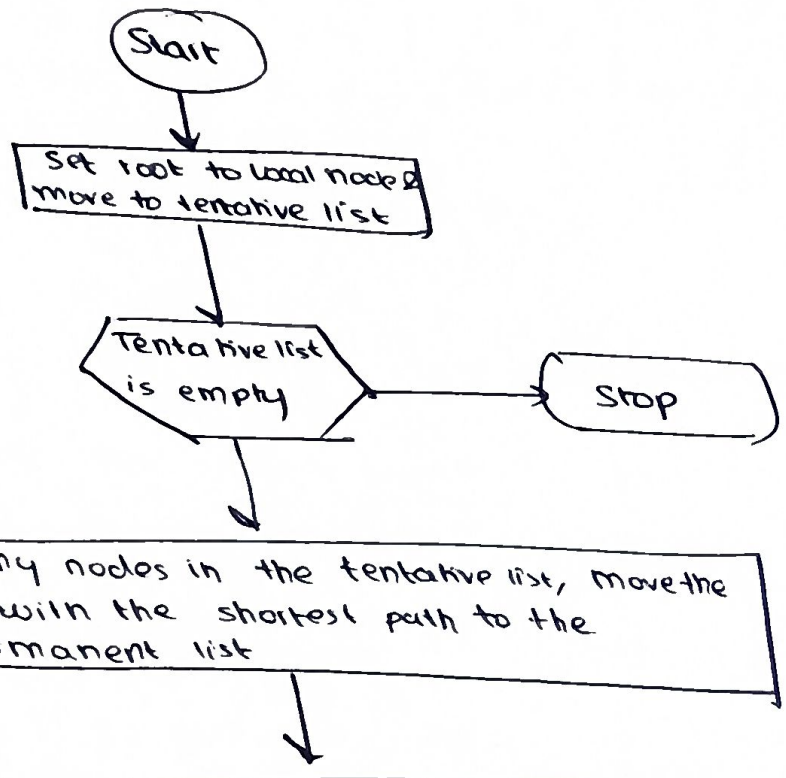
Otherwise:

(i) discard the old LSP & keep the new one

(ii) send a copy of it out of each interface, except from the one from which the packet arrived.

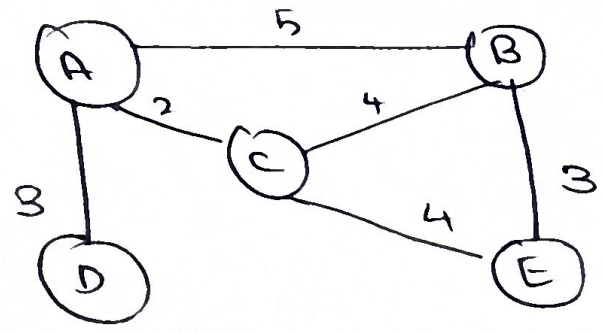
Formation of Least Cost Trees

→ Run the Dijkstra's algorithm.



P9.605 - code version of algo

Example: Construct the routing table using link state routing for the following topology.



Answer

Permanent list	Tentative list
Empty	A(0)
A(0)	B(5), C(2), D(3)
A(0), C(2)	B(5), D(3), E(6)
A(0), C(2), D(3),	E(6)
B(5)	
A(0), C(2), D(3), B(5) E(6)	

Node	Cost	Router
A	0	-
B	5	-
C	2	-
D	3	-
E	6	C

When to share routing tables

→ periodic update - 60 min to 2hrs

→ change in topology.

(3) Path Vector Routing

→ Both link state and distance vector routing are based on the least cost goal - inter domain

→ This does not allow a sender to apply specific policies to the route a packet may take.

→ To accommodate these demands, path vector routing is used.

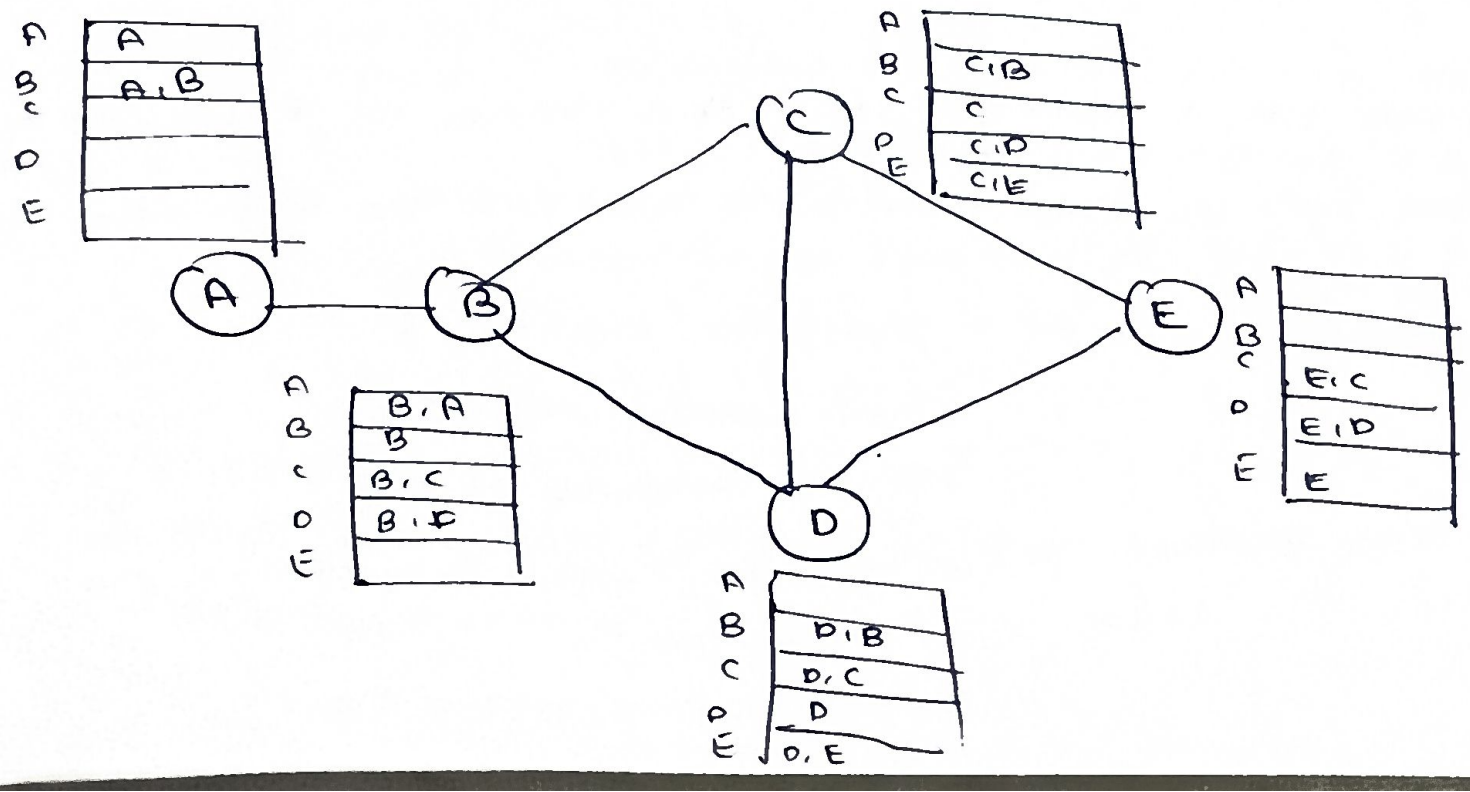
→ The best route is determined by the source using the policy it imposes on the route.

→ The path from a source to all destinations is determined by the best spanning tree.

Creation of Spanning Trees

→ The spanning trees are made gradually & asynchronously by each node.

→ When a node is booted, it creates a path vector based on the info. it can obtain about its immediate neighbor.



Updating Path vectors

$$Path(x,y) = best\{Path(x,y), [x + Path(v,y)]\}$$

Follow the same procedure as in Bellman-Ford for the algo, use the same - change 'min' for distance to 'best'

* Unicast Routing Protocols

3 commonly used protocols:

(i) Routing Information Protocol (RIP) - distance vector algo

(ii) Open Shortest Path First (OSPF) - link-state algo

(iii) Border Gateway Protocol (BGP) - path vector algorithm

* Hierarchical Routing

→ A single protocol is not possible in the internet, there would be scalability and administrative issues.

→ Hierarchical routing means considering each ISP as an autonomous system (AS)

→ Each AS can run a routing protocol that meets its needs, but the global internet runs a global protocol to glue all the AS together

→ Routing protocol in each AS = Intra AS routing protocol
Intra domain "
Interior Gateway Protocol (IGP)

→ Global routing protocol is called
- inter AS routing protocol
Inter domain "
Exterior gateway protocol (EGP)

* Autonomous System

→ Each ISP is an autonomous system when it comes to managing networks and routers under its control.

→ Each AS is given an autonomous number (ASN) by ICANN. (a 16-bit integer)

→ AS are categorized on the way they connect to other ASs.

→ The types of ASs are:

(i) Stub AS - has only one connection to another AS
- data traffic can be either initiated or terminated in a stub AS, data cannot pass through it
- eg. a customer network.

(ii) Multihomed AS - can have more than one connection to other ASs
- does not allow traffic to pass through it.
- eg. a customer who uses services of more than one provider network.

(iii) Transient AS - connected to more than one AS
- allows traffic to pass through
- eg. provider networks & backbone.

* Routing Information Protocol (RIP)

→ Most widely used intradomain protocols based on the distance vector routing algorithm.

Hop Count

→ RIP uses a modified distance-vector algorithm. The changes are:

(i) RIP router advertise the cost of reaching other networks instead of reaching other nodes.

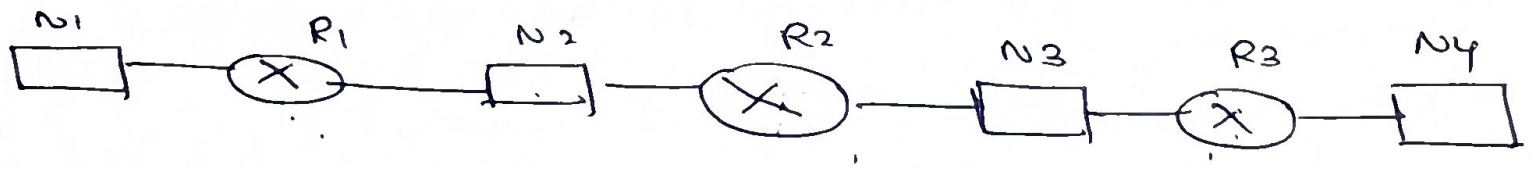
(ii) The cost is defined as the number of hops - meaning the no. of networks a packet needs to travel through from source to destination.

→ Max hop count = 15

Forwarding Tables

→ A 3-column table - w/ destination address, address of the next router, cost in hops.

Consider the following scenario:



Forwarding table for R_1

Destination network	next router	cost in hops
N_1	—	1
N_2	—	1
N_3	R_2	2
N_4	R_2	3

Forwarding Table for R_2

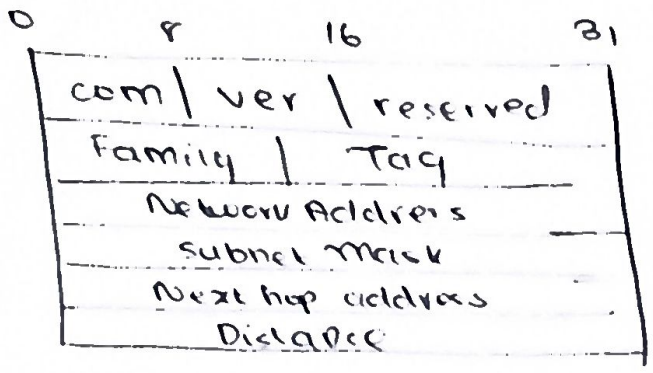
destination network	next router	cost in hops
N_1	R_1	2
N_2	—	1
N_3	—	1
N_4	R_3	2

Forwarding Table for R_3

destination network	next router	cost in hops
N_1	R_2	3
N_2	R_2	2
N_3	—	1
N_4	—	1

count as router - n/w

RIP Message Format



RIP Algorithm

→ Instead of sending only distance vectors, send the whole contents of its forwarding table

Receiver

→ Adds one hop to each cost, and changes next router field to address of sender.

→ Select the old routes as the new ones except in the following cases.

1. received route does not exist in the old fowarding table
2. cost of received route is lower than the cost of the old one
3. cost of received route is higher than old one - but the value of the next router is the same in both cases

RIP Timers

(i) periodic timer - controls the advertising of regular update messages

(ii) expiration timer - governs validity of route - set to 180s per route

(iii) garbage collection timer - purge a route from routing table (120s)

* Open Shortest Path First (OSPF)

→ based on the link-state routing protocol

Metric

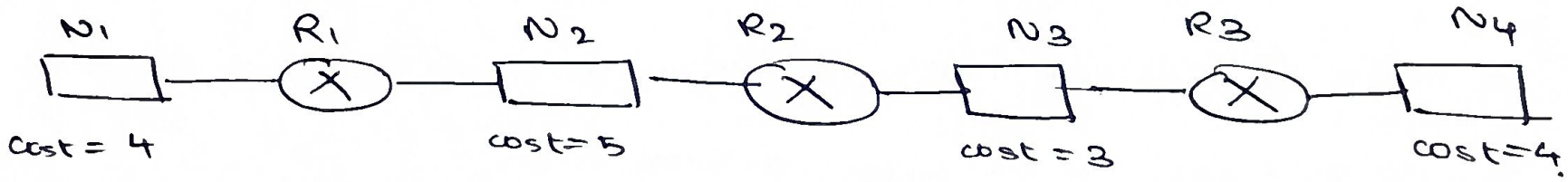
→ The cost of reaching a destination from the host is calculated from the source, by assigning a weight based on the throughput round trip time, reliability etc.

→ Diff. types of services can have different weights

Forwarding Tables

→ create a forwarding table after finding the shortest path tree between the src & destination using Dijkstra's algorithm.

Consider the following scenario:



Forwarding table for R_1

Destination Network	Next Router	Cost
N_1	–	4
N_2	–	5
N_3	R_2	8
N_4	R_2	12

Forwarding table for R_2

Destination Network	Next Router	Cost
N_1	R_1	9
N_2	–	5
N_3	-	3
N_4	R_3	7

Forwarding table for R_2

Destination network	next router	cost
N_1	R_2	12
N_2	R_2	8
N_3	–	3
N_4	–	4

Areas

→ can handle small or large autonomous systems

→ Routers have to flood AS with their LSPs, problematic for large AS - divide into smaller sections called areas.

Link State Advertisements

→ There are 5 types of link state advertisements.

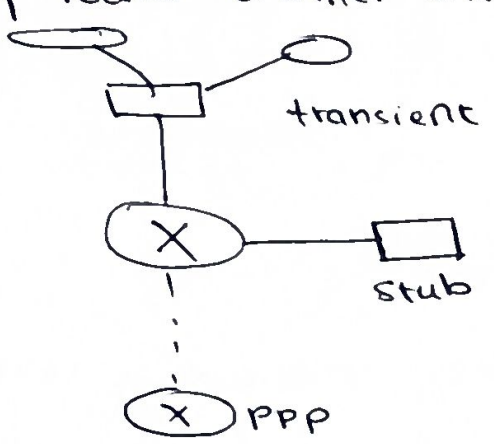
(1) Router Link - advertise existence of node.

- can also connect the advertising router to other entities like:

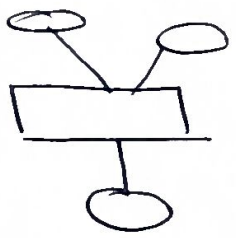
(i) transient link -

(ii) stub link

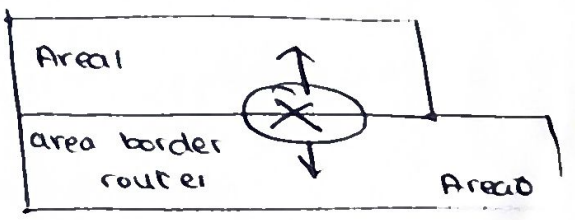
(iii) PtoP link



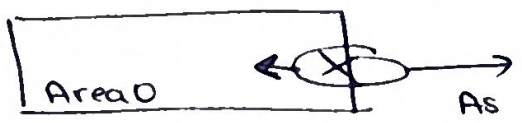
(2) Network Link - a designated router that advertises network as a node.



(3) Summary link to network - advertise summary of links collected by backbone - done by an area border router



(4) Summary link to AS - advertise from AS to backbone



(5) External link AS announce the existence of a single network outside AS

OSPF Implementation

→ has 5 different types of messages

Type 1 - Hello by router

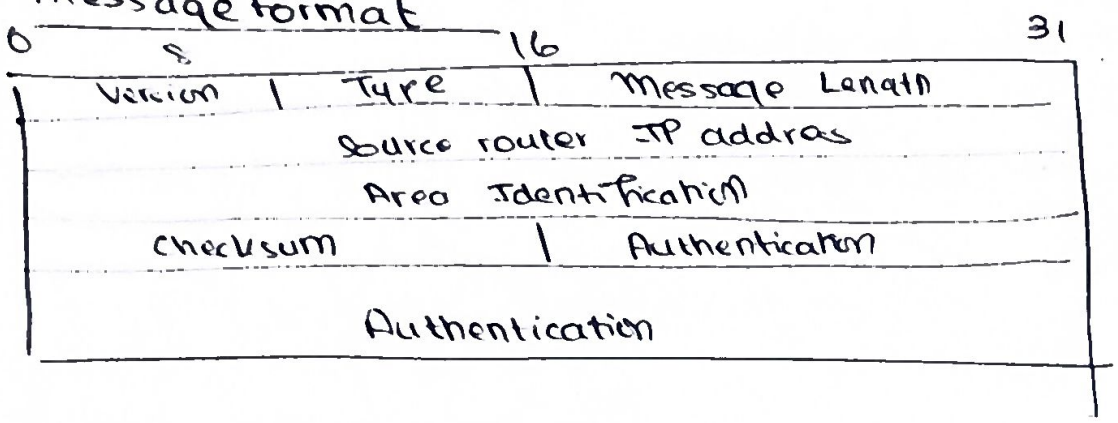
Type 2 - DB description as response

Type 3 - LS requst

Type 4 - LS update

Type 5 - LS ack.

Message Format



* Border Gateway Protocol

↳ the only interdomain routing protocol used in the Internet today.

→ based on the path vector algorithm

→ To enable each router to route a packet to any network on the internet:

(i) install an external BGP (eBGP) on each bader router

(ii) install internal BGP (iBGP) on all routers.

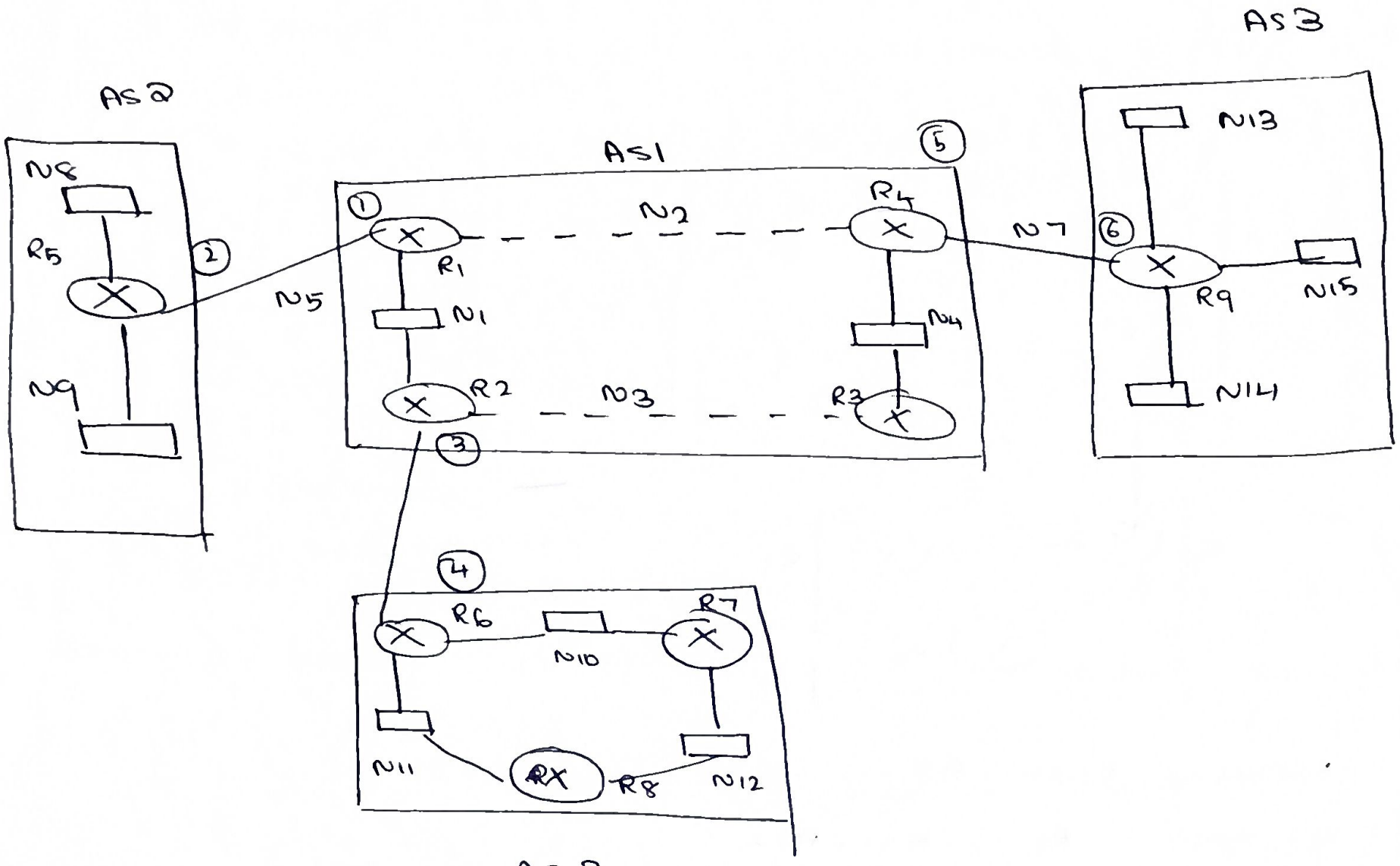
→ This means that border routers run 3 routing protocols

intradomain, eBGP and iBGP.

→ The other routers run 2 protocols

Intradomain & iBGP

Consider the following scenario. Depict the eBGP, iBGP BGP path tables and forwarding tables after injection from BGP.



AS 3

Solution : Step 1: eBGP operations

	Network	Next	AS
①	N1, N2, N3, N4	R1	AS1
②	N8, N9	R5	AS2.

	Network	Next	AS
⑤	N1, N2, N3, N4	R4	AS1
⑥	N13, N14, N15	R9	AS6

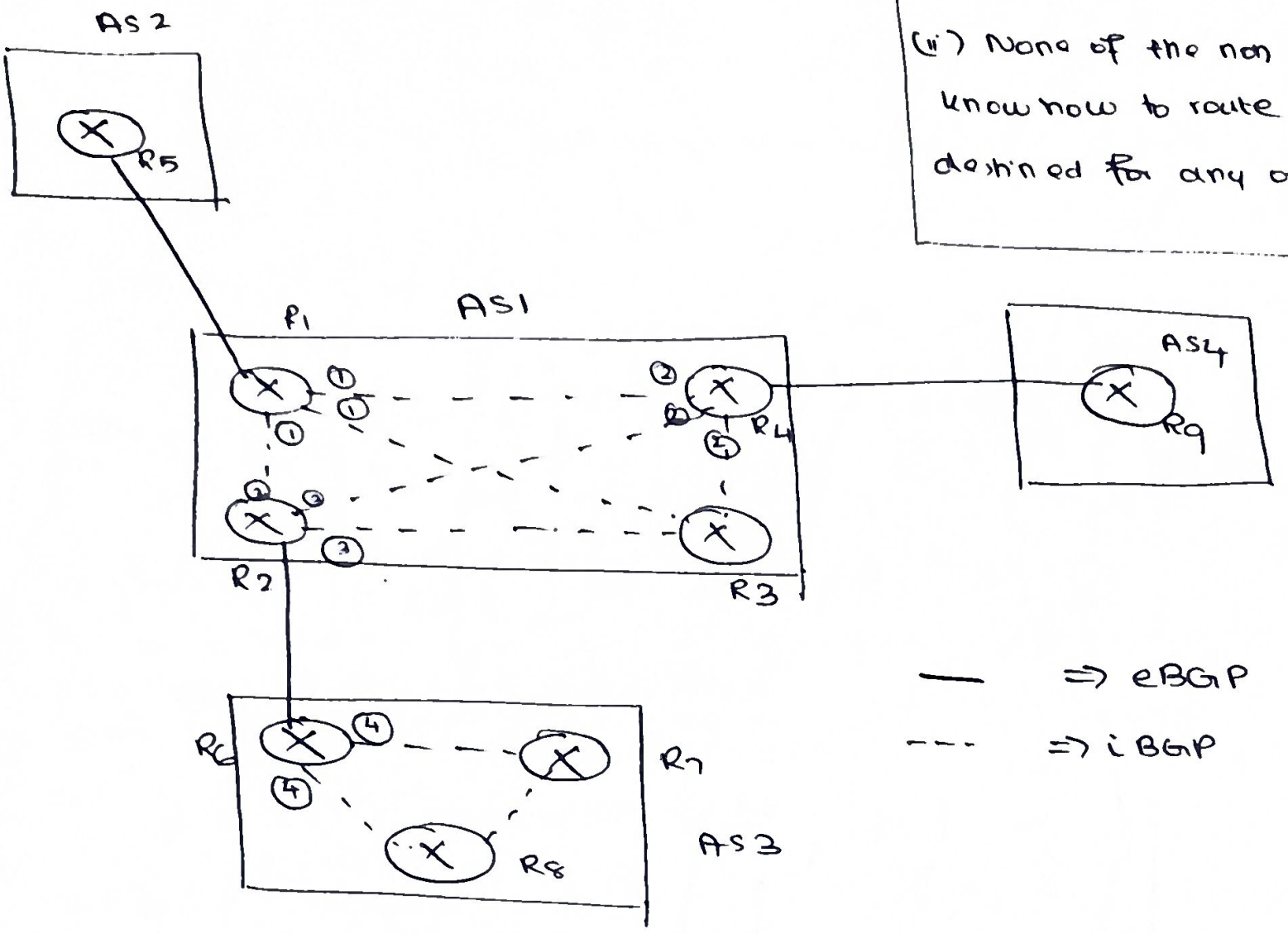
	Network	Next	AS
③	N1, N2, N3, N4	R2	AS1
④	N10, N11, N12	R6	AS 3

Step2: iBGP operations

Why is eBGP not enough?

(i) Routers don't know how to route a packet through non-neighbor As

(ii) None of the non border routers know how to route a packet destined for any other AS



⟶ ⇒ eBGP

⋯ ⇒ iBGP

	Network	Next	AS
①	N8, N9	R1	AS1, AS2
②	N13, N14, N15	R4	AS1, AS4
③	N10, N11, N12	R2	AS1, AS3
④	N1, N2, N3 N4	R6	AS3, AS1

Step3 : Finalized BGP Path tables

P9.626 for remaining tables

1. For R1

Network	Next	Path
N8, N9	R5	AS1, AS2
N10, N11, N12	R2	AS1 AS3
N13, N14, N15	R4	AS1 AS4

2. For R2

Network	Next	Path
N8, N9	R1	AS1, AS2
N10, N11, N0	R6	AS1 AS3
N13, N14, N15	R1	AS1 AS4