UNIT III NETWORK LAYER

Switching : Packet Switching – Internet protocol – IPV4 – IP Addressing – Subnetting – IPV6,ARP,RARP, ICMP, DHCP

1.SWITCHING

The Internet is made of many networks (or links) connected through the connecting devices. In other words, the Internet is an internetwork, a combination of LANs and WANs.

Packetizing

The first duty of the network layer is definitely **packetizing:** encapsulating the payload (data received from upper layer in a network-layer packet at the source and decapsulating the payload from the network-layer packet at the destination.

Routing and Forwarding

Routing

The network layer is responsible for routing the packet from its source to the destination. A physical network is a combination of networks (LANs and WANs) and routers that connect them. This means that there is more than one route from the source to the destination. The network layer is responsible for finding the best one among these possible routes.

Forwarding

Forwarding can be defined as the action applied by each router when a packet arrives at one of its interfaces. The decision-making table a router normally uses for applying this action is sometimes called the forwarding table and sometimes the routing table. When a router receives a packet from one of its attached networks, it needs to forward the packet to another attached network (in unicast routing) or to some attached networks (in multicast routing).

2.PACKET SWITCHING

***Highlight the characteristics of datagram network(2)(Nov/Dec 2017)**

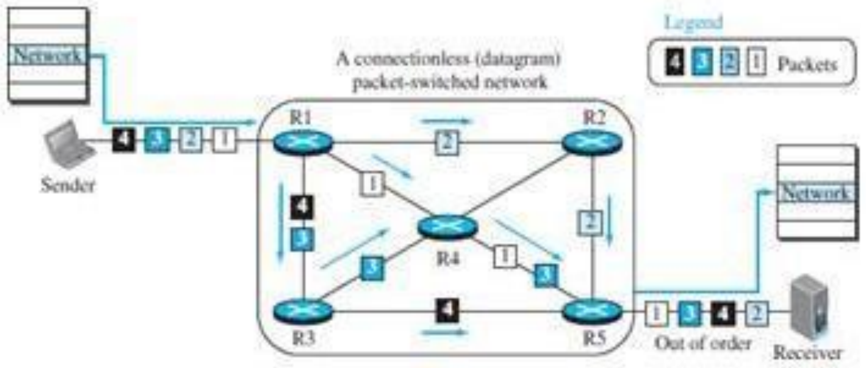
***Differentiate the packet switching and datagram approach(13)**

***Difference between the connection oriented and connection less service(7)**

A router, in fact, is a switch that creates a connection between an input port and an output port. In data communication switching techniques are divided into two broad categories, circuit switching and packet switching, only packet switching is used at the network layer because the unit of data at this layer is a packet. Packet-switched network can use two different approaches to route the packets: the datagram approach and the virtual circuit approach.

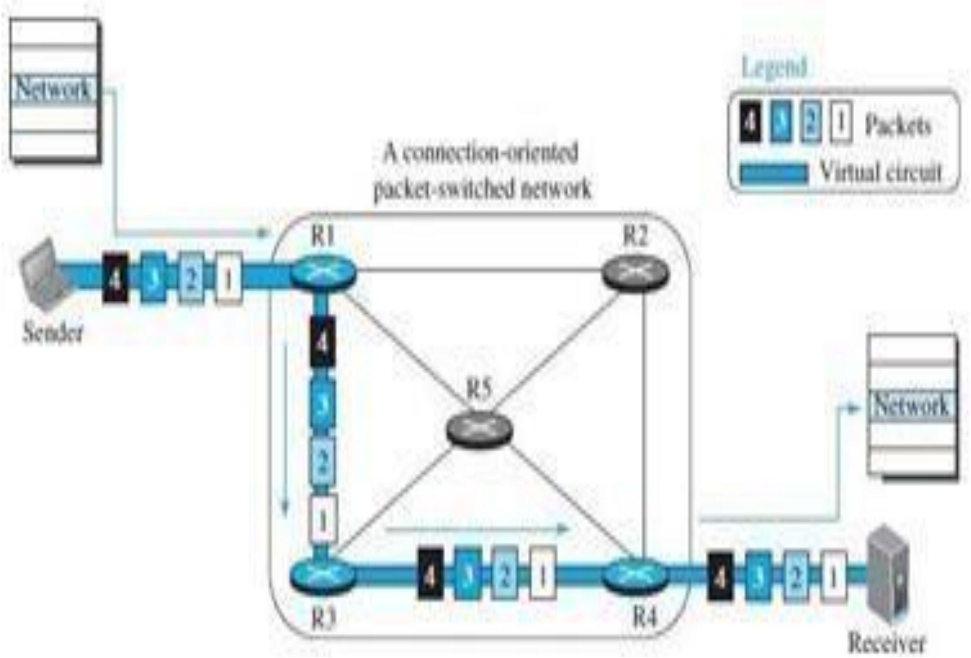
Datagram Approach: Connectionless Service

When the network layer provides a connectionless service, each packet traveling in the Internet is an independent entity; there is no relationship between packets belonging to the same message. The switches in this type of network are called *routers*. A packet belonging to a message may be followed by a packet belonging to the same message or to a different message. A packet may be followed by a packet coming from the same or from a different source.Each packet is routed based on the information contained in its header: source and destination addresses. The destination address defines where it should go; the source address defines where it comes from. The router in this case routes the packet based only on the destination address. The source address may be used to send an error message to the source if the packet is discarded. Figure below shows the forwarding process in a router in this case. We have used symbolic addresses such as A and B.



Virtual-Circuit Approach: Connection-Oriented Service

In a connection-oriented service (also called *virtual-circuit approach*), there is a relationship between all packetsbelonging to a message. Before all datagrams in a message can be sent, a virtual connection should be set up todefine the path for the datagrams. After connection setup, the datagrams can all follow the same path. In this type ofservice, not only must the packet contain the source and destination addresses, it must also contain a flow label, a virtual circuit identifier that defines the virtual path the packet should follow. Figure below shows the concept of connection-oriented service.



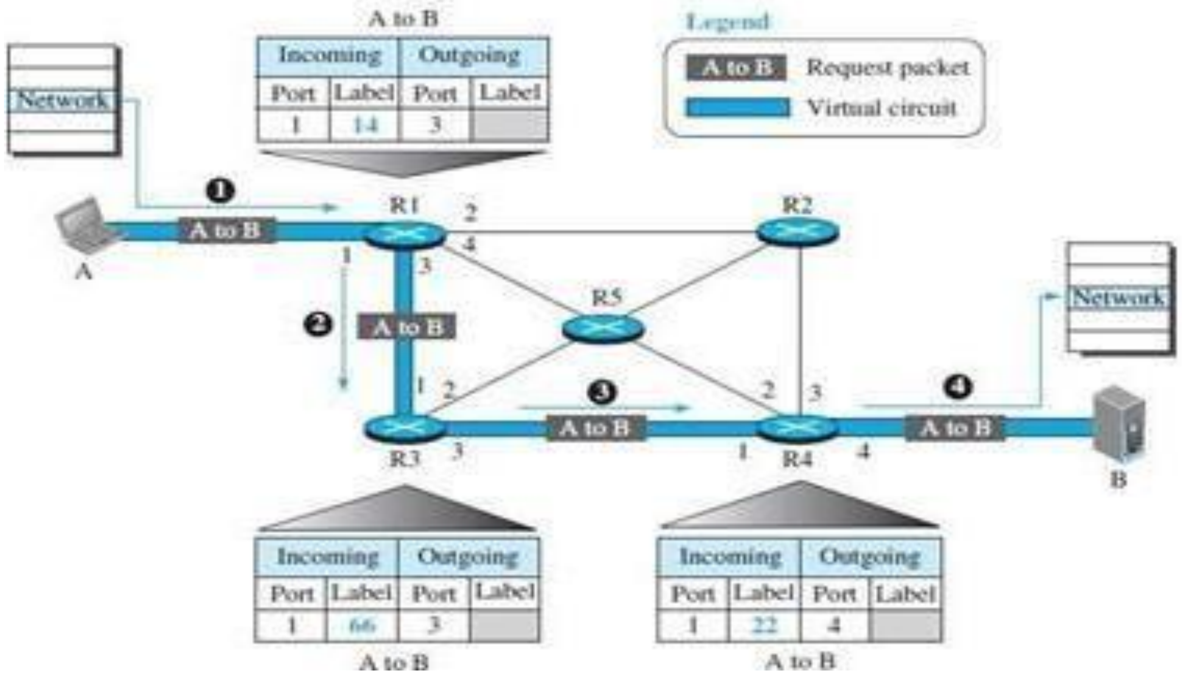
Each packet is forwarded based on the label in the packet. To follow the idea of connection-oriented design to be used in the Internet, we assume that the packet has a label when it reaches the router. Figure below shows the idea. In this case, the forwarding decision is based on the value of the label, or *virtual circuit identifier,* as it is sometimes called. To create a connection-oriented service, a three-phase process is used: setup, data transfer, and teardown. In the setup phase, the source and destination addresses of the sender and receiver are used to make table entries for the connection- oriented service. In the teardown phase, the source and destination inform the router to delete the corresponding entries. Data transfer occurs between these two phases.

Setup Phase

In the setup phase, a router creates an entry for a virtual circuit. For example, suppose source A needs to create a virtual circuit to destination B. Two auxiliary packets need to be exchanged between the sender and the receiver: the request packet and the acknowledgment packet.

Request packet

A request packet is sent from the source to the destination. This auxiliary packet carries the source and destination addresses.



A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	14	3	

Incoming		Outgoing	
Port	Label	Port	Label
1	66	3	

A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	22	4	

A to B

1. Source A sends a request packet to router R1.

2. Router R1 receives the request packet. It knows that a packet going from A to B goes out through port 3. How the router has obtained this information is a point covered later. For the moment, assume that it knows the output port. The router creates an entry in its table for this virtual circuit, but it is only able to fill three of the four columns. The router assigns the incoming port (1) and chooses an available incoming label (14) and the outgoing port (3). It does not yet know the outgoing label, which will be found during the acknowledgment step. The router then forwards the packet through port 3 to router R3.

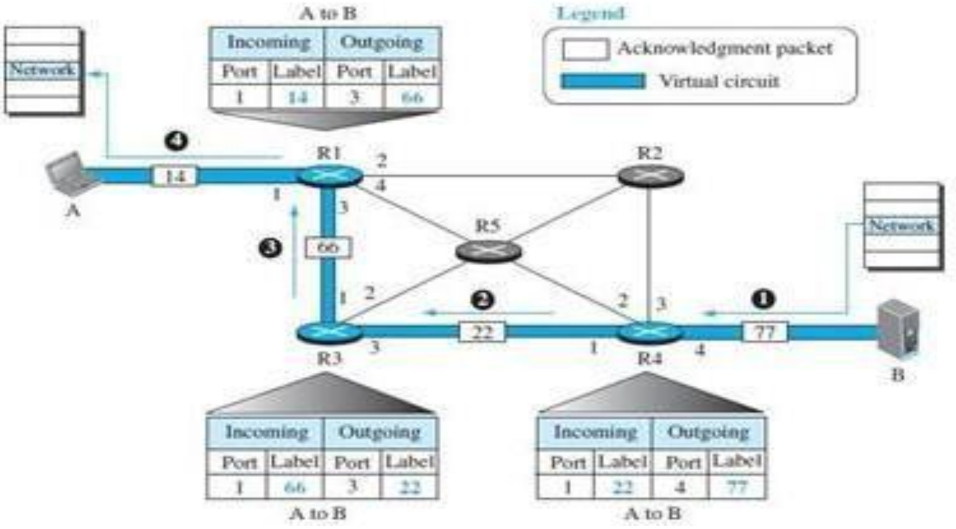
3. Router R3 receives the setup request packet. The same events happen here as at router R1; three columns of the table are completed: in this case, incoming port (1), incoming label (66), and outgoing port (3).

4. Router R4 receives the setup request packet. Again, three columns are completed: incoming port (1), incoming label (22), and outgoing port (4).

5. Destination B receives the setup packet, and if it is ready to receive packets from A, it assigns a label to the incoming packets that come from A, in this case 77, as shown in Figure below. This label lets the destination know that the packets come from A, and not from other sources.

Acknowledgment Packet

A special packet, called the acknowledgment packet, completes the entries in the switching tables.



A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	14	3	66

Incoming		Outgoing	
Port	Label	Port	Label
1	66	3	22

A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	22	4	77

A to B

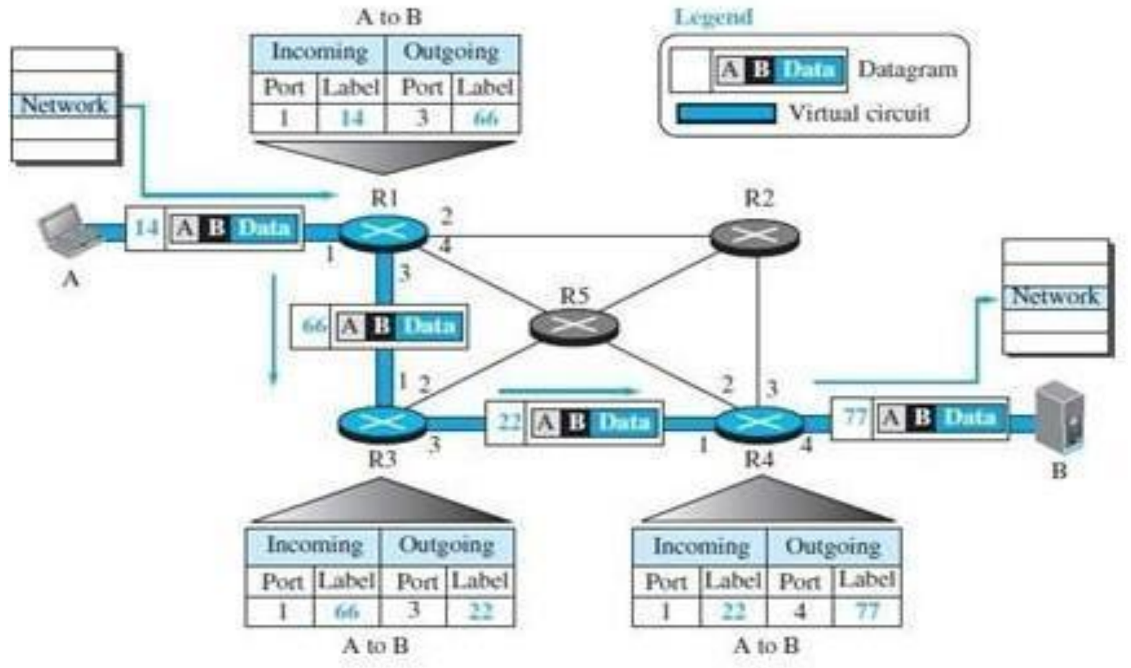
The destination sends an acknowledgment to router R4. The acknowledgment carries the global source and destination addresses so the router knows which entry in the table is to be completed. The packet also carries label77, chosen by the destination as the incoming label for packets from A. Router R4 uses this label to complete the outgoing label column for this entry. Note that 77 is the incoming label for destination B, but the outgoing label for router R4. Router R4 sends an acknowledgment to router R3 that contains its incoming label in the table, chosen in the setup phase. Router R3 uses this as the outgoing label in the table. Router R3 sends an acknowledgment to router R1 that contains its incoming label in the table, chosen in the setup phase. Router R1 uses this as the outgoing label in the table.
Finally router R1 sends an acknowledgment to source A that contains its incoming label in the table, chosen in the setup phase.
The source uses this as the outgoing label for the data packets to be sent to destination B.

Data-Transfer Phase

The second phase is called the data-transfer phase. After all routers have created their forwarding table for a specific virtual circuit, then the network-layer packets belonging to one message can be sent one after another. In Figure below, we show the flow of a single packet, but the process is the same for 1, 2, or 100 packets. The source computer uses the label 14, which it has received from router R1 in the setup phase. Router R1 forwards the packet to router R3, but changes the label to 66.Router R3 forwards the packet to router R4, but changes the label to 22. Finally, router R4 delivers the packet to its final destination with the label 77. All the packets in the message follow the same sequence of labels, and the packets arrive in order at the destination.

Teardown Phase

In the teardown phase, source A, after sending all packets to B, sends a special packet called a teardown packet. Destination B responds with a confirmation packet. All routers delete the corresponding entries from their tables.



A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	14	3	66

Incoming		Outgoing	
Port	Label	Port	Label
1	66	3	22

A to B

Incoming		Outgoing	
Port	Label	Port	Label
1	22	4	77

A to B

3. IPV4 ADDRESSES

***Draw the IPV4 Packet header format(Nov/Dec 2018)(13)**

*** Explain in detail about Subnetting and Supernetting(8)**

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

Address Space

A protocol like IPv4 that defines addresses has an address space. An **address space** is the total number of addressesused by the protocol. If a protocol uses *b* bits to define an address, the address space is 2*b* because each bit can have two different values (0 or 1). IPv4 uses 32-bit addresses, which means that the address space is 232 or 4,294,967,296 (more than four billion)

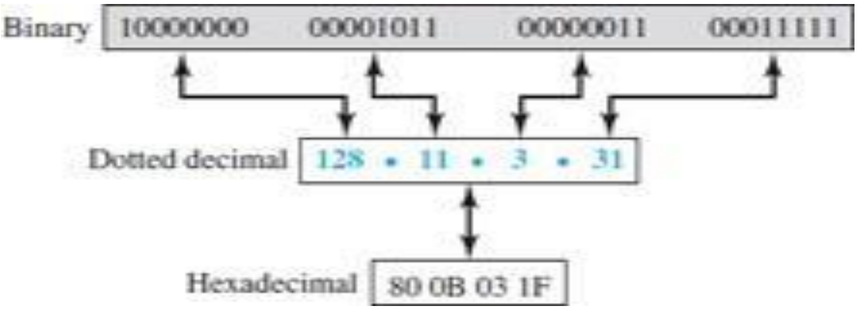
Notation

There are three common notations to show an IPv4 address:

1.Binary notation (base

2),dotted-decimal notation (base 256), and hexadecimal notation (base 16).

In binary notation, an IPv4 address is displayed as 32 bits. To make the address more readable, one or more spaces are usually inserted between each octet (8 bits). Each octet is often referred to as a byte. To make the IPv4 address more compact and easier to read, it is usually written in decimal form with a decimal point (dot) separating the bytes. This format is referred to as dotted-decimal notation. Note that because each byte (octet) is only 8 bits,each number in the dtted-decimal notation is between 0 and 255. We sometimes see an IPv4 address in hexadecimal notation. Each hexadecimal digit is equivalent to four bits.This means that a 32-bit address has 8hexadecimal digits.



Hierarchy in Addressing

A 32-bit IPv4 address is also hierarchical, but divided only into two parts. The first part of the address, called the *prefix*, defines the network; the second part of the address, called the *suffix*, defines the node (connection of a device to the Internet). The prefix length is *n* bits and the suffix length is (32 - *n*) bits.

Classful Addressing

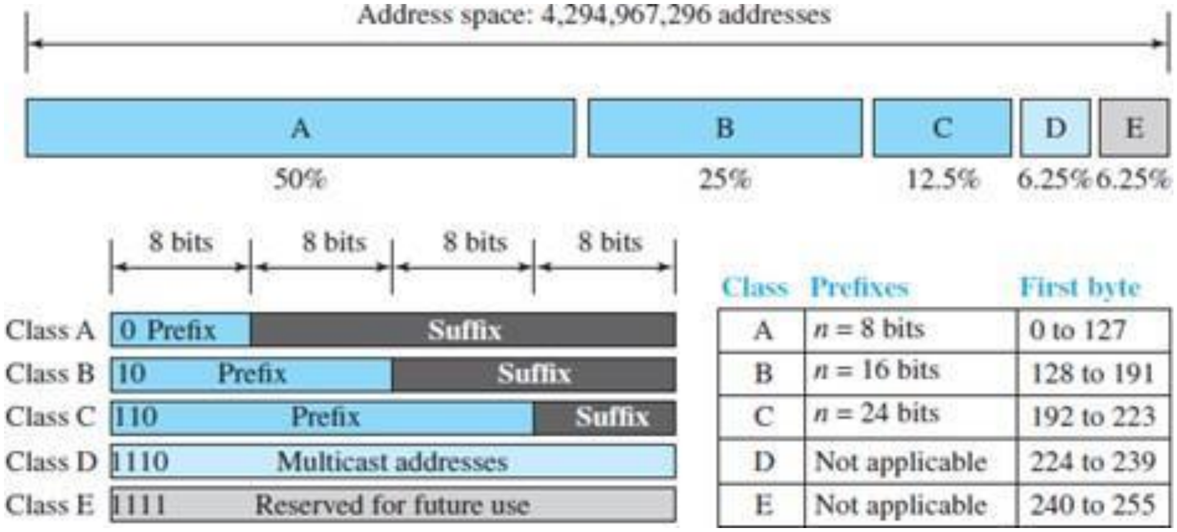
When the Internet started, an IPv4 address was designed with a fixed-length prefix, but to accommodate both small and large networks, three fixed-length prefixes were designed instead of one (*n* =8, *n* =16, and *n* =24). The whole address space was divided into five classes (class A, B, C, D, and E), as shown in Figure 18.18. This scheme is referred to as **classful addressing.**

In class A, the network length is 8 bits, but since the first bit, which is 0, defines the class, we can have only seven bits as the network identifier.

This means there are only 2^7 =128 networks in the world that can have a class A address.

In class B, the network length is 16 bits, but since the first two bits, which are(10)2, define the class, we can have only 14 bits as the network identifier. This means there are only 2^{14} =16,384 networks in the world that can have a class B address. All addresses that

start with (110)2 belong to class C. In class C, the network
length is 24 bits, but since three bits define the class, we can have only 21 bits as the network identifier. This means there are 2^{21} = 2,097,152 networks in the world that can have a class C address. Class D is not divided into prefix andsuffix. It is used for multicast addresses. All addresses that start with 1111 in binary belong to class E. As in Class D, Class E is not divided into prefix and suffix and is used as reserve.



Class	Prefixes	First byte
A	n = 8 bits	0 to 127
B	n = 16 bits	128 to 191
C	n = 24 bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Address Depletion

The reason that classful addressing has become obsolete is address depletion. Since the addresses were not distributed properly, the Internet was faced with the problem of the addresses being rapidly used up, resulting in no more addresses available for organizations and individuals that needed to be connected to the Internet. To understand the problem, let us think about class A. This class can be assigned to only 128 organizations in the world, but each organization needs to have a single network (seen by the rest of the world) with 16,777,216 nodes (computers in this single network). Since there may be only a few organizations that are this large, most of the addresses in this class were wasted (unused). Class B addresses were designed for midsize organizations, but many of the addresses in this class also remained unused. Class C addresses have a completely different flaw in design. The number of addresses that can be used in each network (256) was so small that most companies were not comfortable using a block in this address class. Class E addresses were almost never used, wasting the whole class.

4.SUBNETTING

To alleviate address depletion, two strategies were proposed and, to some extent, implemented: subnetting and supernetting. In subnetting, a class A or class B block is divided into several subnets. Each subnet has a larger prefix length than the original network. For example, if a network in class A is divided into four subnets, each subnet has a prefix of nsub = 10. At the same time, if all of the addresses in a network are not used, subnetting allows the addresses to be divided among several organizations. This idea did not work because most large organizations were not happy about dividing the block and giving some of the unused addresses to smaller organizations. While subnetting was devised to divide a large block into smaller ones, supernetting was devised to combine several class C blocks into a larger block to be attractive to organizations that need more than the 256 addresses available in a class C block. This idea did not work either because it makes the routing of packets more difficult.

Classless Addressing

Subnetting and supernetting in classful addressing did not really solve the address depletion problem. With the growth of the Internet, it was clear that a larger address space was needed as a long-term solution. The larger address space, however, requires that the length of IP addresses also be increased, which means the format of the IP packets needs to be changed.

In 1996, the Internet authorities announced a new architecture called **classless addressing.** In classless addressing, variable-length blocks are used that belong to no classes. We can have a block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on.

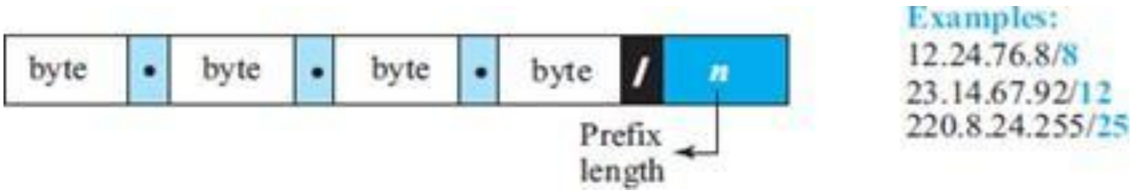
In classless addressing, the whole address space is divided into variable length blocks. The prefix in an address defines the block (network); the suffix defines the node (device).

Theoretically, we can have a block of 2^0, 2^1, 2^2,

.....2^{32} addresses. One of the restrictions, as we discuss later, is that the number of addresses in a block needs to be a power of 2. An organization can be granted one block of addresses.

Prefix Length: Slash Notation

The first question that we need to answer in classless addressing is how to find the prefix length if an address is given. Since the prefix length is not inherent in the address, we need to separately give the length of the prefix. In this case, the prefix length, n, is added to the address, separated by a slash. The notation is informally referred to as slash notation and formally as **classless interdomain routing** or **CIDR** (pronounced cider) strategy.



Extracting Information from an Address

Given any address in the block, we normally like to know three pieces of information about the block to which the address belongs: the number of addresses, the first address in the block, and the last address. Since the value ofprefix length, *n*, is given, we can easily find these three pieces of information.

1. The number of addresses in the block is found as $N =2^{32}$-n.

2. To find the first address, we keep the *n* leftmost bits and set the (32 - *n*) rightmost bits all to 0s.

3. To find the last address, we keep the *n* leftmost bits and set the (32 - *n*) rightmost bits all to 1s.

EXAMPLE:

A classless address is given as 167.199.170.82/27. We can find the above three pieces of information as follows. The number of addresses in the network is $2^{32 - n} = 2^{5}$ = 32 addresses.

Address Mask

Another way to find the first and last addresses in the block is to use the address mask.The address mask is a 32-bit number in which the *n* leftmost bits are set to 1s and the rest of the bits (32 -*n*) are set to 0s. A computer can easily find the address mask because it is the complement of (2^{32-n} -1). The reason for defining a mask in this way is that it can be used by a computer program to extract the information in a block, using the three bit-wise operations NOT, AND, and OR.

1. The number of addresses in the block N =**NOT** (mask) +1.

2. The first address in the block =(Any address in the block) **AND** (mask).

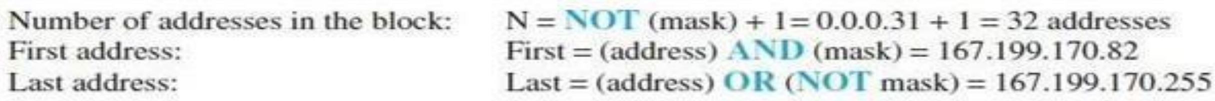
3. The last address in the block =(Any address in the block) **OR** [(**NOT** (mask)]. The mask in dotted-decimal notation is 256.256.256.224. The AND, OR, and NOT operations can be applied toindividual bytes using calculators

Address: 167.199.170.82/27 10100111 11000111 10101010 01010010

First address: 167.199.170.64/27 10100111 11000111 10101010 01000000

Address: 167.199.170.82/27 10100111 11000111 10101010 01011111

Last address: 167.199.170.95/27 10100111 11000111 10101010 01011111



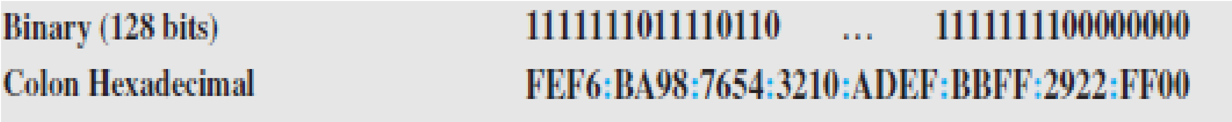
Number of addresses in the block:	N = NOT (mask) + 1= 0.0.0.31 + 1 = 32 addresses
First address:	First = (address) AND (mask) = 167.199.170.82
Last address:	Last = (address) OR (NOT mask) = 167.199.170.255

5.IPv6
PROTOCOL

IPv6 ADDRESSING

***Draw the IPV6 packet header format**

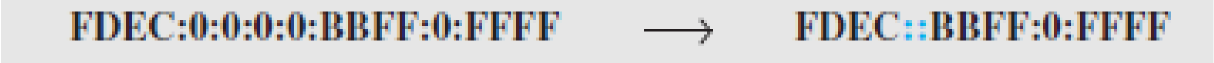
The main reason for migration from IPv4 to IPv6 is the small size of the address space in IPv4. An IPv6 address is128 bits or 16 bytes (octets) long, four times the address length in IPv4. An IPv6 address is 128 bits or 16 bytes (octets) long, four times the address length in IPv4.



Binary (128 bits)	1111111011110110 … 1111111100000000
Colon Hexadecimal	FEF6:BA98:7654:3210:ADEF:BBFF:2922:FF00

Abbreviation

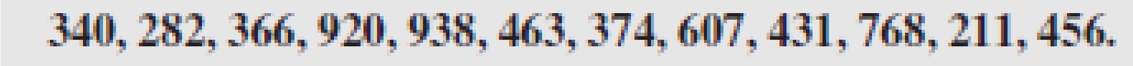
Although an IPv6 address, even in hexadecimal format, is very long, many of the digits are zeros. In this case, we can abbreviate the address. The leading zeros of a section can be omitted. Using this form of abbreviation, 0074 can be written as 74, 000F as F, and 0000 as 0. Note that 3210 cannot be abbreviated. Further abbreviation, often called **zero compression,** can be applied to colon hex notation if there are consecutive sections consisting of zeros only. We can remove all the zeros and replace them with a double semicolon.



FDEC:0:0:0:0:BBFF:0:FFFF ⟶ FDEC::BBFF:0:FFFF

Address Space

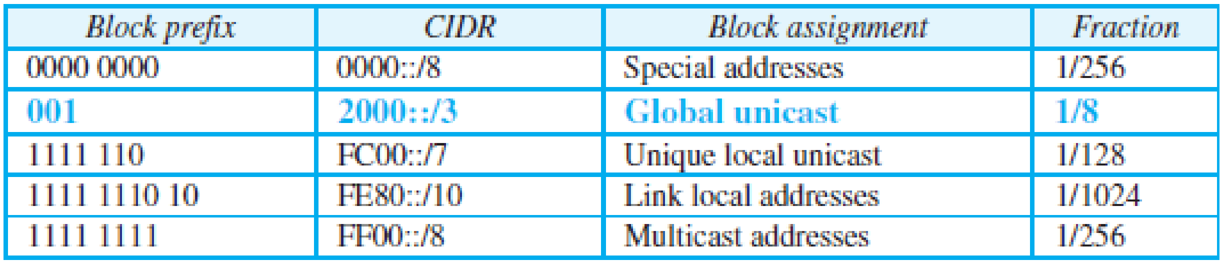
The address space of IPv6 contains 2128 addresses. This address space is 296 times the IPv4 address—definitely no address depletion—as shown, the size of the space is



340, 282, 366, 920, 938, 463, 374, 607, 431, 768, 211, 456.

Address Space Allocation

Like the address space of IPv4, the address space of IPv6 is divided into several blocks of varying size and each block is allocated for a special purpose. Most of the blocks are still unassigned and have been set aside for futureuse.



Block prefix	*CIDR*	*Block assignment*	*Fraction*
0000 0000	0000::/8	Special addresses	1/256
001	2000::/3	Global unicast	1/8
1111 110	FC00::/7	Unique local unicast	1/128
1111 1110 10	FE80::/10	Link local addresses	1/1024
1111 1111	FF00::/8	Multicast addresses	1/256

Global Unicast Addresses

The block in the address space that is used for unicast (one-to-one) communication between two hosts in the Internet is called the global unicast address block. CIDR for the block is 2000::**/3**, which means that the three leftmost bits are the same for all addresses in this block (001). The size of this block is 2125 bits, which is more than enough for Internet expansion for many years to come. An address in this block is divided into three parts: global routing prefix (n bits), subnet identifier (m bits), and interface identifier (q bits)

Auto configuration

One of the interesting features of IPv6 addressing is the **autoconfiguration** of hosts. As we discussed in IPv4, the host and routers are originally configured manually by the network manager. However, the Dynamic Host Configuration Protocol, DHCP, can be used to allocate an IPv4 address to a host that joins the network. In IPv6, DHCP protocol can still be used to allocate an IPv6 address to a host, but a host can also configure itself.

***Draw the IPV6 Packet Format(8)**

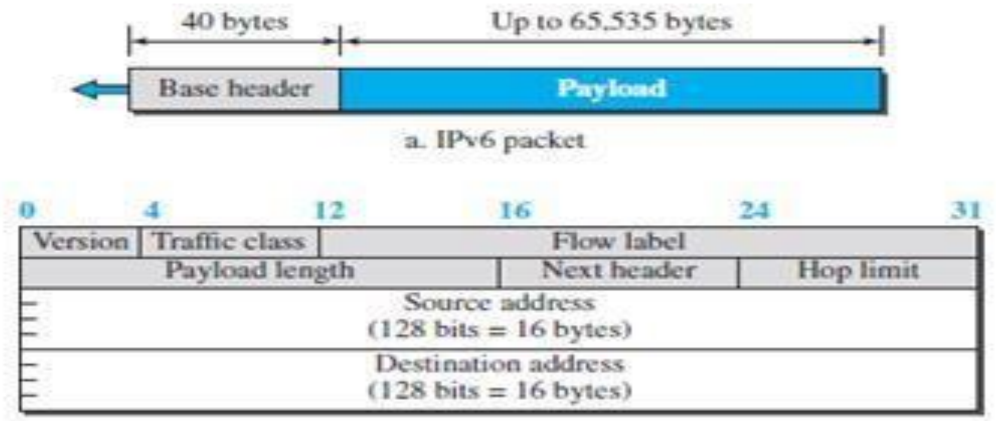
The change of the IPv6 address size requires the change in the IPv4 packet format. The designer of IPv6 decided to implement remedies for other shortcomings now that a change is inevitable. The following shows other changes implemented in the protocol in addition to changing address size and format.

- **Better header format.** IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
- **New options.** IPv6 has new options to allow for additional functionalities.
- **Allowance for extension.** IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
- **Support for resource allocation.** In IPv6, the type-of-service field has been removed, but two new fields, traffic class and flow label, have been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.
- **Support for more security.** The encryption and authentication options in IPv6

provide confidentiality and integrity of the packet.

Packet Format

Each packet is composed of a base header followed by the payload. The base header occupies 40 bytes, whereas payload can be up to 65,535 bytes of information. The description of fields follows.

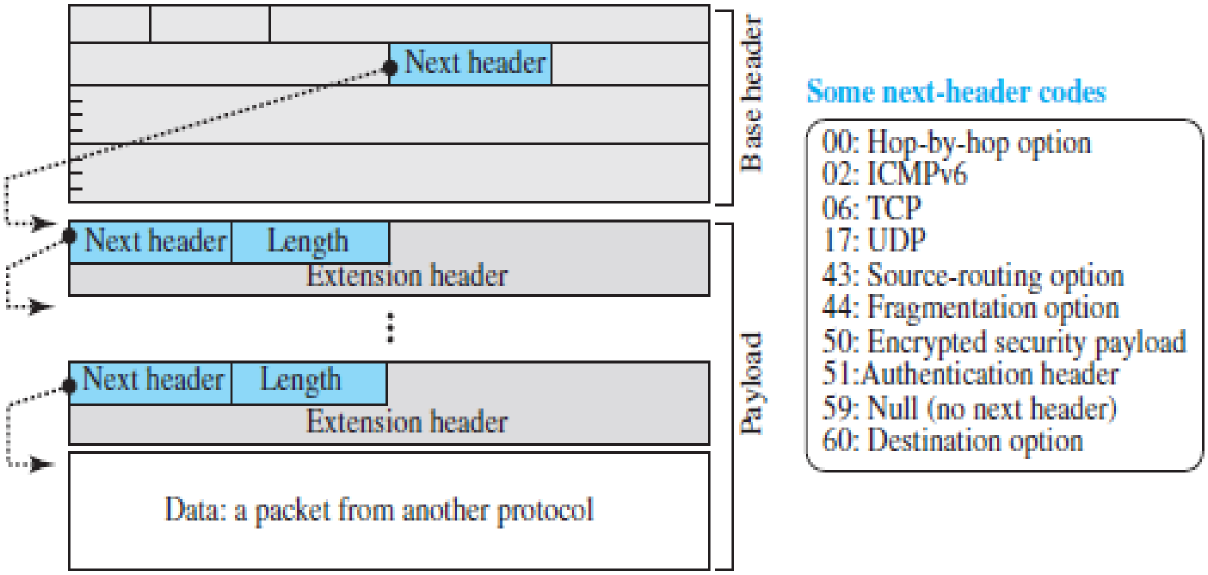


a. IPv6 packet

b. Base header

Version. The 4-bit version field defines the version number of the IP. For IPv6, the value is 6.

- **Traffic class.** The 8-bit traffic class field is used to distinguish different payloads with different deliveryrequirements. It replaces the type-of-service field in IPv4.
- **Flow label.** The flow label is a 20-bit field that is designed to provide special handling for a particular flow of data. We will discuss this field later.
- **Payload length.** The 2-byte payload length field defines the length of the IP datagram excluding the header. Note that IPv4 defines two fields related to the length: header length and total length. In IPv6, the length of the baseheader is fixed (40 bytes); only the length of the payload needs to be defined.
- **Next header.** The **next header** is an 8-bit field defining the type of the first extension header (if present) or the type of the data that follows the base header in the datagram. This field is similar to the protocol field in IPv4
- **Hop limit.** The 8-bit hop limit field serves the same purpose as the TTL field in IPv4.
- **Source and destination addresses.** The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram. The destination address field is a 16-byte (128-bit) Internet address that identifies the destination of the datagram.
- **Payload.** Compared to IPv4, the payload field in IPv6 has a different format and meaning, as shown in Figure below.



6.ARP

ARP: Address Resolution Protocol

ARP maps a logical address to a physical address

ARP Operation

To find the physical address of another host or router on its network n

Send an ARP request message

ARP request message

- The physical address of the sender
- The IP address of the sender
- The physical address of the receiver is 0s
- The IP address of the receiver

Then, ARP request message is broadcast by the physical layer

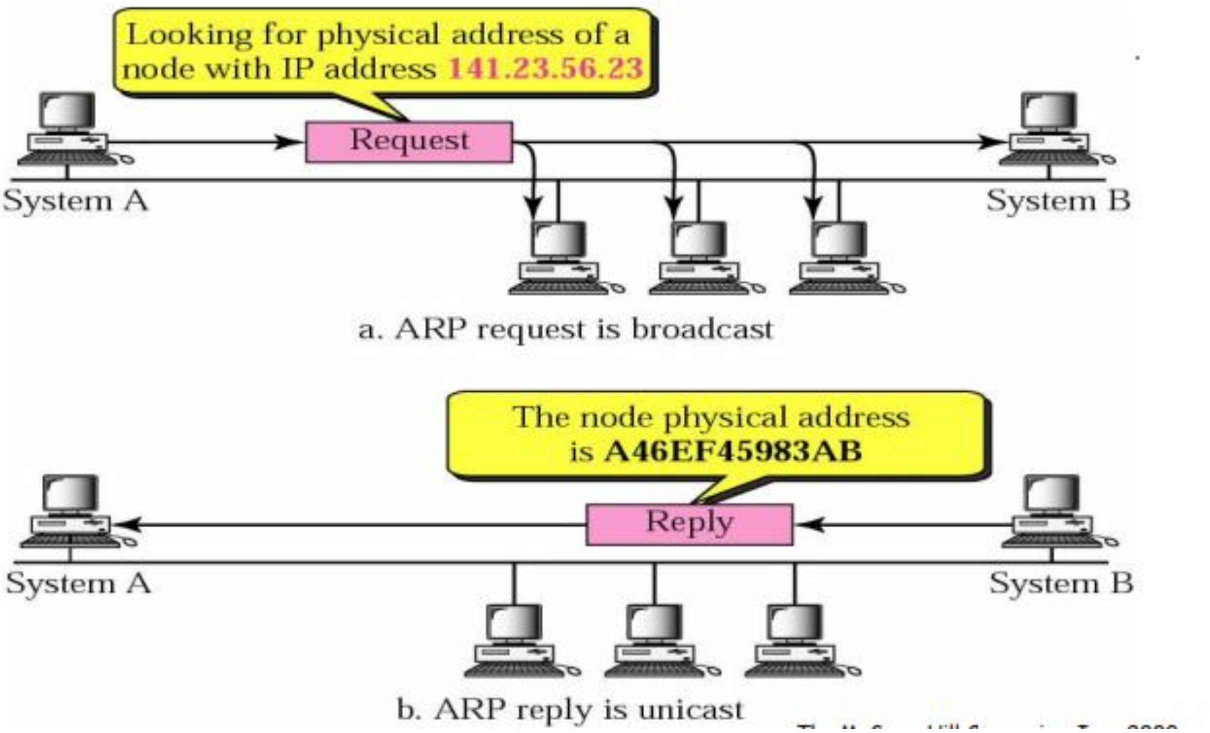
For example: in Ethernet, MAC header ’ s destination address is all 1s (broadcast address)

Received by every station on the physical network

The intended recipient send back an ARP reply message

ARP reply message packet is unicast

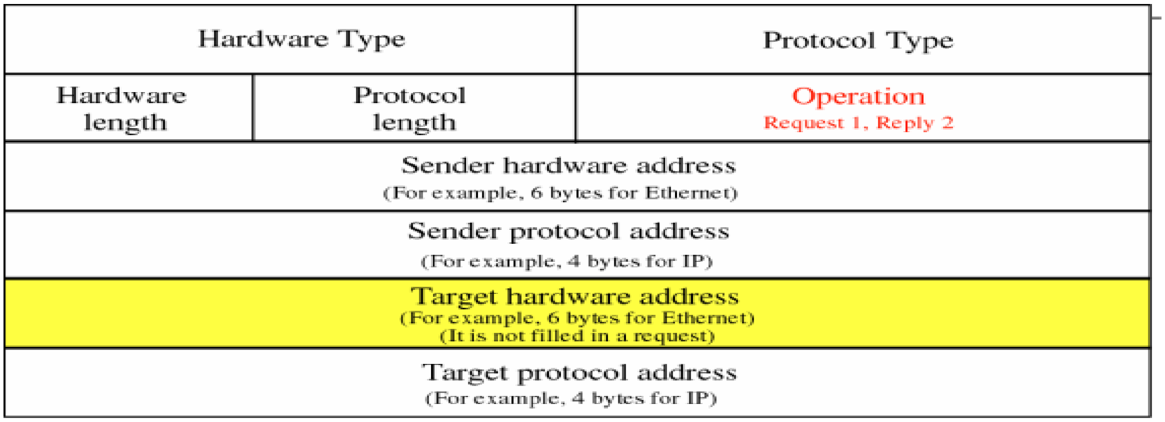
ARP Operation


Looking for physical address of a node with IP address 141.23.56.23
Request
System A
System B
The node physical address is A46EF45983AB
Reply
System A
System B

a. ARP request is broadcast

b. ARP reply is unicast

ARP Packet



Hardware Type		Protocol Type
Hardware length	Protocol length	Operation Request 1, Reply 2
Sender hardware address (For example, 6 bytes for Ethernet)		
Sender protocol address (For example, 4 bytes for IP)		
Target hardware address (For example, 6 bytes for Ethernet) (It is not filled in a request)		
Target protocol address (For example, 4 bytes for IP)		

Packet Format

- HTYPE (Hardware type):16-bit field defining the underlying type of the network
    - Ethernet is given the type 1
    - ARP can be used on any physical network

- PTYPE (Protocol type) :16-bit field defining the protocol
    - IPv4 is 080016
    - ARP can be used with any higher-level protocol

- HLEN (Hardware length) : 8-bit field defining the length of the physical address in bytes
    - Ethernet has the value of 6

- PLEN (Protocol length) :8-bit field defining the length of the logical address in bytes o IPv4 has the value of 4 o OPER (Operation) n 16-bit field defining the type of packet n (1) = ARP request, (2) = ARP reply

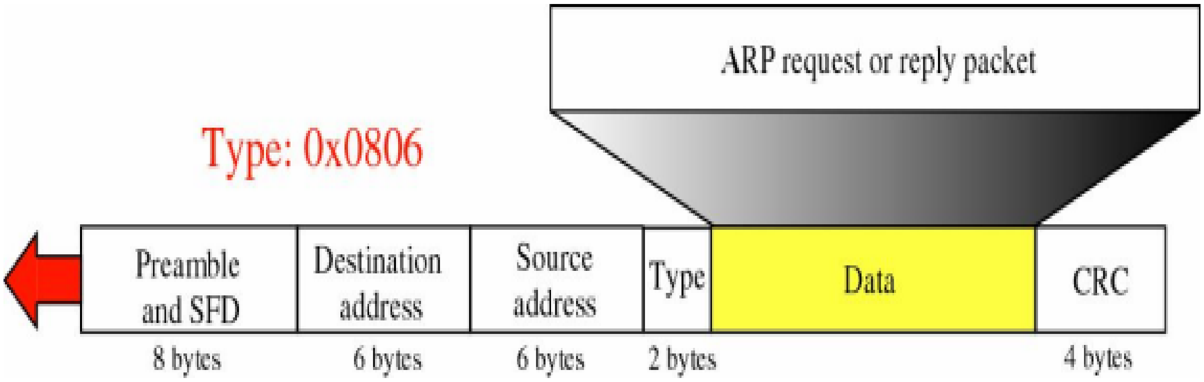
- SHA (Sender hardware address) :A variable-length field defining the physical address of the sender

- SPA (Sender protocol address) : A variable-length field defining the logical address of the sender

- THA (Target hardware address) : A variable-length field defining the physical address of the target

- For an ARP request operation packet o This field is all 0s o TPA (Target protocol address) n A variable-length field defining the logical address of the target

Encapsulation of ARP Packet

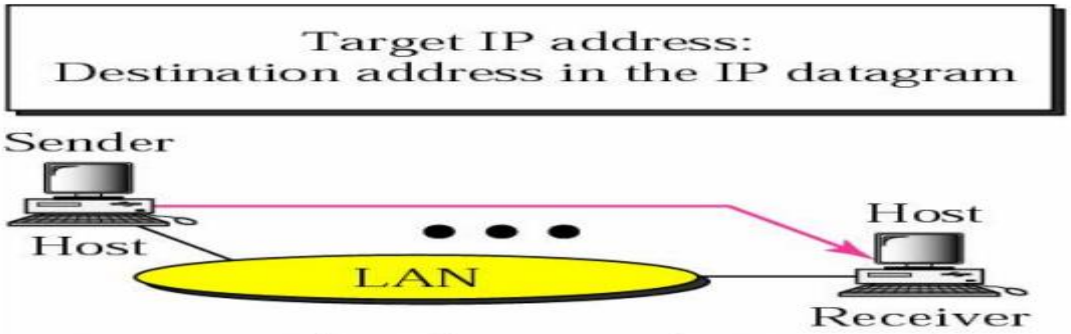


- An ARP packet is encapsulated directly into a data link frame
- Type field indicates that the data carried by the frame is an ARP packet

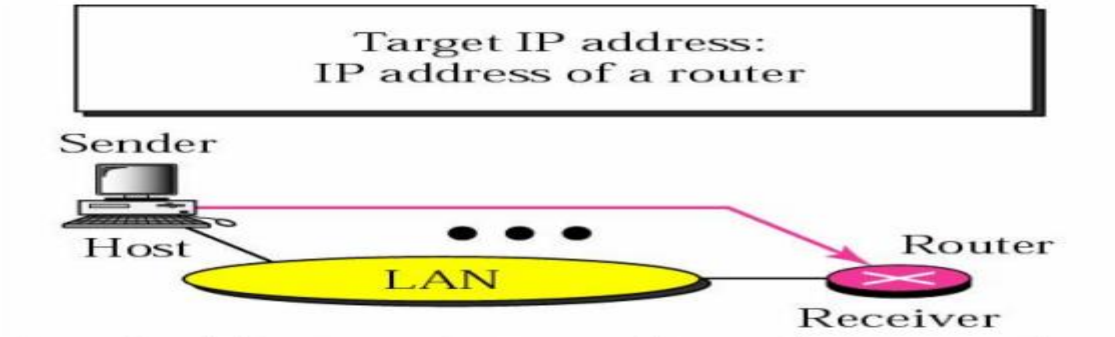
- Every host or routers receives the frame and since the destination address is broadcast, pass it to the ARP
- All machines ’ ARP except the one targeted drop the packet
- The target reply with an ARP reply message that contains its physical address and is unicast o
- The sender receives the reply message and knows the target’ s physical address

Four Cases to Use ARP

Case 1: The sender is a host and wants to send a packet to another host on the same network
Use ARP to find another host’ s physical address
Case 2: The sender is a host and wants to send a packet to another host on another network
Sender looks at its routing table
Find the IP address of the next hop (router) for this destination
Use ARP to find the router ’ s physical address



Case 1. A host has a packet to send to another host on the same network.



Case 2. A host wants to send a packet to another host on another network.
It must first be delivered to a router.

Case 3: the sender is a router and received a datagram destined for a host on another network

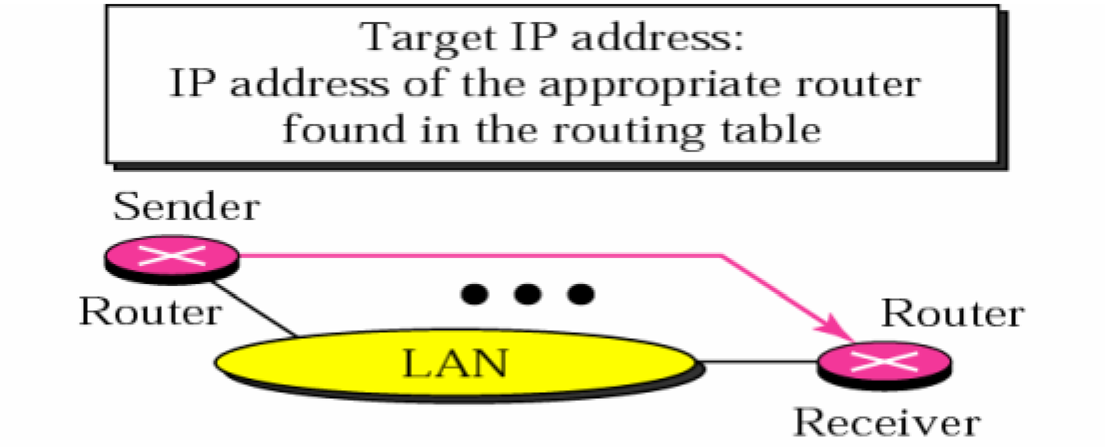
Router check its routing table

Find the IP address of the next router

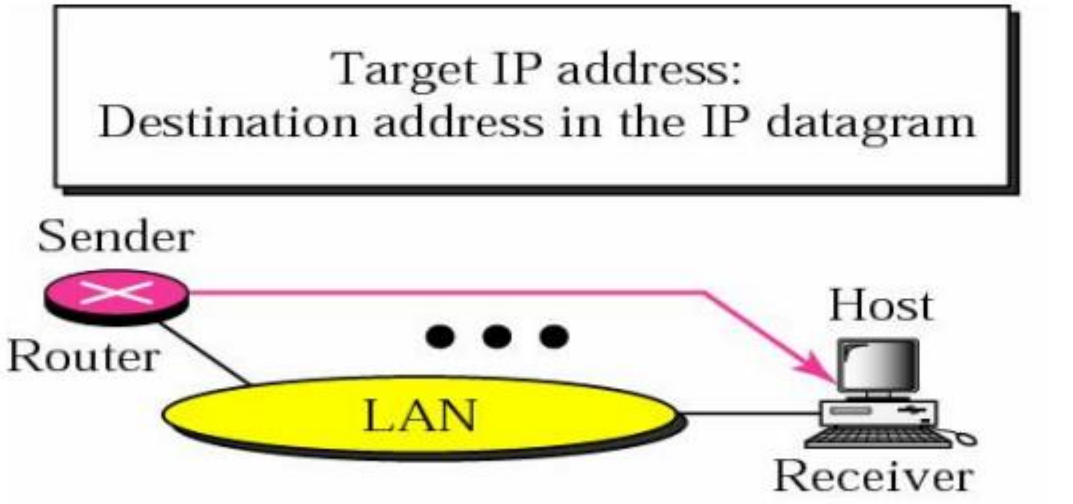
Use ARP to find the next router ' s physical address

Case 4: the sender is a router that has received a datagram destined for a host in the same network

Use ARP to find this host' s physical address


Target IP address:
IP address of the appropriate router
found in the routing table
Sender
Router
Router
LAN
Receiver

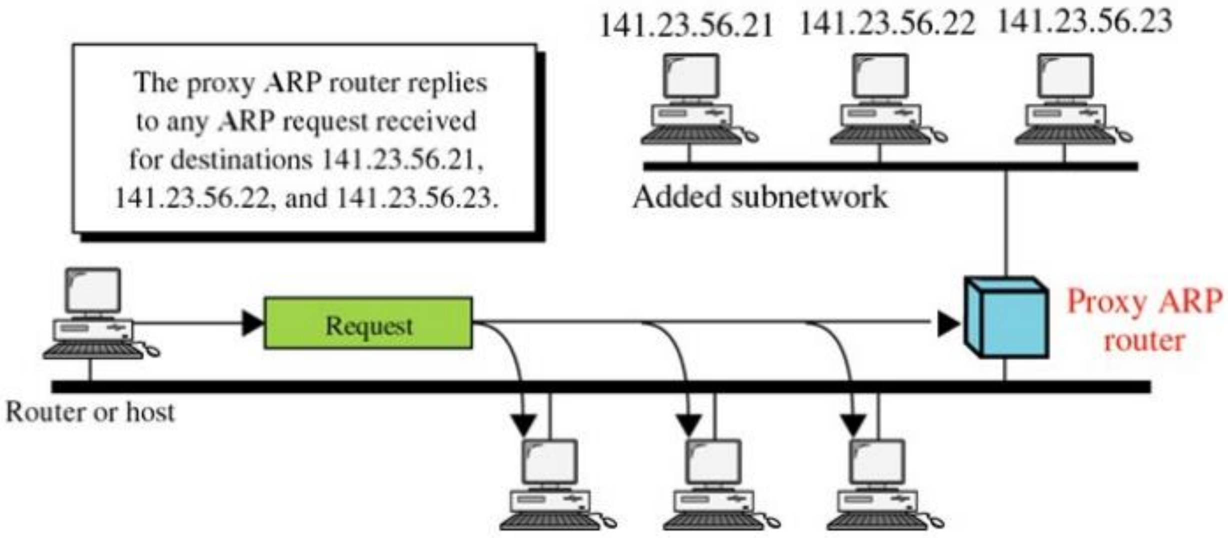
Case 3. A router receives a packet to be sent to a host on another network. It must first be delivered to the appropriate router.



Case 4. A router receives a packet to be sent to a host on the same network.

Proxy ARP

- A proxy ARP is an ARP that acts on be half of a set of host..
- Whenever the the router running a proxy ARP receives an ARP request looking for the IP address of one of these hosts, the router sends an ARP reply announcing its own hardware(physical) address.
- Later when the router receives the actual IP packet, it will send the packet to the appropriate host or router.


141.23.56.21
141.23.56.22
141.23.56.23
The proxy ARP router replies to any ARP request received for destinations 141.23.56.21, 141.23.56.22, and 141.23.56.23.
Added subnetwork
Request
Proxy ARP router
Router or host

ARP DESIGN

ARP package involves five component:

– A cache table

–A queues

–An output module

–An input module

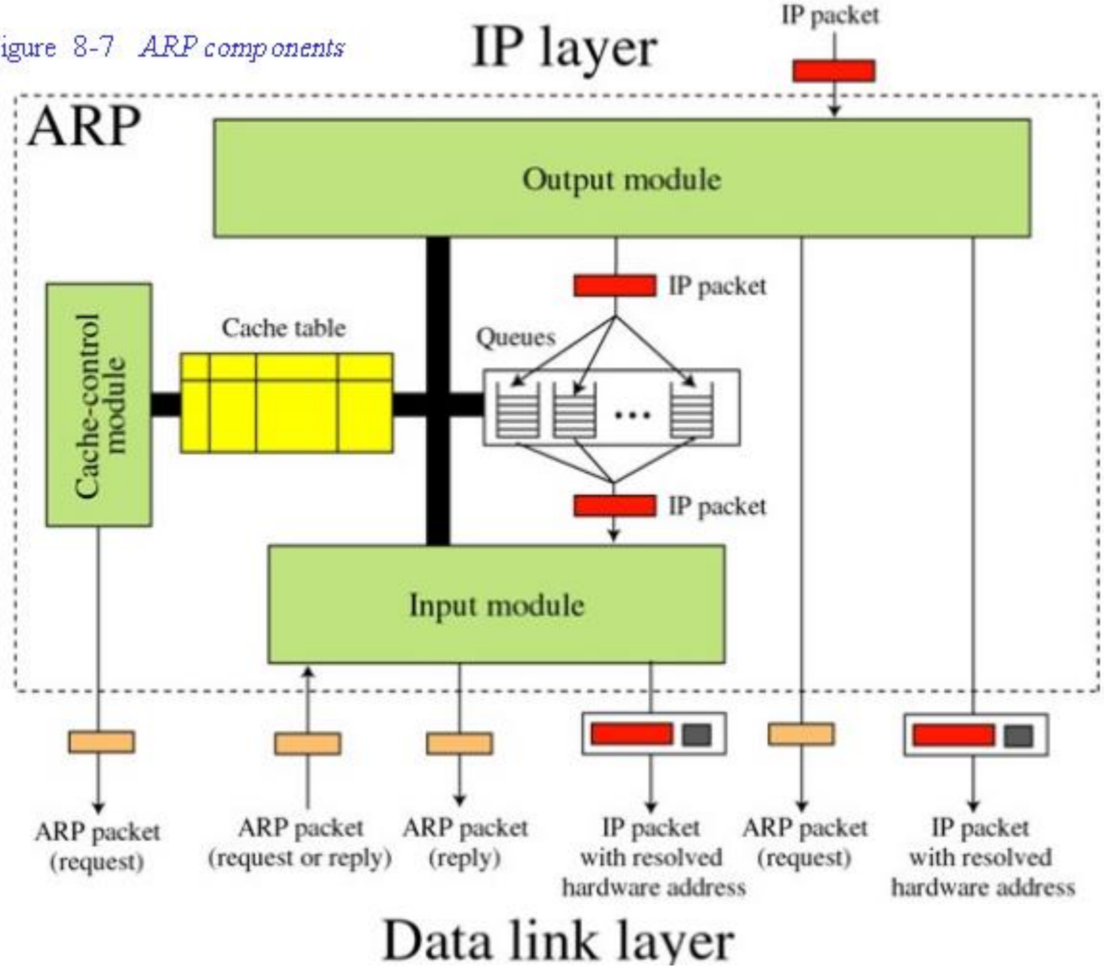
–A cache-control module

Cache Table

When a host or router receives the corresponding physical address for IP datagram, the address can be saved in the cache table.

This address can be used for the datagram destined for the same receiver within the next few minutes.

Figure 8-7 *ARP components*



Cache Table.. .

The cache table is implemented as an array of entries, In our design, each entry contains the following fields:

State: This column shows the state of the entry. It can have one of three values:

FREE: The time-to-live for this entry has expired. The space can be used for a new entry.
PENDING: A request for this entry has been sent, but the reply has not yet been received. –
RESOLVED: The entry is complete. The entry now has the physical address of the destination. The packets waiting to be sent this destination can use information in this entry.

Hardware type : This field is the same as corresponding field in the ARP packet.

Protocol type: This field is the same as corresponding field in the ARP packet.

Hardware length : This field is the same as corresponding field in the ARP packet.

Protocol length : This field is the same as corresponding field in the ARP packet.

Interface number: A router (or multihomed host) can be connected to different networks, each with a different interface number. Each networks can have different hardware and protocol types.

Queue number: ARP uses different queues to enqueue the packets waiting for address resolution. Packets for the same destination are usually enqueued in the same queue. The queue number refers to the queue whose packets are waiting for this entry to be resolved.

Attempts : This column shows how many times an ARP requesu is sent out for this entry.

Time-out : This column shows the lifetime of an entry in seconds.

Hardware address : This column shows the destination hardware address. It remains empty until resolved by an ARP reply.

Protocol address : This column shows the destination IP address.

Queues

The ARP package maintains a set of queues, one for each destination, to hold the IP packets while ARP tries to resolve the hardware address.

The output module sends unresolved packets into the corresponding queue.

The input module removes a packet from a queue and sends it, with the resolved physical address, to the data link layer for transmission.

OUTPUT MODULE

```
1.Sleep until an IP packet is received from IP software.
 2.Check the cache table to find an entry corresponding to this IP packet.
 3.If (found)
        1.If (the state is RESOLVED)
                1.Extract the value of the hardware address from th entry.
                2.Send the packet and the hardware address to data link layer.
                3.Return.
```

```
    2.If (the state is PENDING)
        1.Enqueue the packet to the corresponding queue.
        2.Return.
4.If (not found)
    1.Create a queue.
    2.Enqueue the packet.
    3.Create a cache entry with state set to PENDING and ATTEMPTS set to 1.
    4.Send an ARP request.
5.Return.
```

Input Module

```
1.Sleep until an ARP packet(request or reply) arrives.
2.Check the cache table to find an entry corresponding to this ARP packet.
3.If (found)
    1.If (the state is PENDING)
        1.Update the entry.
         2.While the queue is not empty.
            1.Dequeue one packet.
            2.Send the packet and the hardware address to data link.
    2.If (the state is RESOLVED)
        1.Update the entyry.
4.If (not found)
    1.Create an entry.
     2.Add the entry to the table.
 5.If (the packet is a request)
     1.Send an ARP reply
6.Return.
```

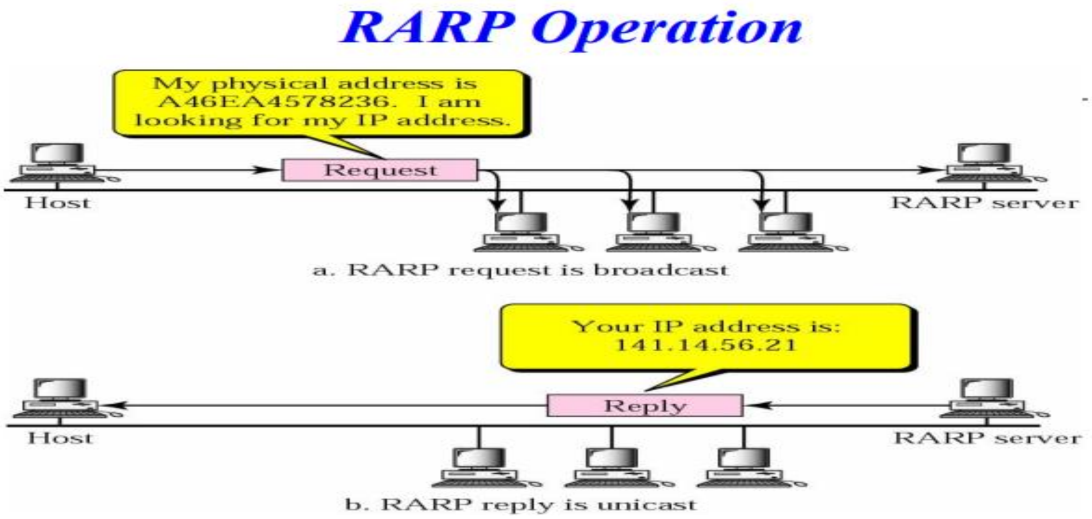
Cache-Control Module

```
1.Sleep until the periodic timer matures.
2.For every entry in the cache table.
     1.If (the state is FREE)
        1.Continue.
    2.If (the state is PENDING)
        1.Increment the value of attempts by 1.
        2.If (attempts greater than maxium)
            1.Change the state to FREE
            2.Destroy the corresponding queue.
     3.If (not) 1.Send an ARP request.
    4.Continue.
3.If (the state is RESOLVED)
```

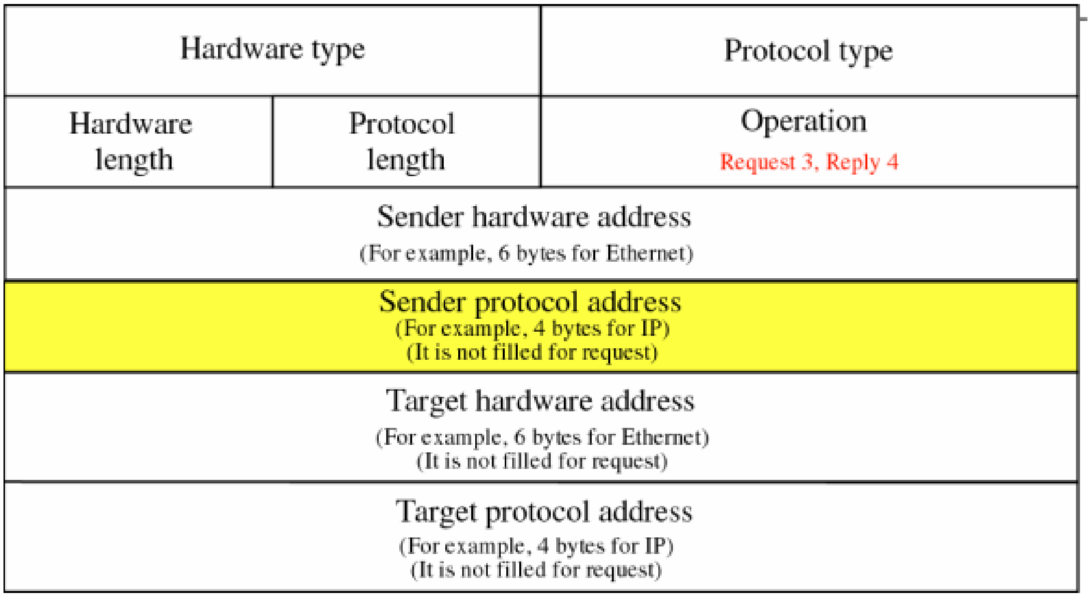
```
1.Decrement the value of time-out by the value of elapsed time.
2.If (time-out less than or equal to zero)
        1.Change the state to FREE
        2.Destroy the corresponding queue.
3.Return.
```

7.RARP

- The RARP is designed to resolve the address mapping problem in which a machine knows its physical address but does not know its logical address.
- To create an IP datagram, a host or a router needs to know its own IP address or address.
- The IP address of a machine is usually read from its configuration file stored on a disk file.
- However, a diskless machine is usually booted from ROM, which has minimum booting information.
- The machine can get its physical address (by reading its NIC, for example), which is unique locally. It can then use the physical address to get the logical address using the RARP protocol.
- A ARP request is created and broadcast on the local network.
- Another machine on the local network that knows all the IP address will respond with RARP reply.

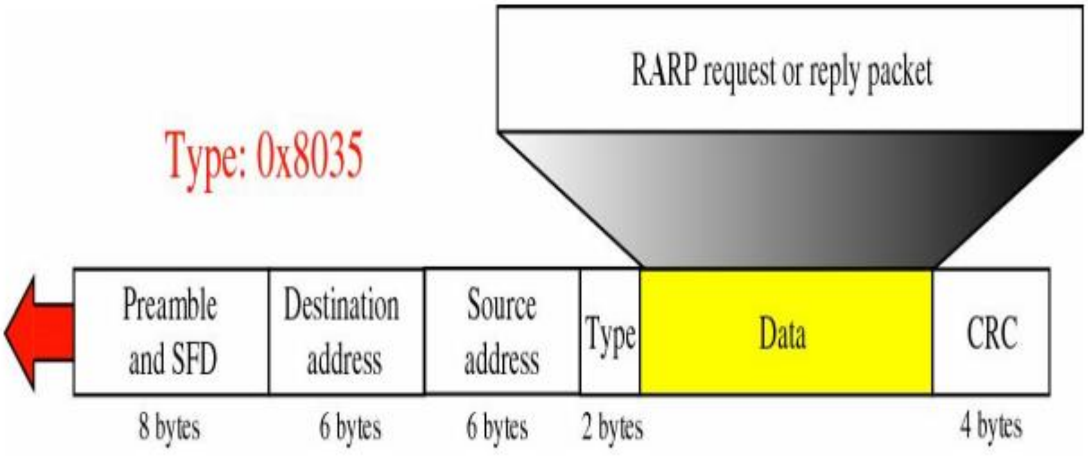

RARP Operation
My physical address is A46EA4578236. I am looking for my IP address.
Request
Host
RARP server
a. RARP request is broadcast
Your IP address is: 141.14.56.21
Reply
Host
RARP server
b. RARP reply is unicast

RARP Packet



Hardware type		Protocol type
Hardware length	Protocol length	Operation Request 3, Reply 4
Sender hardware address (For example, 6 bytes for Ethernet)		
Sender protocol address (For example, 4 bytes for IP) (It is not filled for request)		
Target hardware address (For example, 6 bytes for Ethernet) (It is not filled for request)		
Target protocol address (For example, 4 bytes for IP) (It is not filled for request)		

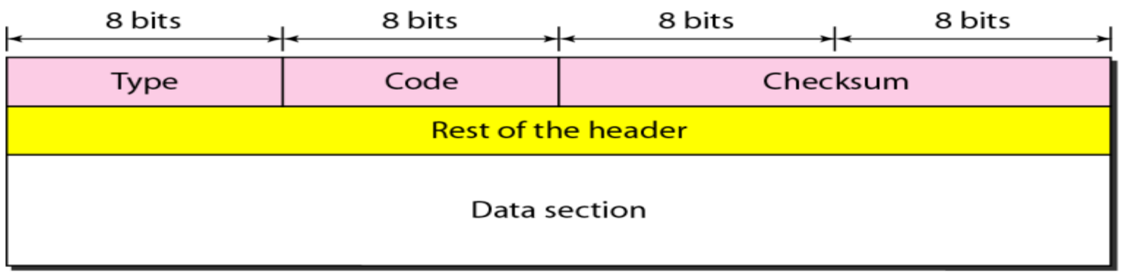
Encapsulation of RARP Packet



8.ICMP

The IP protocol has no error-reporting or error correcting mechanism. The IP protocol also lacks a mechanism for host and management queries. The Internet Control Message Protocol (ICMP) has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol.

General format of ICMP messages

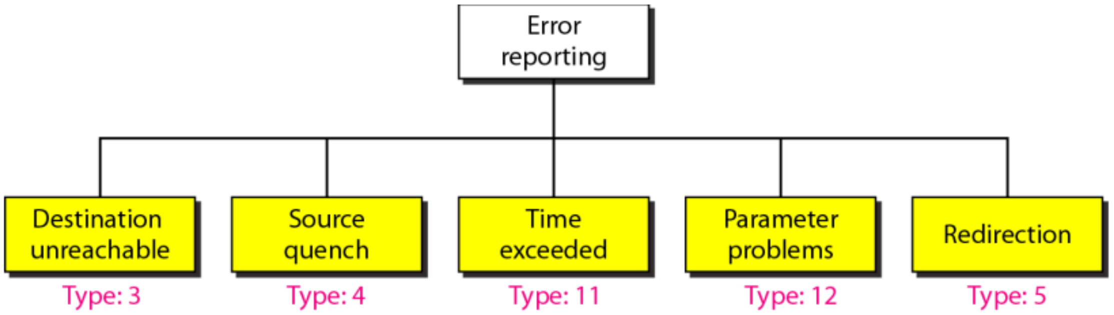


Types of Messages

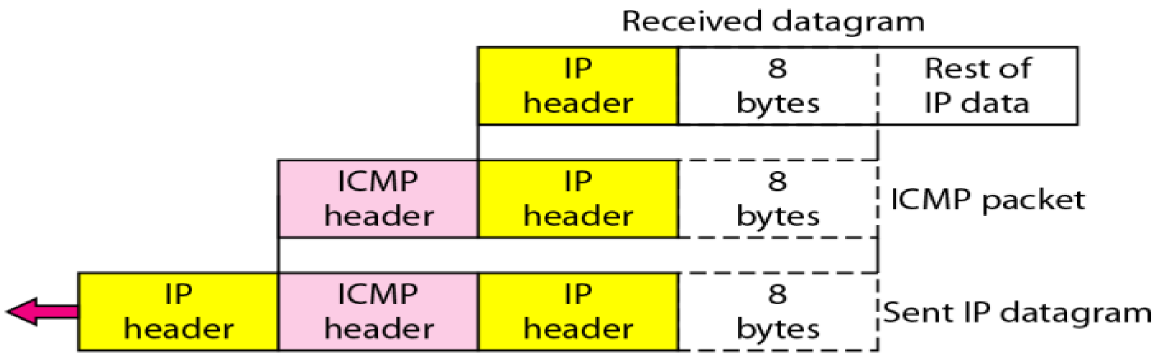
ICMP messages are divided into two broad categories:

Error-reporting messages and
Query messages

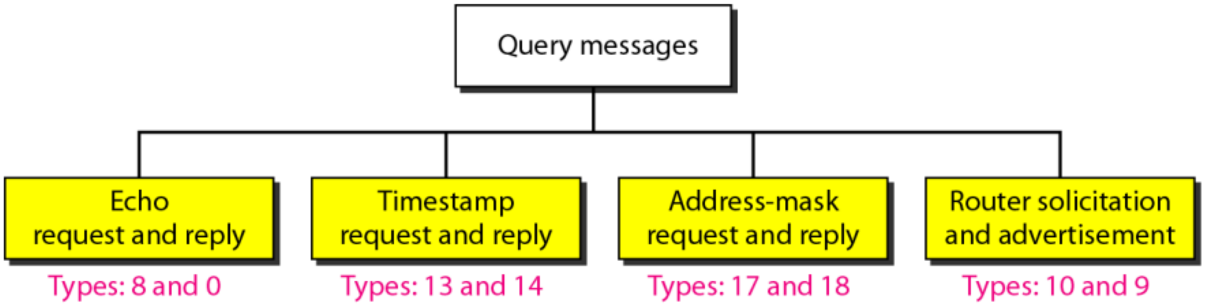
Error-reporting messages



Contents of data field for the error messages



Query messages



Echo Request and Reply: The echo-request and echo-reply messages are designed for diagnostic purposes

Timestamp Request and Reply Two machines (hosts or routers) can use the timestamp request and timestamp reply messages to determine the round-trip time needed for an IP datagram to travel between them. It can also be used to synchronize the clocks in two machines

Address-Mask Request and Reply

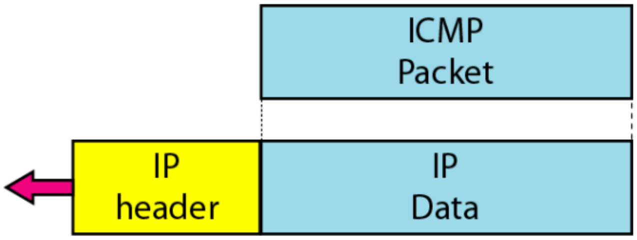
A host may know its IP address, but it may not know the corresponding mask.

Router Solicitation and Advertisement

A host that wants to send data to a host on another network needs to know the address of routers connected to its own network.

checksum In ICMP the checksum is calculated over the entire message (header and data).

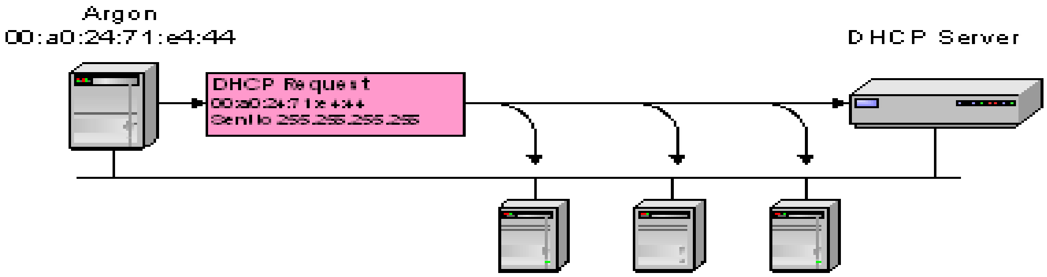
Encapsulation of ICMP query messages

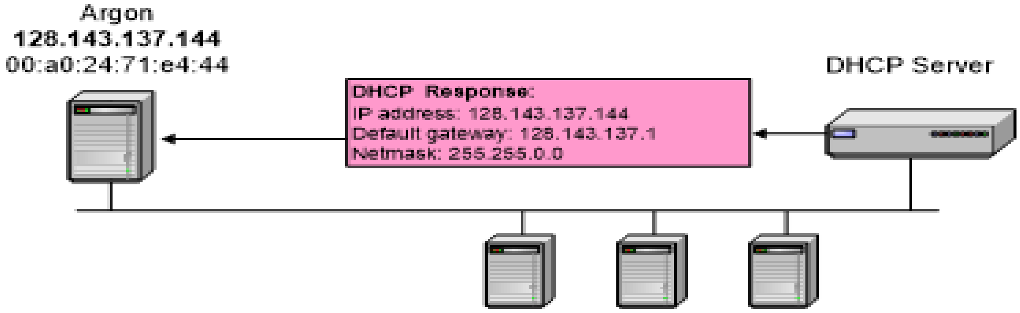


8.DHCP

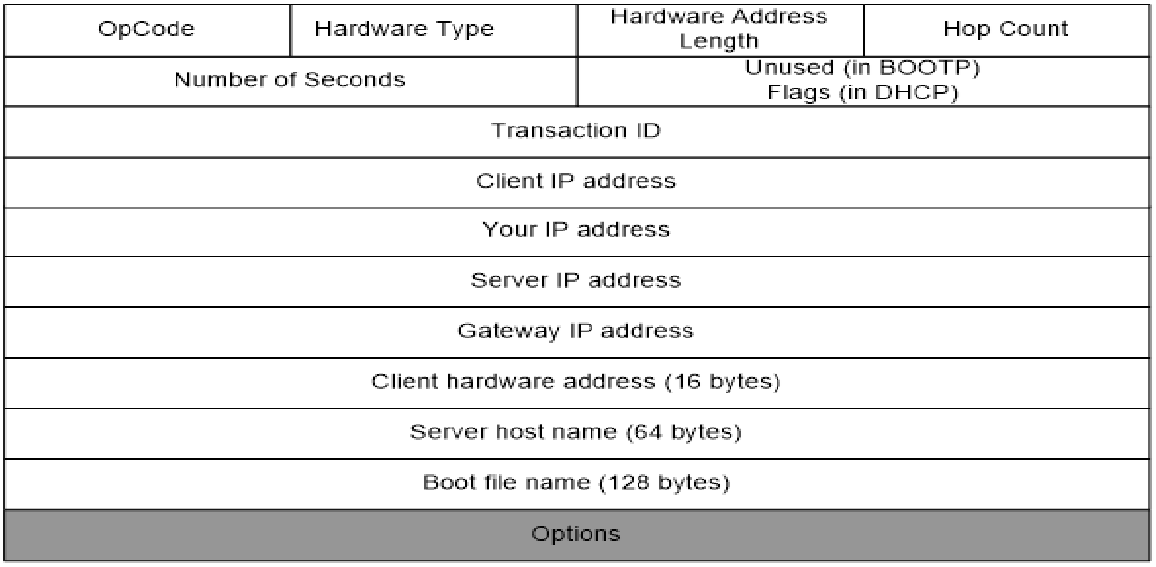
- Dynamic assignment of IP addresses is desirable for several reasons:
  - IP addresses are assigned on-demand
  - Avoid manual IP configuration
  - Support mobility of laptops
  - DHCP is the preferred mechanism for dynamic assignment of IP addresses
  - DHCP can interoperate with BOOTP clients.

DHCP Interaction (simplified)





DHCP Message Format

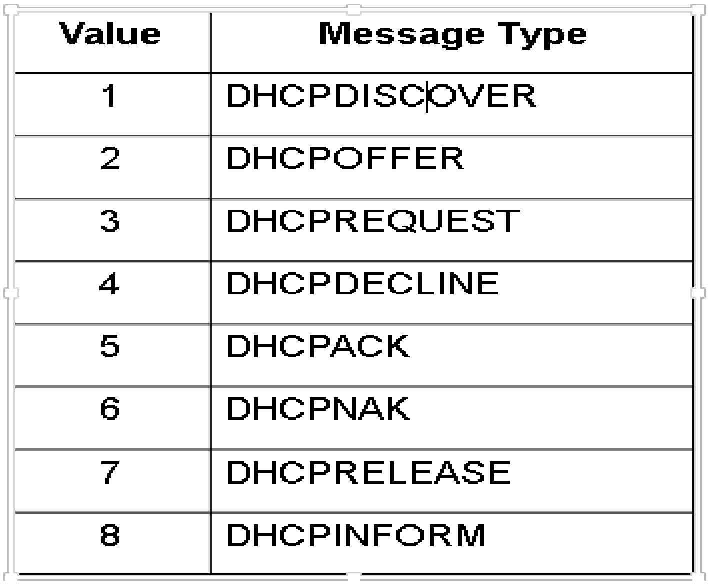


OpCode	Hardware Type	Hardware Address Length	Hop Count
Number of Seconds		Unused (in BOOTP) Flags (in DHCP)	
Transaction ID			
Client IP address			
Your IP address			
Server IP address			
Gateway IP address			
Client hardware address (16 bytes)			
Server host name (64 bytes)			
Boot file name (128 bytes)			
Options			

- **OpCode**: *1 (Request), 2(Reply)*   *Note: DHCP message type is sent in an option*

- **Hardware Type**: *1 (for Ethernet)*
- **Hardware address length**: *6 (for Ethernet)*
- **Hop count**: *set to 0 by client*
- **Transaction ID**: *Integer (used to match reply to response)*
- **Seconds:** *number of seconds since the client started to boot*
- **Client IP address, Your IP address, server IP address, Gateway IP address, client hardware address, server host name, boot file name:**
  client fills in the information that it has, leaves rest blank

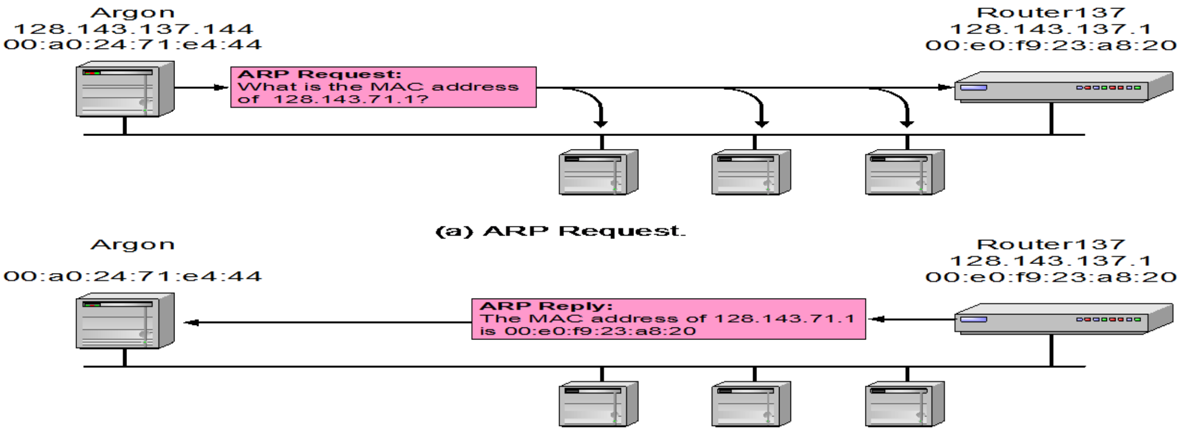
Message Type



Value	Message Type
1	DHCPDISCOVER
2	DHCPOFFER
3	DHCPREQUEST
4	DHCPDECLINE
5	DHCPACK
6	DHCPNAK
7	DHCPRELEASE
8	DHCPINFORM

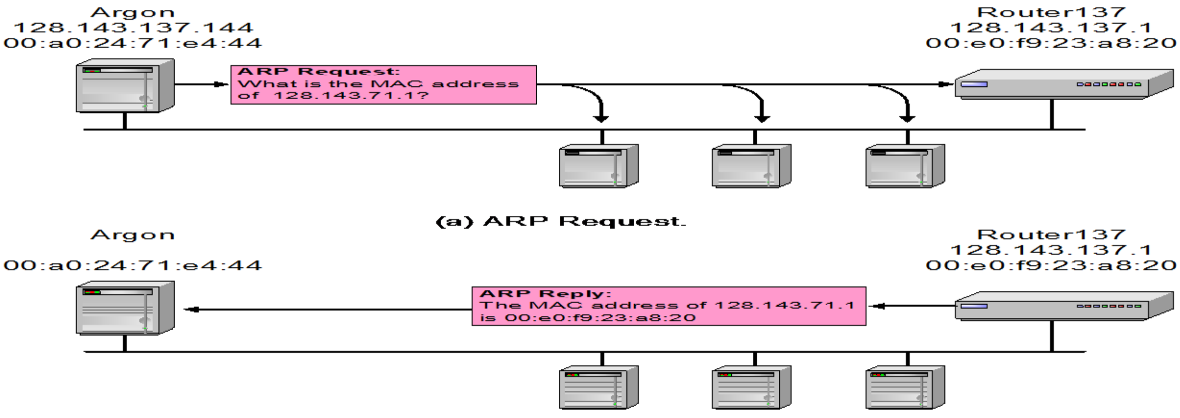
DHCP Operation

- DCHP DISCOVER
- DCHP OFFER
- DCHP RELEASE



(a) ARP Request.

(b) ARP Reply.

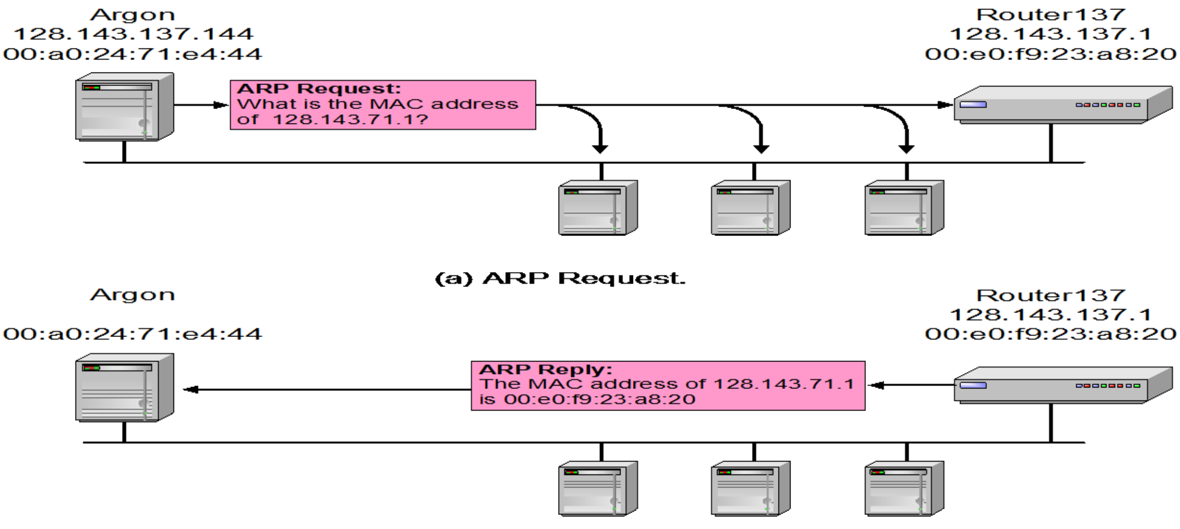


(a) ARP Request.

(b) ARP Reply.

- **DCHP DISCOVER**
- At this time, the DHCP client can start to use the IP address
- Renewing a Lease(sent when 50% of lease has expired)

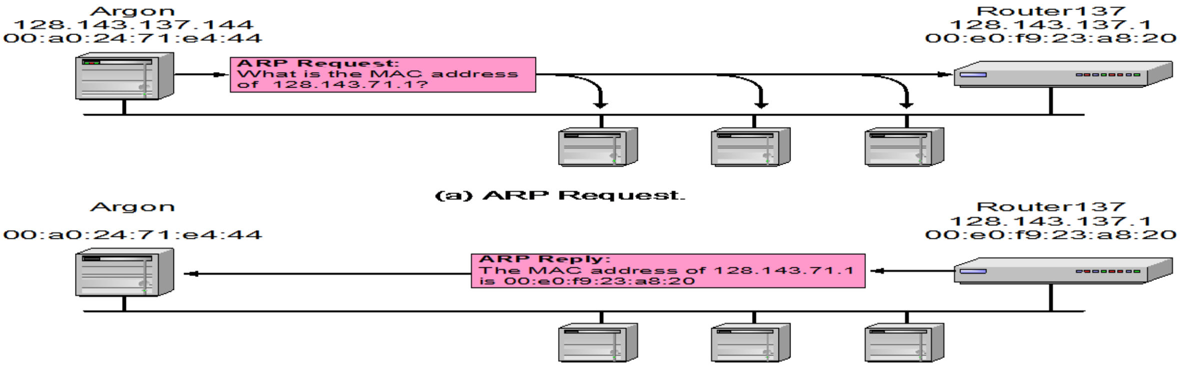
If DHCP server sends DHCPNACK, then address is released.



(a) ARP Request.

(b) ARP Reply.

- **DCHP RELEASE**
- At this time, the DHCP client has released the IP address



(a) ARP Request.

(b) ARP Reply.