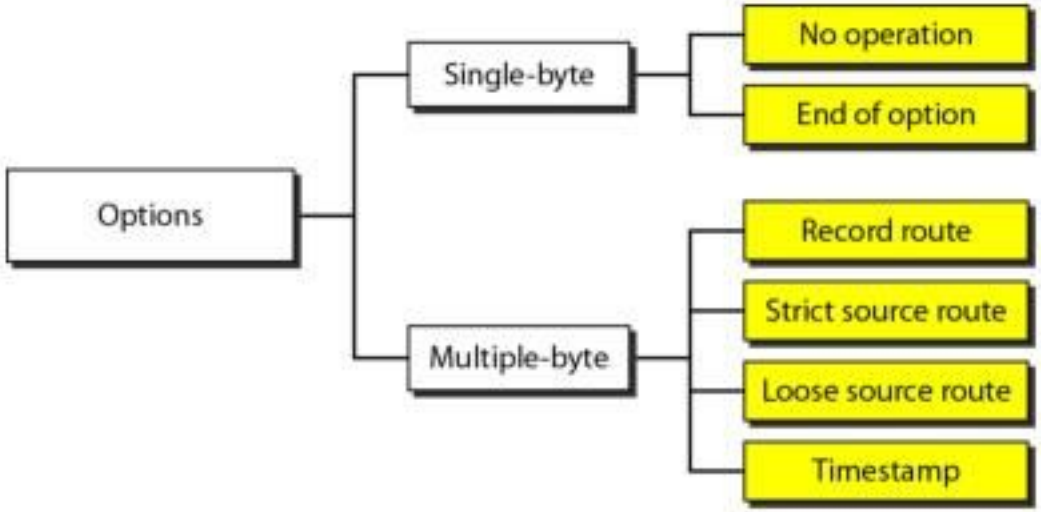
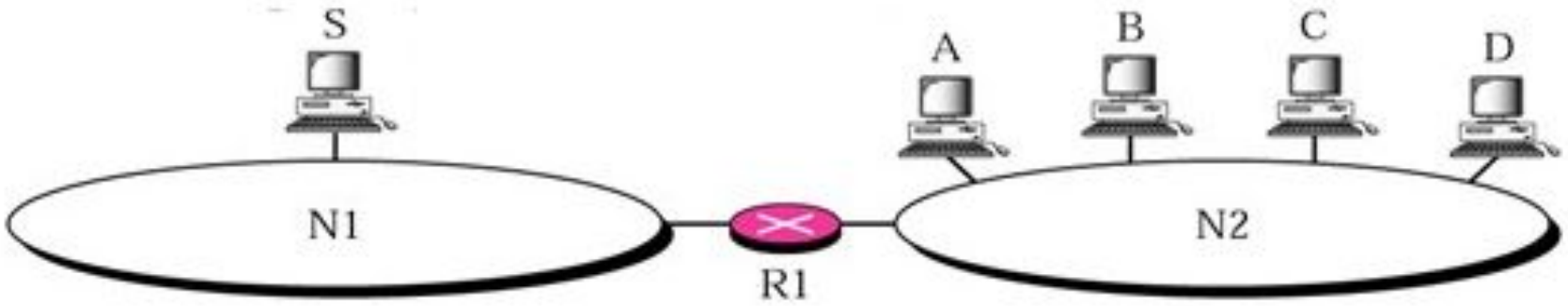
# Options

- IPv4 header is made of two part: a fixed part and a variable part
- Fixed part: 20 bytes long
- Variable part comprises the options that can be a maximum of 40 bytes



Router - It connects different networks together and sends data packets from one network to another.



# How to decide two hosts are on same network?

Case : 1

Host A: 192.168.29.1/24

Host B: 192.168.29.50/24

Case : 2

Host A: 192.168.29.1/24

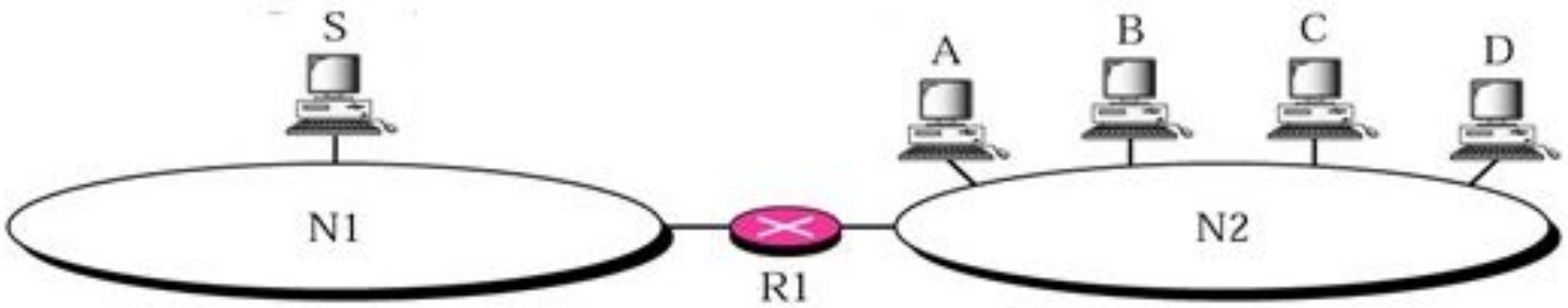
Host B: 200.144.32.2/24

Which case represents that hosts are on the same network?

# Interfaces and ip address of router

How many interfaces and ip address a router has?

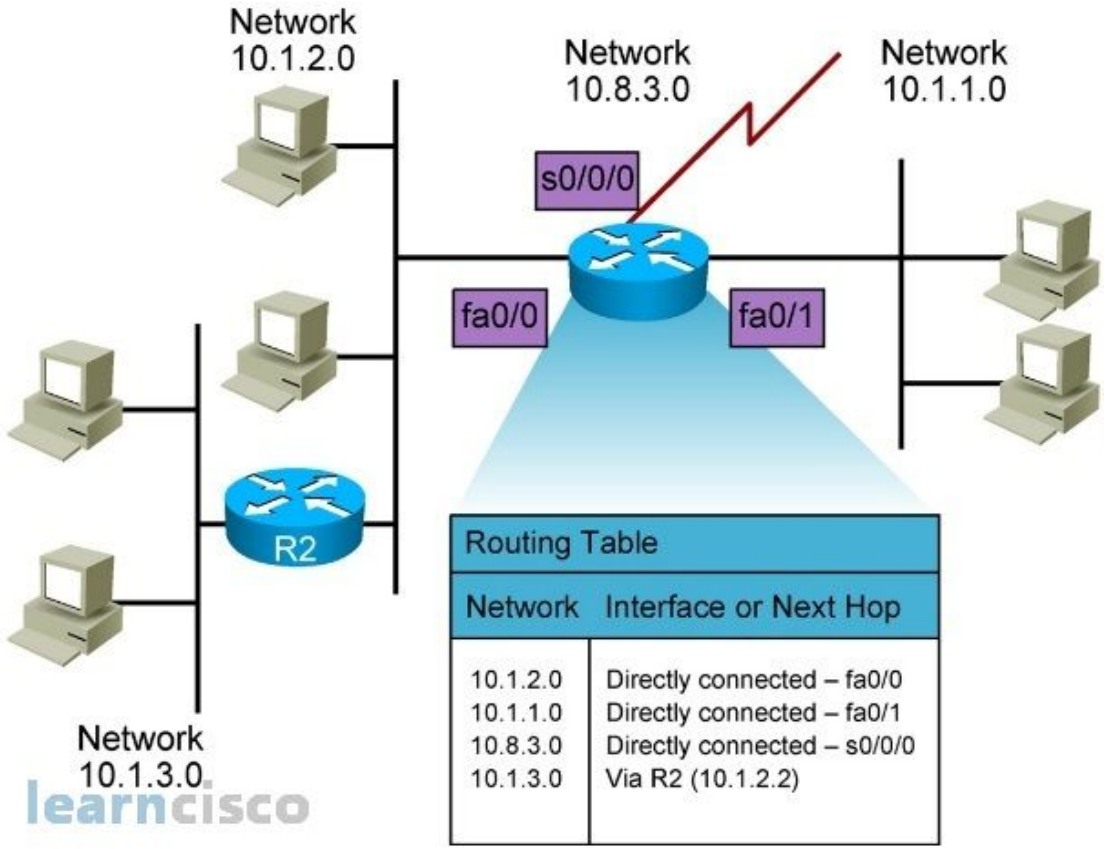
Suggest IP addresses for its all interfaces.




Note 1: To configure router we need to provide ip address to all router interfaces

# How forwarding takes place?

Router maintains routing table:

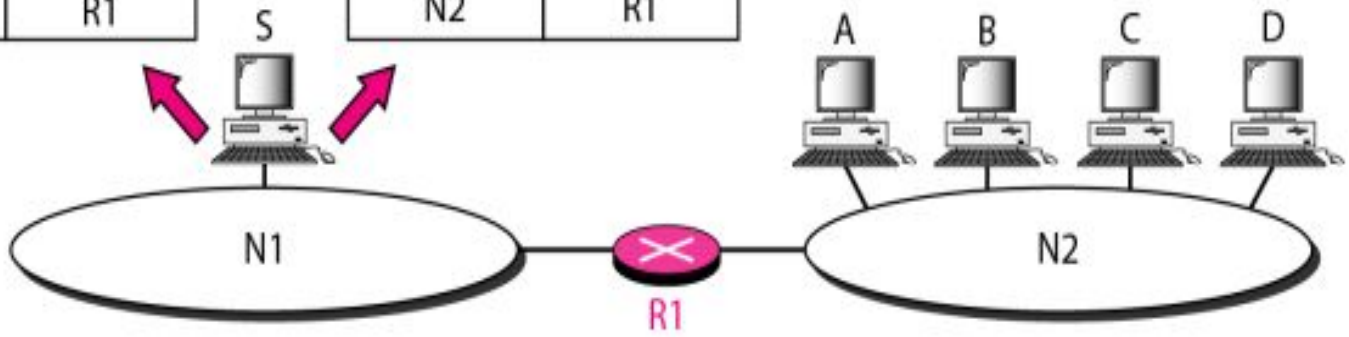


Routing table for host S based on host-specific method

| Destination | Next hop |
|---|---|
| A | R1 |
| B | R1 |
| C | R1 |
| D | R1 |

Routing table for host S based on network-specific method

| Destination | Next hop |
|---|---|
| N2 | R1 |



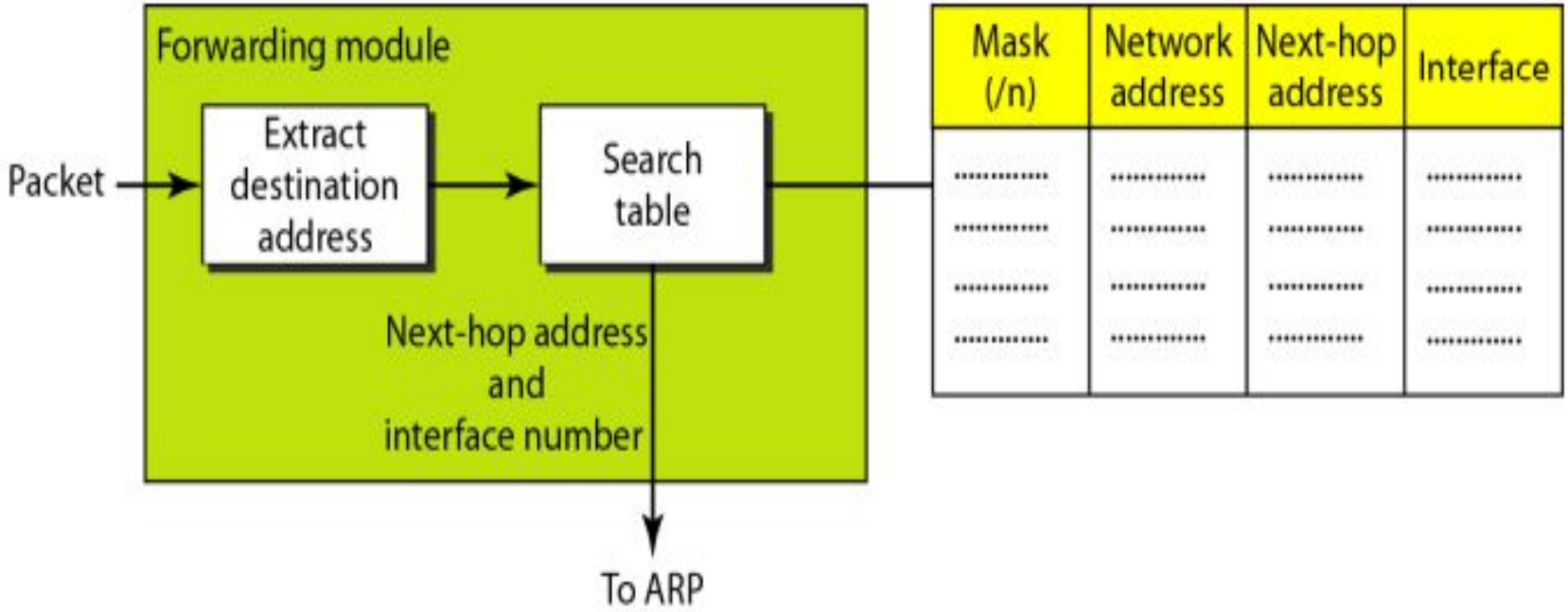
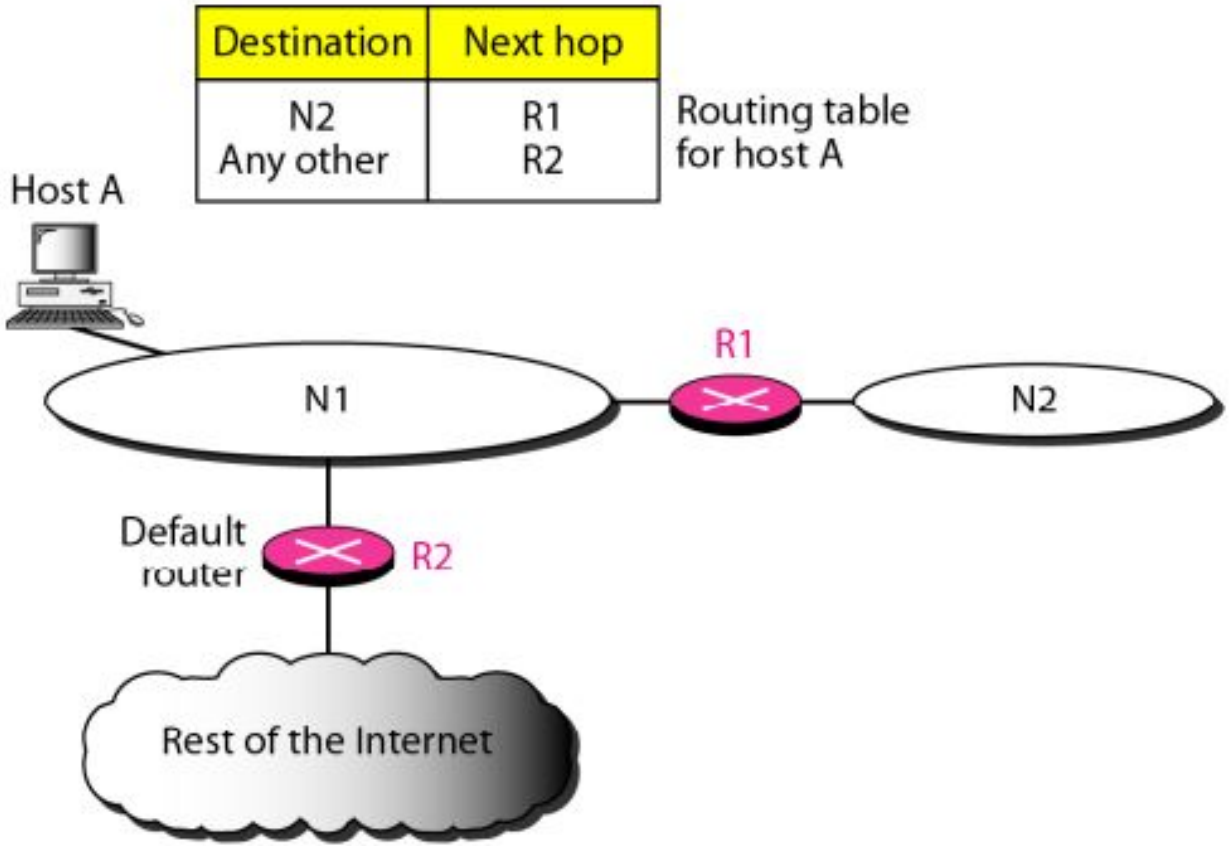
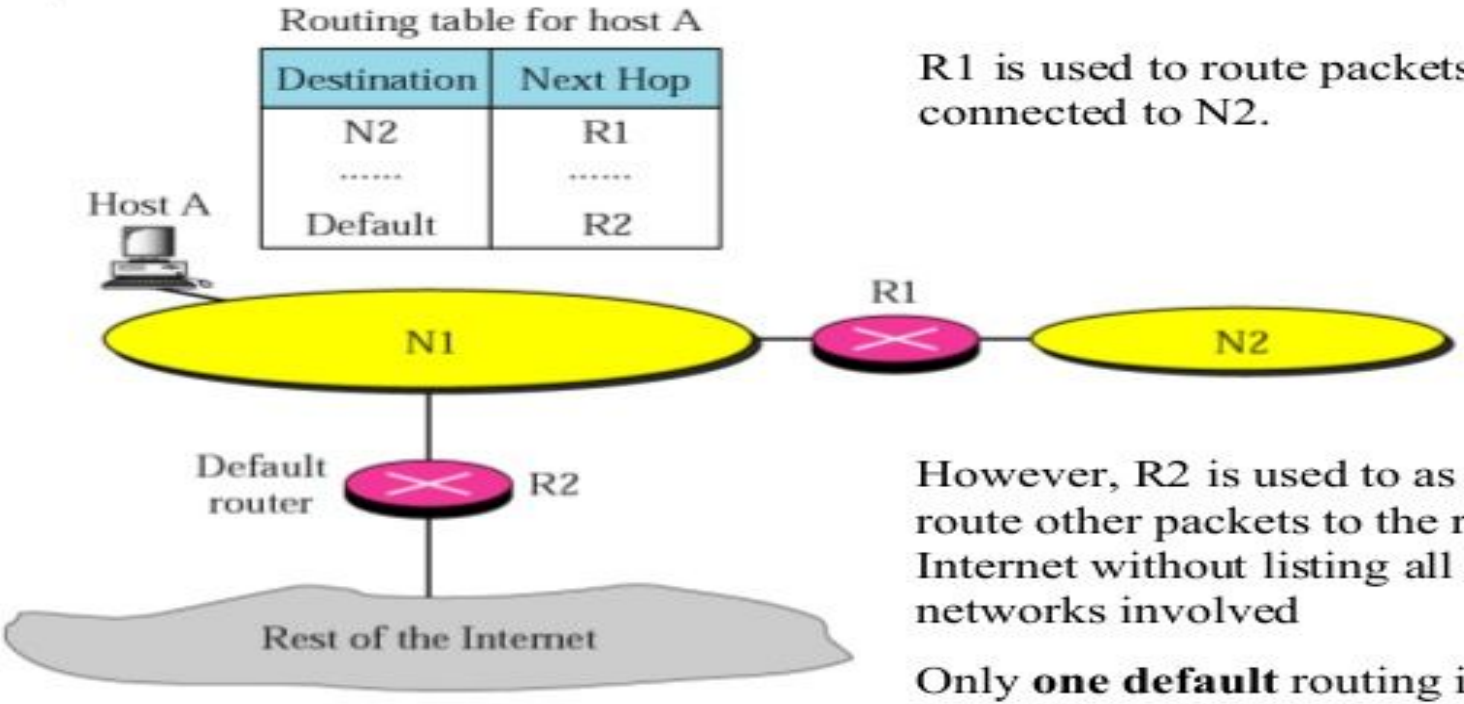

Forwarding module
Packet
Extract destination address
Search table
Next-hop address and interface number
To ARP
Mask (/n)
Network address
Next-hop address
Interface

**Figure 22.4** *Default method*



# Default routing




R1 is used to route packets to hosts connected to N2.

However, R2 is used to as default to route other packets to the rest of Internet without listing all the networks involved

Only **one default** routing is allowed with network address 0.0.0.0

# Example: make routing table for R1.

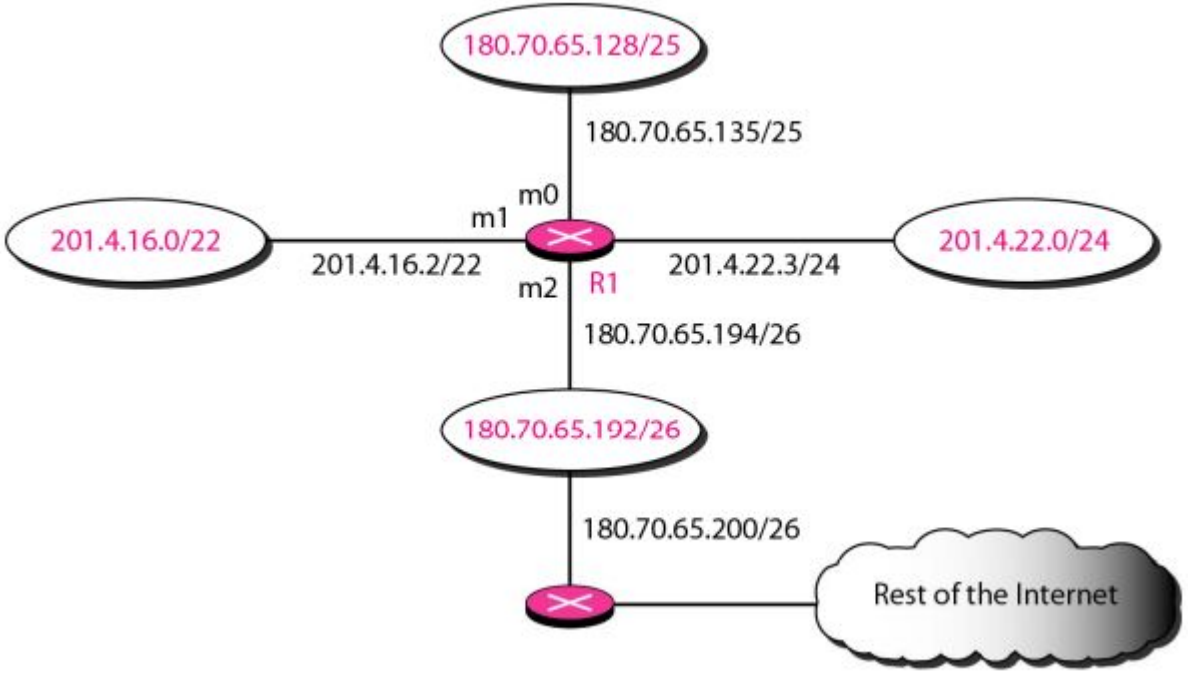


**Table 22.1** ***Routing table for router R1 in Figure 22.6***

| *Mask* | *Network Address* | *Next Hop* | *Interface* |
|---|---|---|---|
| /26 | 180.70.65.192 | — | m2 |
| /25 | 180.70.65.128 | — | m0 |
| /24 | 201.4.22.0 | — | m3 |
| /22 | 201.4.16.0 | .... | m1 |
| Any | Any | 180.70.65.200 | m2 |

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

*The router performs the following steps:*

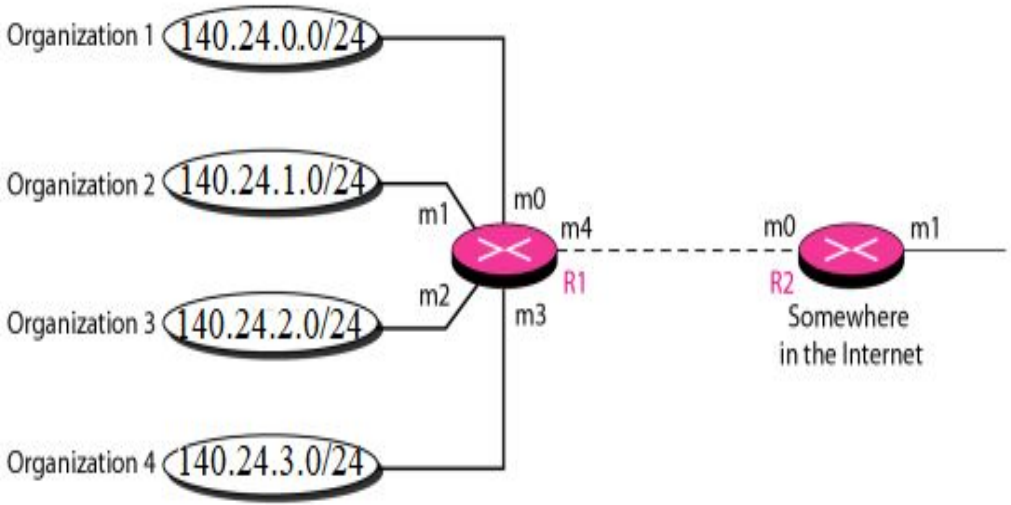
1. *The first mask (/26) is applied to the destination address. The result is 180.70.65.128, which does not match the corresponding network address.*
2. *The second mask (/25) is applied to the destination address. The result is 180.70.65.128, which matches the corresponding network address. The next-hop address and the interface number m0 are passed to ARP for further processing.*

*Show the forwarding process if a packet arrives at R1 in with the destination address 18.24.32.78.*

## *Solution*

*This time all masks are applied, one by one, to the destination address, but no matching network address is found. When it reaches the end of the table, the module gives the next-hop address 180.70.65.200 and interface number m2 to ARP. This is probably an outgoing package that needs to be sent, via the default router, to someplace else in the Internet.*

# Address aggregation




Routing table for R2

| | | |
|---|---|---|
| 24 | 140.24.0.0 | m0 |
| 24 | 140.24.1.0 | m0 |
| 24 | 140.24.2.0 | m0 |
| 24 | 140.24.3.0 | m0 |

Can we aggregate??

140.24.0.0⇒10001100 00011000 00000000 00000000

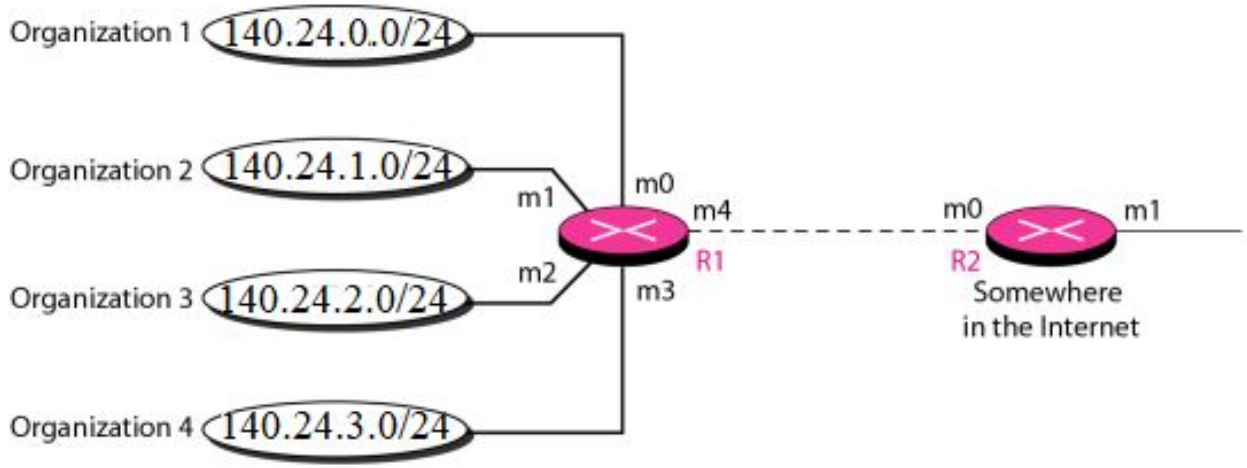
140.24.1.0⇒ 10001100 00011000 00000001 00000000

140.24.2.0⇒10001100 00011000 00000010 00000000

140.24.3.0⇒10001100 00011000 00000011 00000000

What is the network address? What is the mask of aggregated address?

# Address aggregation



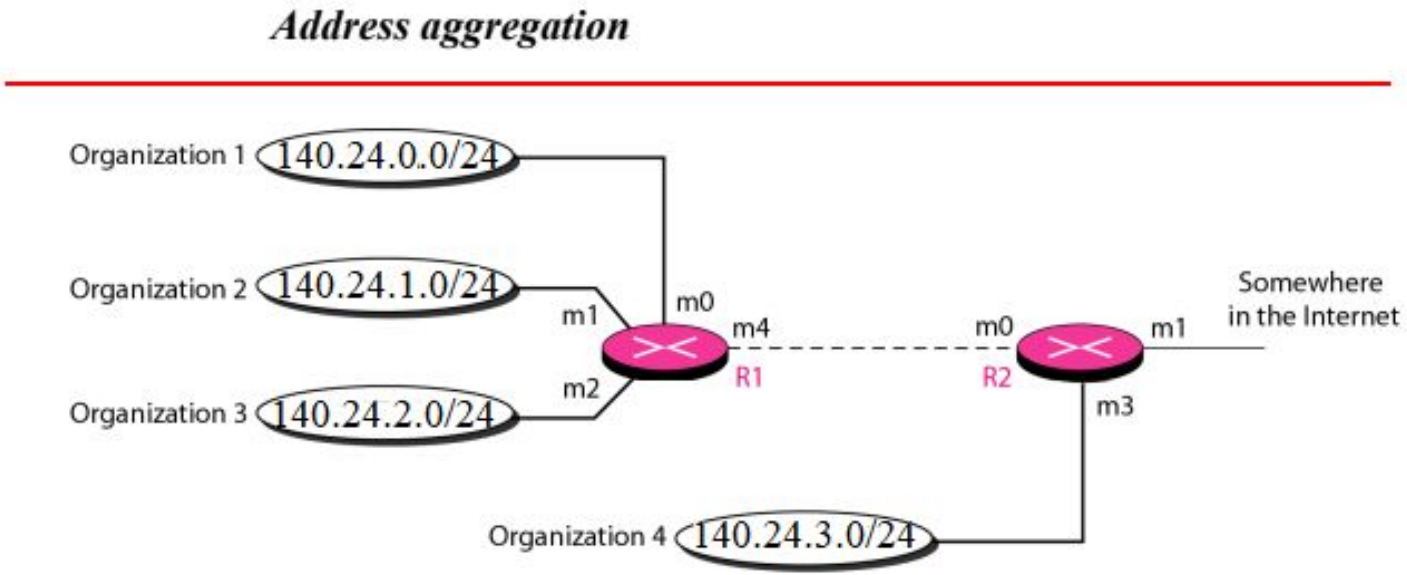

| Mask | Network address | Next-hop address | Interface |
|---|---|---|---|
| /24 | 140.24.0.0 | ---------- | m0 |
| /24 | 140.24.1.0 | ---------- | m1 |
| /24 | 140.24.2.0 | ---------- | m2 |
| /24 | 140.24.3.0 | ---------- | m3 |
| /0 | 0.0.0.0 | Default | m4 |

Routing table for R1

| Mask | Network address | Next-hop address | Interface |
|---|---|---|---|
| /22 | 140.24.0.0 | ---------- | m0 |
| /0 | 0.0.0.0 | Default | m1 |

Routing table for R2

# What if?
# Write routing table of R2...



# *Longest mask matching*

| Mask | Network address | Next-hop address | Interface |
|---|---|---|---|
| /24 | 140.24.3.0 | ---------- | m3 |
| /22 | 140.24.0.0 | ---------- | m0 |
| /0 | 0.0.0.0 | Default | m1 |

Routing table for R2

*Note*

**An IPv6 address is 128 bits long.**

**Figure 19.14** *IPv6 address in binary and hexadecimal colon notation*

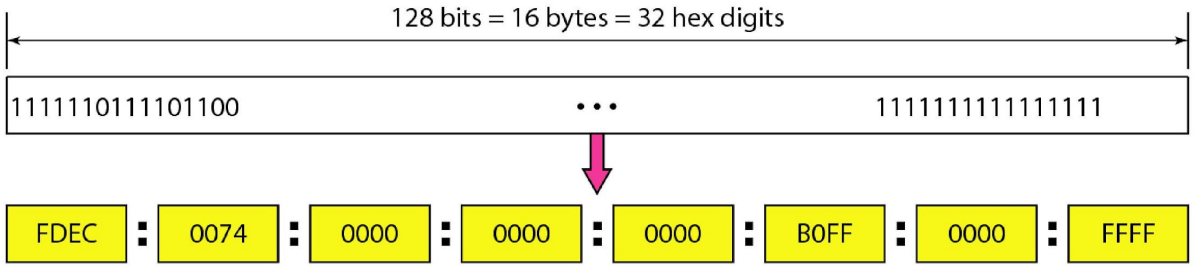
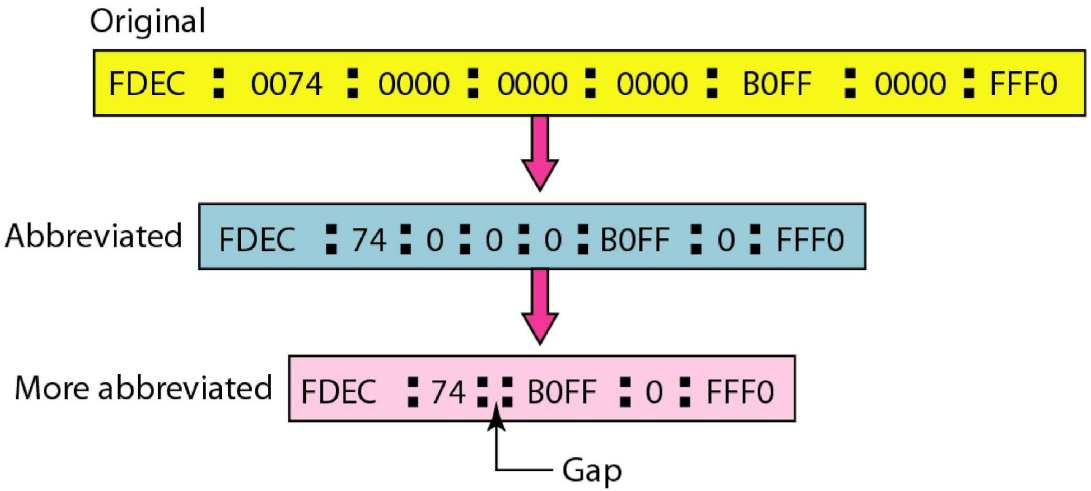


**Figure 19.15** *Abbreviated IPv6 addresses*



## Example 19.11

*Expand the address 0:15::1:12:1213 to its original.*

### Solution

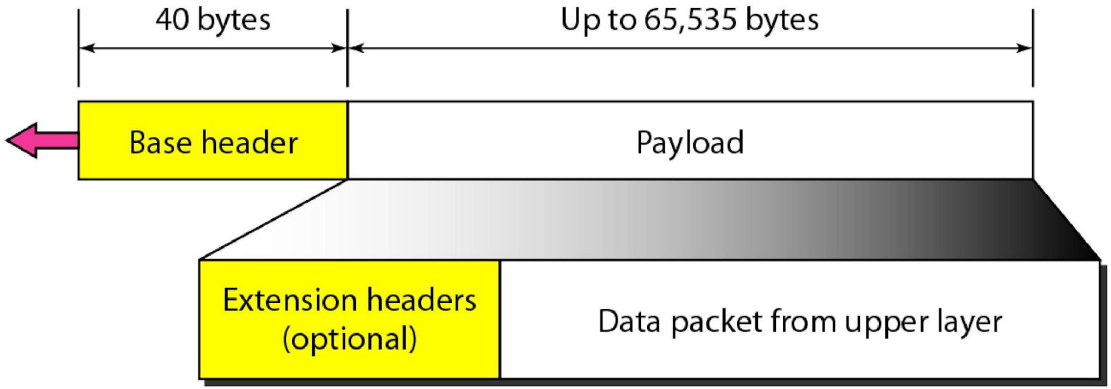
*We first need to align the left side of the double colon to the left of the original pattern and the right side of the double colon to the right of the original pattern to find how many 0s we need to replace the double colon.*

```
xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
 0:   15:                   :    1:   12:1213
```

*This means that the original address is.*

```
0000:0015:0000:0000:0000:0001:0012:1213
```

**Figure 20.15** *IPv6 datagram header and payload*



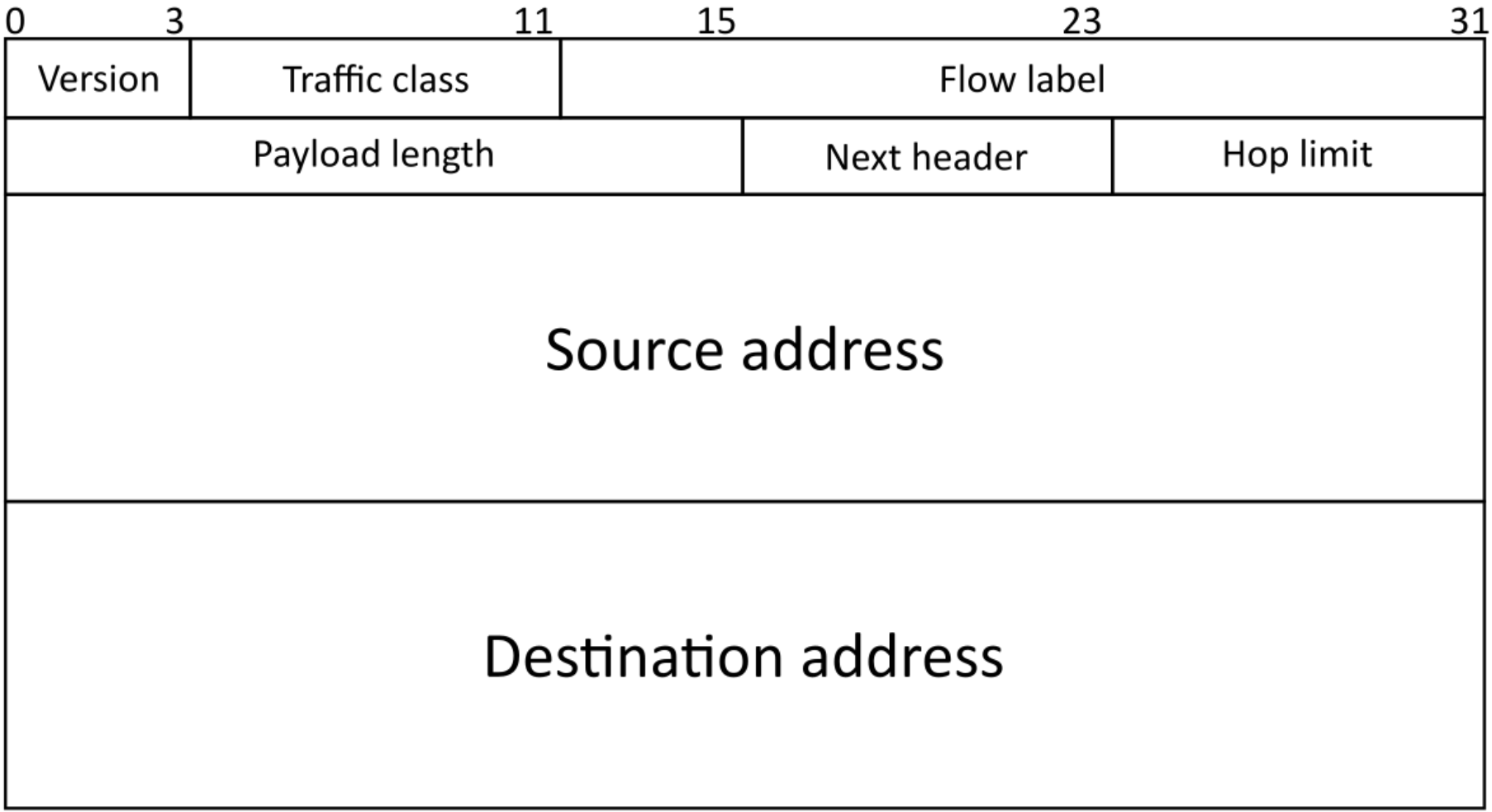

0
3
11
15
23
31
Version
Traffic class
Flow label
Payload length
Next header
Hop limit
Source address
Destination address

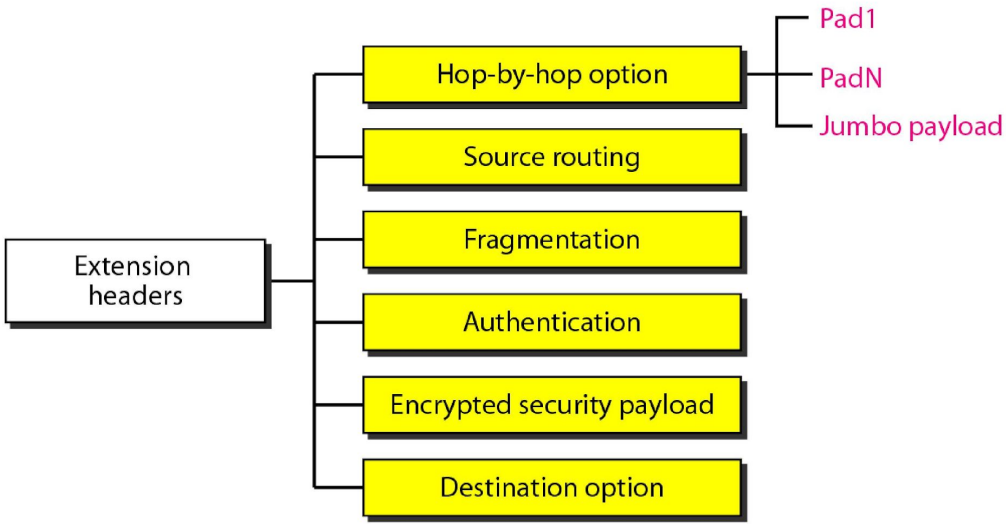
**Table 20.6** *Next header codes for IPv6*

| *Code* | *Next Header* |
|---|---|
| 0 | Hop-by-hop option |
| 2 | ICMP |
| 6 | TCP |
| 17 | UDP |
| 43 | Source routing |
| 44 | Fragmentation |
| 50 | Encrypted security payload |
| 51 | Authentication |
| 59 | Null (no next header) |
| 60 | Destination option |

## Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

| *Comparison* |
| --- |
| 1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version. |
| 2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field. |
| 3. The total length field is eliminated in IPv6 and replaced by the payload length field. |
| 4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header. |
| 5. The TTL field is called hop limit in IPv6. |
| 6. The protocol field is replaced by the next header field. |
| 7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level. |
| 8. The option fields in IPv4 are implemented as extension headers in IPv6. |

## Figure 20.17 *Extension header types*



## Table 20.10 *Comparison between IPv4 options and IPv6 extension headers*

| *Comparison* |
| --- |
| 1. The no-operation and end-of-option options in IPv4 are replaced by Pad1 and PadN options in IPv6. |
| 2. The record route option is not implemented in IPv6 because it was not used. |
| 3. The timestamp option is not implemented because it was not used. |
| 4. The source route option is called the source route extension header in IPv6. |
| 5. The fragmentation fields in the base header section of IPv4 have moved to the fragmentation extension header in IPv6. |
| 6. The authentication extension header is new in IPv6. |
| 7. The encrypted security payload extension header is new in IPv6. |