Module 3:

Basics of Network, Transport and Application Layers

3.1 Network Layer

The network layer in the TCP/IP protocol suite is responsible for the host-to-host delivery of datagrams. It provides services to the transport layer and receives services from the data-link layer.

3.1.1 INTERNET PROTOCOL (IP)

The network layer in version 4 can be thought of as one main protocol and three auxiliary ones. The main protocol, Internet Protocol version 4 (IPv4), is responsible for packetizing, forwarding, and delivery of a packet at the network layer. The Internet Control Message Protocol version 4 (ICMPv4) helps IPv4 to handle some errors that may occur in the network-layer delivery. The Internet Group Management Protocol (IGMP) is used to help IPv4 in multicasting. The Address Resolution Protocol (ARP) is used to glue the network and data-link layers in mapping network-layer addresses to link-layer addresses. Figure 3.1 shows the positions of these four protocols in the TCP/IP protocol suite.

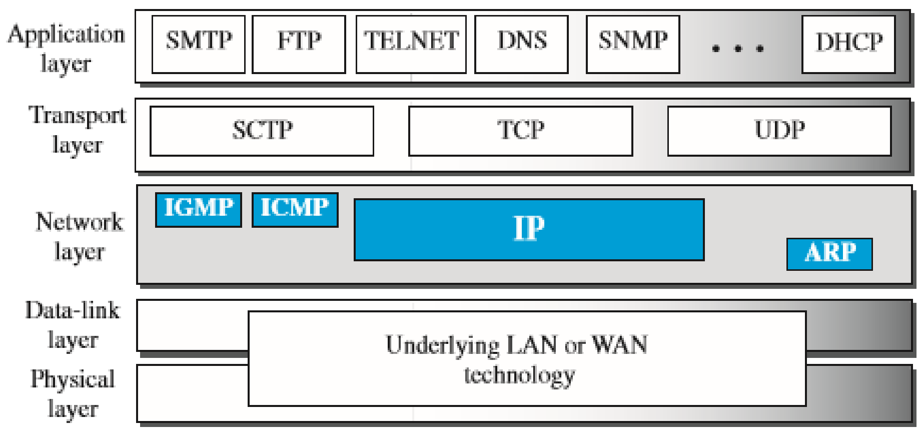


Figure 3.1: Position of IP and other network-layer protocols in TCP/IP protocol suite.

IPv4 is an unreliable datagram protocol—a best-effort delivery service. The term best-effort means that IPv4 packets can be corrupted, be lost, arrive out of order, or be delayed, and may create congestion for the network. If reliability is important, IPv4 must be paired with a reliable transport-layer protocol such as TCP. An example of a more commonly understood best-effort delivery service is the post office. The post office does its best to deliver the regular mail but does not always succeed. If an unregistered letter is lost or damaged, it is up to the sender or would-be recipient to discover this. The post office itself does not keep track of every letter and cannot notify a sender of loss or damage of one.

IPv4 is also a connectionless protocol that uses the datagram approach. This means that each datagram is handled independently, and each datagram can follow a different route to the destination. This implies that datagrams sent by the same source to the same destination could arrive out of order. Again, IPv4 relies on a higher-level protocol to take care of all these problems.

Datagram Format

In this section, we begin by discussing the first service provided by IPv4, packetizing. We show how IPv4 defines the format of a packet in which the data coming from the upper layer or other protocols are encapsulated. Packets used by the IP are called datagrams. Figure 3.2 shows the IPv4 datagram format. A datagram is a variable-length packet consisting of two parts: header and payload (data). The header is 20 to 60 bytes in length and contains information essential to routing and delivery. It is customary in TCP/IP to show the header in 4-byte sections.

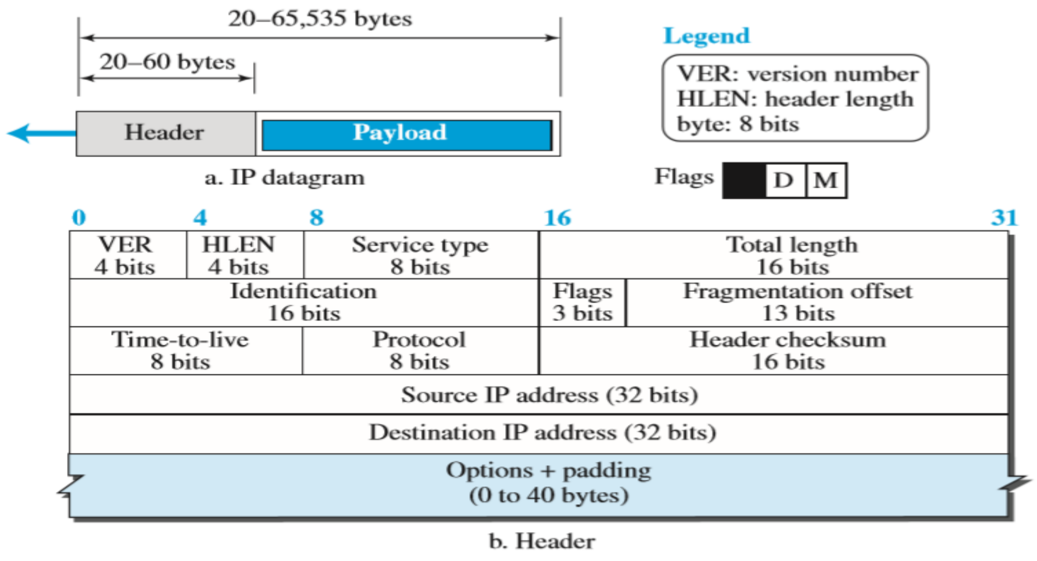


Figure 3.2: IP datagram.

Discussing the meaning and rationale for the existence of each field is essential to understanding the operation of IPv4; a brief description of each field is in order.

- **Version Number:** The 4-bit version number (VER) field defines the version of the IPv4 protocol, which, obviously, has the value of 4.
- **Header Length**: The 4-bit header length (HLEN) field defines the total length of the datagram header in 4-byte words. The IPv4 datagram has a variable-length header. When a device receives a datagram, it needs to know when the header stops and the data, which is encapsulated in the packet, starts. However, to make the value of the header length (number of bytes) fit in a 4-bit header length, the total length of the header is calculated as 4-byte words. The total length is divided by 4 and the value is inserted in the field. The receiver needs to multiply the value of this field by 4 to find the total length.
- **Service Type**: In the original design of the IP header, this field was referred to as type of service (TOS), which defined how the datagram should be handled. In the late 1990s, IETF redefined the field to provide differentiated services (DiffServ). The use of 4-byte words for the length header is also logical because the IP header always needs to be aligned in 4-byte boundaries.
- **Total Length**: This 16-bit field defines the total length (header plus data) of the IP datagram in bytes. A 16-bit number can define a total length of up to 65,535 (when all bits are 1s). However, the size of the datagram is normally much less than this. This field helps the receiving device to know when the packet has completely arrived. To find the length of the data coming from the upper layer, subtract the header length from the total length. The header length can be found by multiplying the value in the HLEN field by 4.

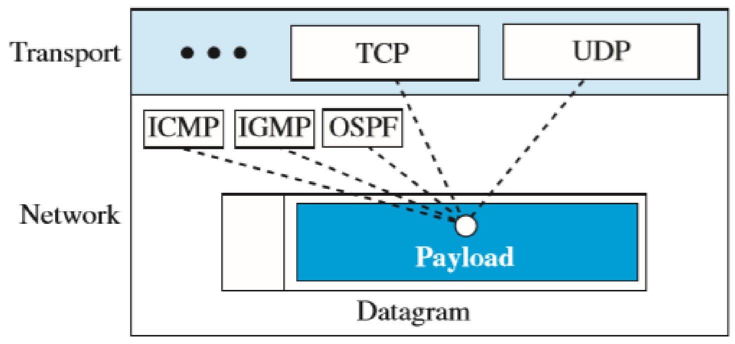
  $$\text{Length of data} = \text{total length} - (\text{HLEN}) \times 4$$

  Though a size of 65,535 bytes might seem large, the size of the IPv4 datagram may increase in the near future as the underlying technologies allow even more throughput (greater bandwidth). One may ask why we need this field anyway. When a machine (router or host) receives a frame, it drops the header and the trailer, leaving the datagram. Why include an extra field that is not needed? The answer is that in many cases we really do not need the value in this field. However, there are occasions in which the datagram is not the only thing encapsulated in a frame; it may be that padding has been added. For example, the Ethernet protocol has a minimum and maximum restriction on the size of data that can be encapsulated in a frame (46 to 1500 bytes). If the size of an IPv4 datagram is less than 46 bytes, some padding will be added to meet this requirement. In this case, when a machine decapsulates the datagram, it needs to check the total length field to determine how much is really data and how much is padding.

- **Identification, Flags, and Fragmentation Offset**: These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry.

- **Time-to-live**: Due to some malfunctioning of routing protocols (discussed later) a datagram may be circulating in the Internet, visiting some networks over and over without reaching the destination. This may create extra traffic in the Internet. The time-to-live (TTL) field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately two times the maximum number of routers between any two hosts. Each router that processes the datagram decrements this number by one. If this value, after being decremented, is zero, the router discards the datagram.

- **Protocol**: In TCP/IP, the data section of a packet, called the payload, carries the whole packet from another protocol. A datagram, for example, can carry a packet belonging to any transport-layer protocol such as UDP or TCP. A datagram can also carry a packet from other protocols that directly use the service of the IP, such as some routing protocols or some auxiliary protocols. The Internet authority has given any protocol that uses the service of IP a unique 8-bit number which is inserted in the protocol field. When the payload is encapsulated in a datagram at the source IP, the corresponding protocol number is inserted in this field; when the datagram arrives at the destination, the value of this field helps to define to which protocol the payload should be delivered. In other words, this field provides multiplexing at the source and demultiplexing at the destination, as shown in Figure 3.3. The protocol fields at the network layer play the same role as the port numbers at the transport layer. However, we need two port numbers in a transport-layer packet because the port numbers at the source and destination are different, but we need only one protocol field because this value is the same for each protocol no matter whether it is located at the source or the destination.



Some protocol values

ICMP	01
IGMP	02
TCP	06
UDP	17
OSPF	89

Figure 3.3 Multiplexing and demultiplexing using the value of the protocol field

- **Header checksum**: IP is not a reliable protocol; it does not check whether the payload carried by a datagram is corrupted during the transmission. IP puts the burden of error checking of the payload on the protocol that owns the payload, such as UDP or TCP. The datagram header, however, is added by IP, and its error-checking is the responsibility of IP. Errors in the IP header can be a disaster. For example, if the destination IP address is corrupted, the packet can be delivered to the wrong host. If the protocol field is corrupted, the payload may be delivered to the wrong protocol. If the fields related to the fragmentation are corrupted, the datagram cannot be reassembled correctly at the destination, and so on. For these reasons, IP adds a header checksum field to check the header, but not the payload. We need to remember that, since the value of some fields, such as TTL, which are related to fragmentation and options, may change from router to router, the checksum needs to be recalculated at each router. Checksum in the Internet normally uses a 16-bit field, which is the complement of the sum of other fields calculated using 1s complement arithmetic.

- **Source and Destination Addresses**: These 32-bit source and destination address fields define the IP address of the source and destination respectively. The source host should know its IP address. The destination IP address is either known by the protocol that uses the service of IP or is provided by the DNS. Note that the value of these fields must remain unchanged during the time the IP datagram travels from the source host to the destination host.
- **Options**: A datagram header can have up to 40 bytes of options. Options can be used for network testing and debugging. Although options are not a required part of the IP header, option processing is required of the IP software. This means that all implementations must be able to handle options if they are present in the header. The

existence of options in a header creates some burden on the datagram handling; some options can be changed by routers, which forces each router to recalculate the header checksum. There are one-byte and multi-byte options that we will briefly discuss later in the chapter. The complete discussion is posted at the book website.

- **Payload**: Payload, or data, is the main reason for creating a datagram. Payload is the packet coming from other protocols that use the service of IP. Comparing a datagram to a postal package, payload is the content of the package; the header is only the information written on the package.

3.1.2 IP standards

IP is a layer-3 protocol and supports variety of layer 2 protocols (IEEE 802.2, Ethernet-DIX, PPP, HDLC, ATM, frame relay). It provides service to layer 4 protocols which include TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

- TCP and Internet Protocol (IP) are two of the best known members of the Internet protocol suite, a suite of network communication protocols developed by Stanford University to facilitate heterogeneous connectivity.
- The Internet protocol suite provides the protocols that enable all of the networks of the Internet to communicate. Every computer on the Internet supports TCP/IP.
- The protocols in the TCP/IP suite are layered protocols. Layered protocol suites separate duties between individual protocols in the group. IP is the heart of the Internet protocol suite.
- IP addresses are globally unique, 32-bit numbers assigned by the Network Information Center. Globally unique addresses permit IP networks anywhere in the world to communicate with each other. Internet routers use these IP addresses to deliver IP packets to their destination.
- TCP is a connection-oriented transport protocol that sends data as an unstructured stream of bytes. TCP provides two services that IP is missing: guaranteed delivery and serialization of data.

3.1.3 Versions

There are two versions of Internet Protocol:

- IP version 4 (IPv4)
- IP version 6 (IPv6)

IPv4: Internet Protocol Version 4 (IPv4) is the fourth revision of the Internet Protocol (IP) used to identify devices on a network through an addressing system. The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks

IPv6: IPv6 is the successor to Internet Protocol Version 4 (IPv4). It was designed as an evolutionary upgrade to the Internet Protocol and will, in fact, coexist with the older IPv4 for some time. IPv6 is designed to allow the Internet to grow steadily, both in terms of the number of hosts connected and the total amount of data traffic transmitted.

3.1.4 IP functions

IP as having four basic functions:

- Addressing: In order to perform the job of delivering datagrams, IP must know where to deliver them to! For this reason, IP includes a mechanism for host addressing. Furthermore, since IP operates over internetworks, its system is designed to allow unique addressing of devices across arbitrarily large networks. It also contains a structure to facilitate the routing of datagrams to distant networks if that is required.
- Data Encapsulation and Formatting/Packaging: As the TCP/IP network layer protocol, IP accepts data from the transport layer protocols UDP and TCP. It then encapsulates this data into an IP datagram using a special format prior to transmission.
- Fragmentation and Reassembly: IP datagrams are passed down to the data link layer for transmission on the local network. However, the maximum frame size of each physical/data-link network using IP may be different. For this reason, IP includes the ability to fragment IP datagrams into pieces so they can each be carried on the local network. The receiving device uses the reassembly function to recreate the whole IP datagram again.
- Routing / Indirect Delivery: When an IP datagram must be sent to a destination on the same local network, this can be done easily using the network's underlying LAN/WLAN/WAN protocol using what is sometimes called direct delivery. However, in many (if not most cases) the final destination is on a distant network not directly attached to the source. In this situation the datagram must be delivered indirectly. This is accomplished by routing the datagram through intermediate devices (called routers). IP accomplishes this in concert with support from the other protocols including ICMP and the TCP/IP gateway/routing protocols such as RIP and BGP.

3.1.5 IPv4 addressing, IPv4 address Classes and IPv4 address types

3.1.5.1 IPV4 ADDRESSES

The identifier used in the IP layer of the TCP/IP protocol suite to identify the connection of each device to the Internet is called the Internet address or IP address. 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 the router, because if the device is moved to another network, the IP address may be changed. IPv4 addresses are unique in the sense that each address defines one, and only one, connection to the Internet. If a device has two connections to the Internet, via two networks, it has two IPv4 addresses. IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

Address Space

A protocol like IPv4 that defines addresses has an address space. An address space is the total number of addresses used by the protocol. If a protocol uses b bits to define an address, the address space is 2b 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). If there were no restrictions, more than 4 billion devices could be connected to the Internet.

- **Notation:** There are three common notations to show an IPv4 address: 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 dotted-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 8 hexadecimal digits. This notation is often used in network programming. Figure 3.4 shows an IP address in the three discussed notations.

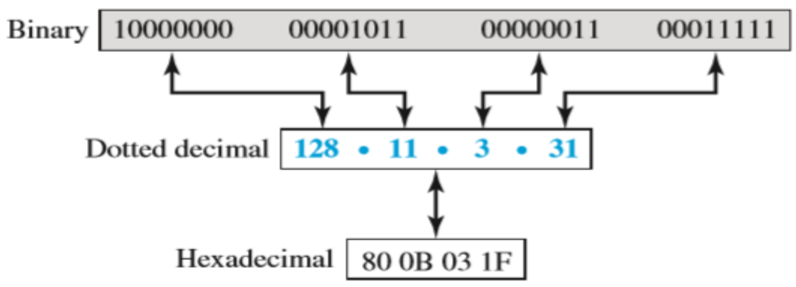


Figure 3.4: Three different notations in IPv4 addressing.

- **Hierarchy in Addressing**: In any communication network that involves delivery, such as a telephone network or a postal network, the addressing system is hierarchical. In a postal network, the postal address (mailing address) includes the country, state, city, street, house number, and the name of the mail recipient. Similarly, a telephone number is divided into the country code, area code, local exchange, and the connection.

  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). Figure 3.5 shows the prefix and suffix of a 32-bit IPv4 address. The prefix length is n bits and the suffix length is (32 − n) bits.

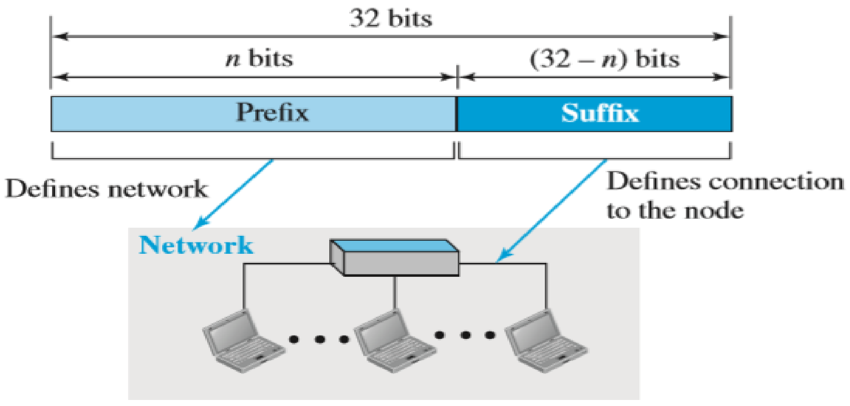


Figure 3.5: Hierarchy in addressing.

A prefix can be fixed length or variable length. The network identifier in the IPv4 was first designed as a fixed-length prefix. This scheme, which is now obsolete, is referred to as classful addressing. The new scheme, which is referred to as classless addressing, uses a variable-length network prefix.

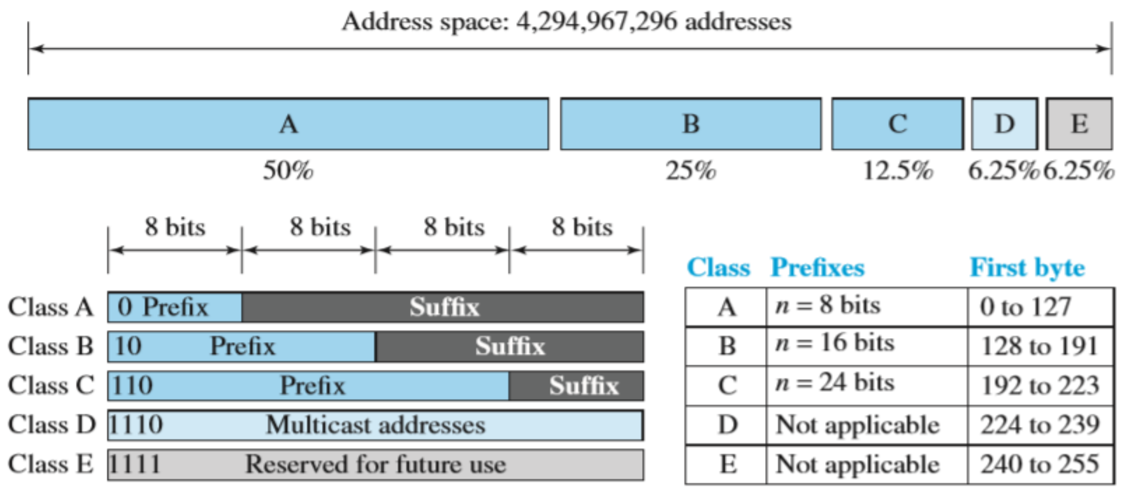
Classful Addressing

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 3.6. This scheme is referred to as classful addressing. Although classful addressing belongs to the past, it helps us to understand classless addressing, discussed later.

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 27 = 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 214 = 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 221 = 2,097,152 networks in the world that can have a class C address.



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

Figure 3.6: Occupation of the address space in classful addressing.

Class D is not divided into prefix and suffix. 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.

- **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.
- **Subnetting and Supernetting**: 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 $n_{sub} = 10$. 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.
- **Advantage of Classful Addressing**: Although classful addressing had several problems and became obsolete, it had one advantage: Given an address, we can easily find the class of the address and, since the prefix length for each class is fixed, we can find the prefix length immediately. In other words, the prefix length in classful addressing is inherent in the address; no extra information is needed to extract the prefix and the suffix.

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. Although the long-range solution has already been devised and is called IPv6 (discussed later), a short-term solution was also devised to use the same address space but to change the distribution of addresses to provide a fair share to each organization. The short-term solution still uses IPv4 addresses, but it is called classless addressing. In other words, the class privilege was removed from the distribution to compensate for the address depletion.

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 20, 21, 22, ...,232 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. Figure 3.7 shows the division of the whole address space into nonoverlapping blocks.

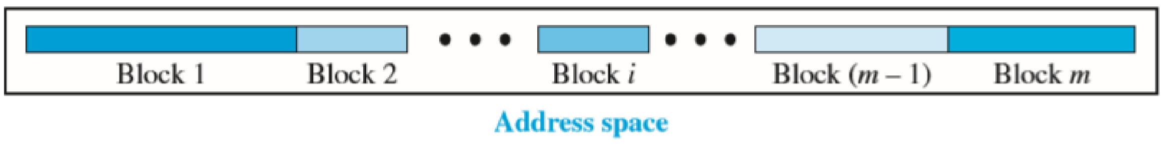


Figure 3.7: Variable-length blocks in classless addressing.

Unlike classful addressing, the prefix length in classless addressing is variable. We can have a prefix length that ranges from 0 to 32. The size of the network is inversely proportional to the length of the prefix. A small prefix means a larger network; a large prefix means a smaller network. We need to emphasize that the idea of classless addressing can be easily applied to classful addressing. An address in class A can be thought of as a classless address in which the prefix length is 8. An address in class B can be thought of as a classless address in which the prefix is 16, and so on. In other words, classful addressing is a special case of classless addressing.

- **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. An address in classless addressing can then be represented as shown in Figure 3.8.

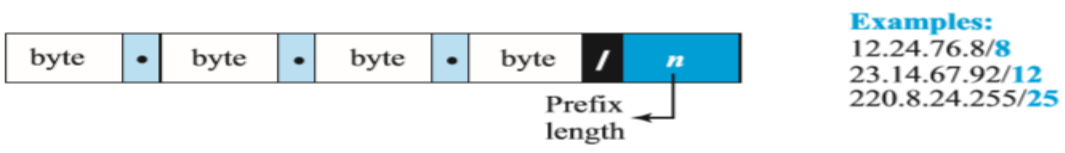


Figure 3.8: Slash notation (CIDR)

In other words, an address in classless addressing does not, per se, define the block or network to which the address belongs; we need to give the prefix length also.

- **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 of prefix length, n, is given, we can easily find these three pieces of information, as shown in Figure 3.9.

  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.

  Problem 1: 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.

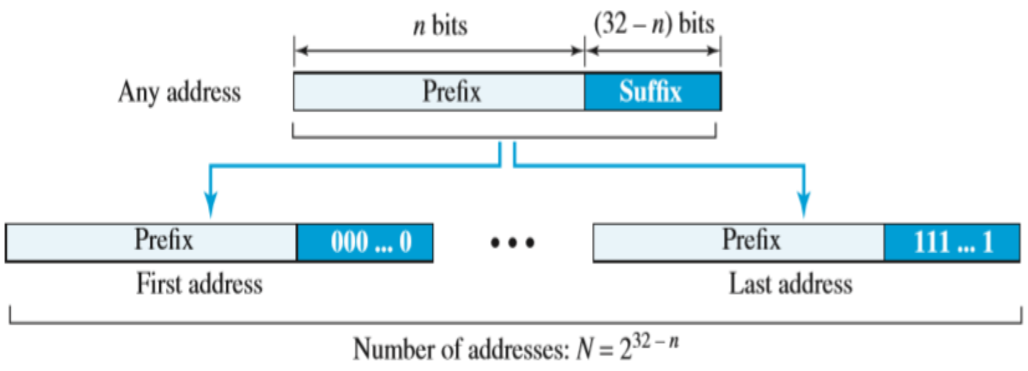


Figure 3.9: Information extraction in classless addressing

The first address can be found by keeping the first 27 bits and changing the rest of the bits to 0s.

```
Address: 167.199.170.82/27          10100111  11000111  10101010  01010010
First address: 167.199.170.64/27    10100111  11000111  10101010  01000000
```

The last address can be found by keeping the first 27 bits and changing the rest of the bits to 1s.

```
Address: 167.199.170.82/27          10100111  11000111  10101010  01011111
Last address: 167.199.170.95/27     10100111  11000111  10101010  01011111
```

- **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)].

  Problem 2: We repeat problem 1 using the mask. The mask in dotted-decimal notation is 256.256.256.224. The AND, OR, and NOT operations can be applied to individual bytes using calculators and applets at the book website.

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

- **Network Address**: The first address, the network address, is particularly important because it is used in routing a packet to its destination network. For the moment, let us assume that an internet is made of m networks and a router with m interfaces. When a packet arrives at the router from any source host, the router needs to know to which network the packet should be sent: from which interface the packet should be sent out. When the packet arrives at the network, it reaches its destination host using another strategy that we discuss later. After the network address has been found, the router consults its forwarding table to find the corresponding interface from which the packet should be sent out. The network address is actually the identifier of the network; each network is identified by its network address.

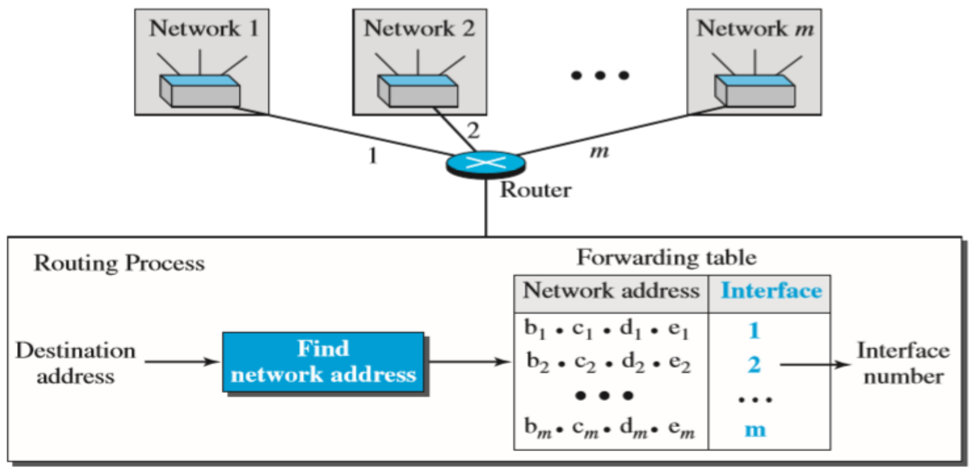


Figure 3.10: Network address.

- **Block Allocation**: The next issue in classless addressing is block allocation. How are the blocks allocated? The ultimate responsibility of block allocation is given to a global authority called the Internet Corporation for Assigned Names and Numbers (ICANN). However, ICANN does not normally allocate addresses to individual Internet users. It assigns a large block of addresses to an ISP (or a larger organization that is considered an ISP in this case). For the proper operation of the CIDR, two restrictions need to be applied to the allocated block.

  1. The number of requested addresses, N, needs to be a power of 2. The reason is that N = 232 − n or n = 32 − log2N. If N is not a power of 2, we cannot have an integer value for n.

  2. The requested block needs to be allocated where there is an adequate number of contiguous addresses available in the address space. However, there is a restriction on choosing the first address in the block. The first address needs to be divisible by the number of addresses in the block. The reason is that the first address needs to be the prefix followed by (32 − n) number of 0s. The decimal value of the first address is then

  first address = (prefix in decimal) × 232 − n = (prefix in decimal) × N.

- **Subnetting**: More levels of hierarchy can be created using subnetting. An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet). Note that nothing stops the organization from creating more levels. A subnetwork can be divided into several sub-subnetworks. A sub-subnetwork can be divided into several sub-sub-subnetworks, and so on.

Designing Subnets: The subnetworks in a network should be carefully designed to enable the routing of packets. We assume the total number of addresses granted to the organization is N, the prefix length is n, the assigned number of addresses to each subnetwork is N_{sub} and the prefix length for each subnetwork is n_{sub}. Then the following steps need to be carefully followed to guarantee the proper operation of the subnetworks.

- The number of addresses in each subnetwork should be a power of 2.
- The prefix length for each subnetwork should be found using the following formula:

  $$n_{sub} = 32 - \log_2 N_{sub}$$

- The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we first assign addresses to larger subnetworks.

Finding Information about Each Subnetwork: After designing the subnetworks, the information about each subnetwork, such as first and last address, can be found using the process we described to find the information about each network in the Internet.

Problem 3: An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets: one subblock of 10 addresses, one subblock of 60 addresses, and one subblock of 120 addresses. Design the subblocks.

Solution: There are $2^{32-24} = 256$ addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one. a. The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2. We allocate 128 addresses. The subnet mask for this subnet can be found as $n_1 = 32 - \log_2 128 = 25$. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25. b. The number of addresses in the second largest subblock, which requires 60 addresses, is not a power of 2 either. We allocate 64 addresses. The subnet mask for this subnet can be found as $n_2 = 32 - \log_2 64 = 26$. The first address in this block is 14.24.74.128/26; the last address is

14.24.74.191/26. c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2 either. We allocate 16 addresses. The subnet mask for this subnet can be found as $n_3 = 32 - \log_2 16 = 28$. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28. If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet. Figure 3.11 shows the configuration of blocks. We have shown the first address in each block.

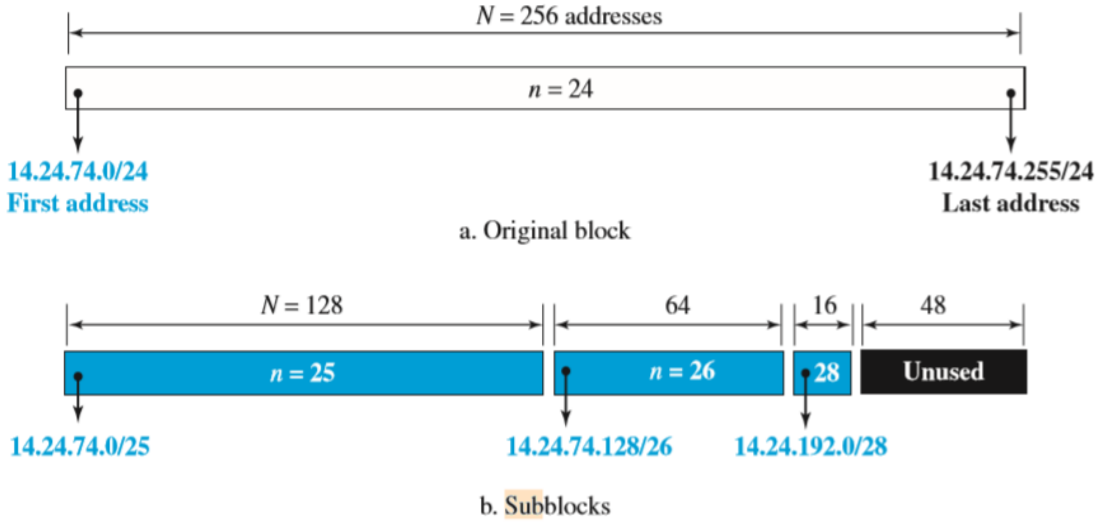


Figure 3.11: Solution to problem 3.

- **Address Aggregation**: One of the advantages of the CIDR strategy is address aggregation (sometimes called address summarization or route summarization). When blocks of addresses are combined to create a larger block, routing can be done based on the prefix of the larger block. ICANN assigns a large block of addresses to an ISP. Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers.

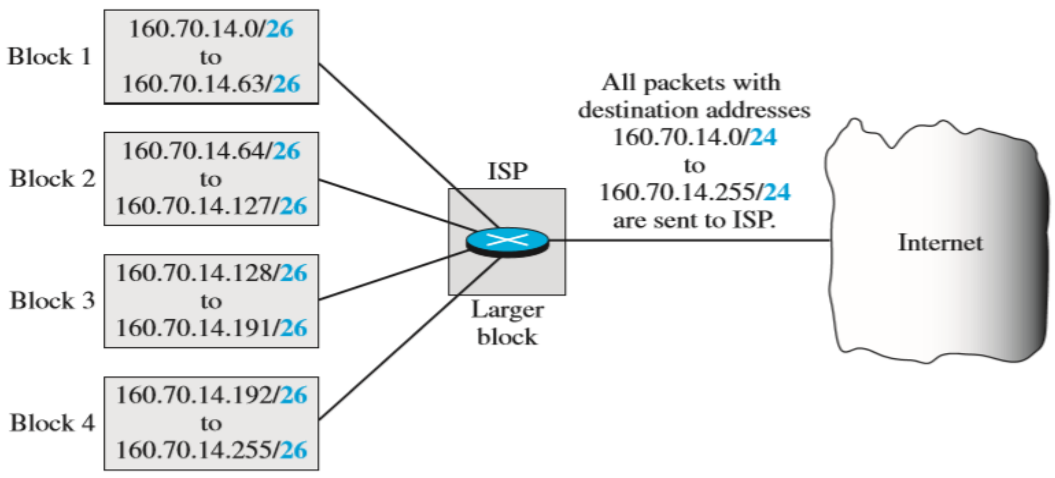


Figure 3.12: Example of address aggregation.

3.1.6 Default Gateway

A default gateway allows computers on a network to communicate with computers on another network. Without it, the network is isolated from the outside. Basically, computers send data that is bound for other networks (one that does not belong to its local IP range) through the default gateway. Network administrators configure the computer's routing capability with an IP range's starting address as the default gateway and point all clients to that IP address.

3.1.7 Public & Private IP Address

All IPv4 IP addresses can be divided into two major groups: global, or public, or external - this group can also be called 'WAN addresses' — those that are used in the Internet, and private, or local, or internal addresses — those that are used in the local network (LAN).

There are also special-use addresses, intended for technical purposes such as protocol functions etc. Normally these are not exposed to a user at all.

Public IP-address

It is public global addresses that are used in the Internet. A public IP address is an IP address that is used to access the Internet. Public (global) IP addresses are routed on the Internet, unlike private addresses.

The presence of a public IP address on our router or computer will allow us to organize our own server (VPN, FTP, WEB, etc.), remote access to our computer, video surveillance cameras and access them from anywhere in the global network.

With a public IP address, we can set up any home server to publish it on the Internet: Web (HTTP), VPN (PPTP/IPSec/OpenVPN), media (audio/video), FTP, NAS network drive, game server, etc.

Private IP-address

Private internal addresses are not routed on the Internet and no traffic cannot be sent to them from the Internet, they only supposed to work within the local network. Private addresses include IP addresses from the following subnets:

- Range from 10.0.0.0 to 10.255.255.255 — a 10.0.0.0 network with a 255.0.0.0 or an /8 (8-bit) mask
- Range from 172.16.0.0 to 172.31.255.255 — a 172.16.0.0 network with a 255.240.0.0 (or a 12-bit) mask
- A 192.168.0.0 to 192.168.255.255 range, which is a 192.168.0.0 network masked by 255.255.0.0 or /16
- A special range 100.64.0.0 to 100.127.255.255 with a 255.192.0.0 or /10 network mask; this subnet is recommended according to rfc6598 for use as an address pool for CGN (Carrier-Grade NAT)

Those are reserved IP addresses. These addresses are intended for use in closed local area networks and the allocation of such addresses is not globally controlled by anyone. Direct access to the Internet using a private IP address is not possible. In this case, the connection to the Internet is via NAT (network address translation replaces the private IP address with a public one). Private IP addresses within the same local network must be unique and cannot be repeated.

3.1.8 Methods of assigning IP address

Address assignment includes assigning an IP address, a default gateway, one or more domain name servers that resolve names to IP addresses, time servers, and so forth. Before selecting the desired IP address assignment method, the following questions should be answered:

- How many devices need an IP address?
- Which devices require static IP address assignment?
- Is IP address renumbering expected in the future?
- Is the administrator required to track devices and their IP addresses?

- Do additional parameters (default gateway, name server, and so forth) have to be configured?
- Are there any availability issues?
- Are there any security issues?

The two basic IP address assignment strategies:

Static: An IP address is statically assigned to a system. The network administrator configures the IP address, default gateway, and name servers manually by entering them into a special file or files on the end system with either a graphical or text interface.

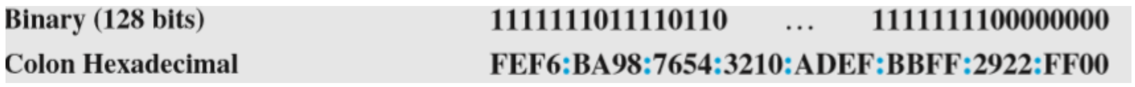
Dynamic: IP addresses are dynamically assigned to the end systems. Dynamic address assignment relieves the administrator of manually assigning an address to every network device.

3.1.9 IPv6 ADDRESSING

The main reason for migration from IPv4 to IPv6 is the small size of the address space in IPv4. In this section, we show how the huge address space of IPv6 prevents address depletion in the future. We also discuss how the new addressing responds to some problems in the IPv4 addressing mechanism. An IPv6 address is 128 bits or 16 bytes (octets) long, four times the address length in IPv4.

3.1.9.1 Representation

A computer normally stores the address in binary, but it is clear that 128 bits cannot easily be handled by humans. Several notations have been proposed to represent IPv6 addresses when they are handled by humans. The following shows two of these notations: binary and colon hexadecimal.



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

Binary notation is used when the addresses are stored in a computer. The colon hexadecimal notation (or colon hex for short) divides the address into eight sections, each made of four hexadecimal digits separated by colons.

- **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

Note that this type of abbreviation is allowed only once per address. If there is more than one run of zero sections, only one of them can be compressed.

- **Mixed Notation**

  Sometimes we see a mixed representation of an IPv6 address: colon hex and dotted-decimal notation. This is appropriate during the transition period in which an IPv4 address is embedded in an IPv6 address (as the rightmost 32 bits). We can use the colon hex notation for the leftmost six sections and four-byte dotted-decimal notation instead of the rightmost two sections. However, this happens when all or most of the leftmost sections of the IPv6 address are 0s. For example, the address (::130.24.24.18) is a legitimate address in IPv6, in which the zero compression shows that all 96 leftmost bits of the address are zeros.

- **CIDR Notation**

  IPv6 uses hierarchical addressing. For this reason, IPv6 allows slash or CIDR notation. For example, the following shows how we can define a prefix of 60 bits using CIDR.

FDEC::BBFF:0:FFFF/60

3.1.9.2 Address Space

The address space of IPv6 contains 2^{128} addresses. This address space is 2^{96} 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.

To give some idea about the number of addresses, we assume that only 1/64 (almost 2 percent) of the addresses in the space can be assigned to the people on planet Earth and the rest are reserved for special purposes. We also assume that the number of people on the earth is soon to be 234 (more than 16 billion). Each person can have 288 addresses to use. Address depletion in this version is impossible.

Three Address Types

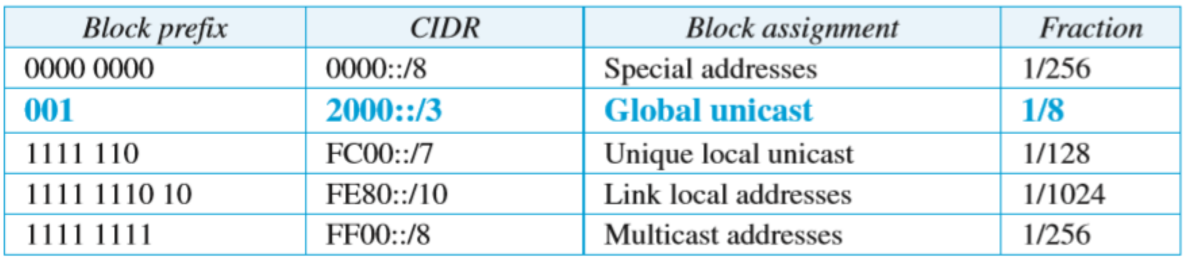
In IPv6, a destination address can belong to one of three categories: unicast, anycast, and multicast.

- **Unicast Address**: A unicast address defines a single interface (computer or router). The packet sent to a unicast address will be routed to the intended recipient.
- **Anycast Address**: An anycast address defines a group of computers that all share a single address. A packet with an anycast address is delivered to only one member of the group, the most reachable one. An anycast communication is used, for example, when there are several servers that can respond to an inquiry. The request is sent to the one that is most reachable. The hardware and software generate only one copy of the request; the copy reaches only one of the servers. IPv6 does not designate a block for anycasting; the addresses are assigned from the unicast block.
- **Multicast Address**: A multicast address also defines a group of computers. However, there is a difference between anycasting and multicasting. In anycasting, only one copy of the packet is sent to one of the members of the group; in multicasting each member of the group receives a copy. As we will see shortly, IPv6 has designated a block for multicasting from which the same address is assigned to the members of the group. It is interesting that IPv6 does not define broadcasting, even in a limited version. IPv6 considers broadcasting as a special case of multicasting.

3.1.9.3 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 future use. Table 3.1 shows only the assigned blocks. In this table, the last column shows the fraction each block occupies in the whole address space.

Table 3.1: Prefixes for assigned IPv6 addresses



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), as shown in Figure 3.13. The figure also shows the recommended length for each part.

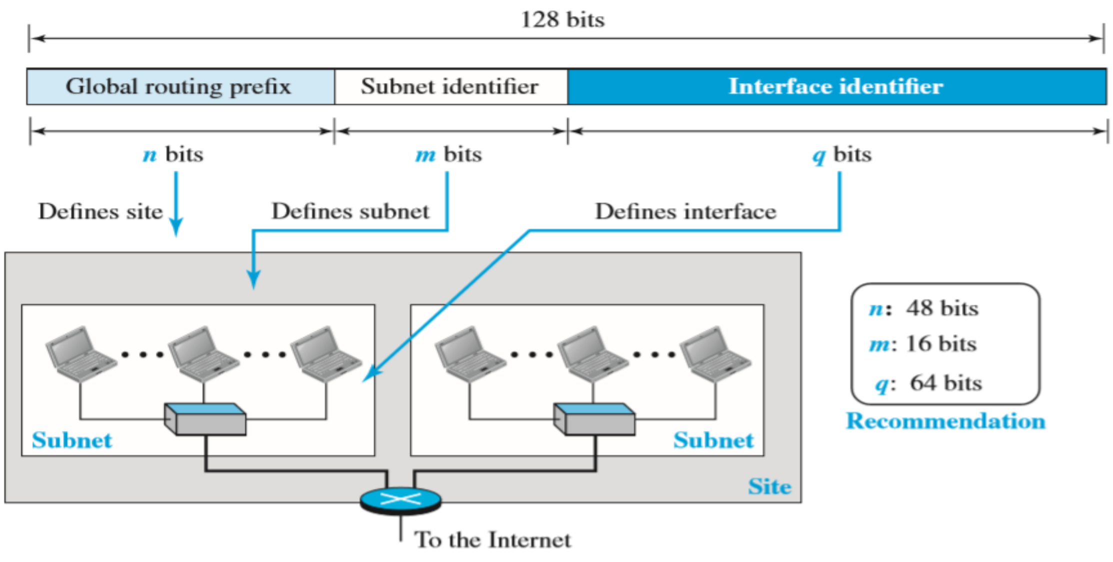


Figure 3.13: Global unicast address.

The global routing prefix is used to route the packet through the Internet to the organization site, such as the ISP that owns the block. Since the first three bits in this part are fixed (001), the rest of the 45 bits can be defined for up to 245 sites (a private organization or an ISP). The global routers in the Internet route a packet to its destination site based on the value of n. The next m bits (16 bits based on recommendation) define a subnet in an organization. This means that an organization can have up to 216 = 65,536 subnets, which is more than enough.

The last q bits (64 bits based on recommendation) define the interface identifier. The interface identifier is similar to hostid in IPv4 addressing, although the term interface identifier is a better choice because, as we discussed earlier, the host identifier actually defines the interface, not the host. If the host is moved from one interface to another, its IP address needs to be changed.

In IPv4 addressing, there is not a specific relation between the hostid (at the IP level) and link-layer address (at the data-link layer) because the link-layer address is normally much longer than the hostid. The IPv6 addressing allows this relationship. A link-layer address whose length is less than 64 bits can be embedded as the whole or part of the interface identifier, eliminating the mapping process. Two common link-layer addressing schemes can be considered for this purpose: the 64-bit extended unique identifier (EUI-64) defined by IEEE and the 48-bit link-layer address defined by Ethernet.

Mapping EUI-64

To map a 64-bit physical address, the global/local bit of this format needs to be changed from 0 to 1 (local to global) to define an interface address, as shown in Figure 3.14.

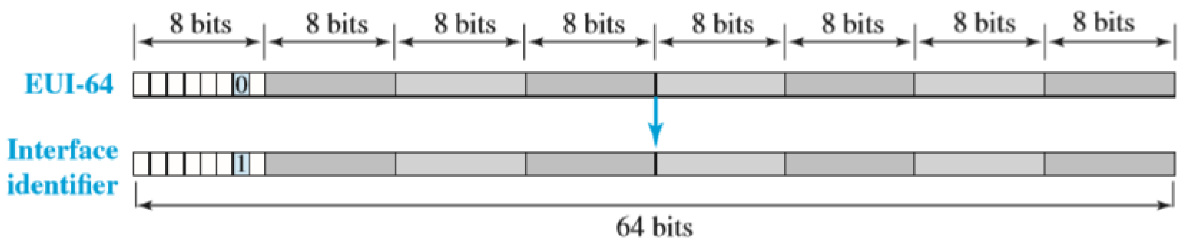


Figure 3.14: Mapping for EUI-64

Mapping Ethernet MAC Address Mapping a 48-bit Ethernet address into a 64-bit interface identifier is more involved. We need to change the local/global bit to 1 and insert an additional 16 bits. The additional 16 bits are defined as 15 ones followed by one zero, or $FFFE_{16}$.

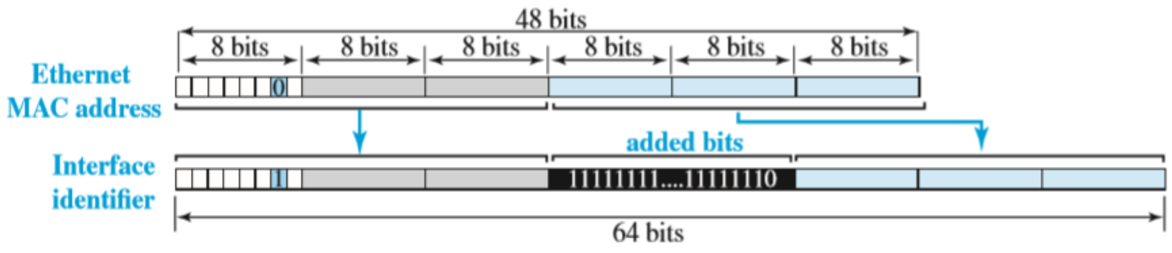


Figure 3.15: Mapping for Ethernet MAC.

Special Addresses

Addresses that use the prefix (0000::/8) are reserved, but part of this block is used to define some special addresses.

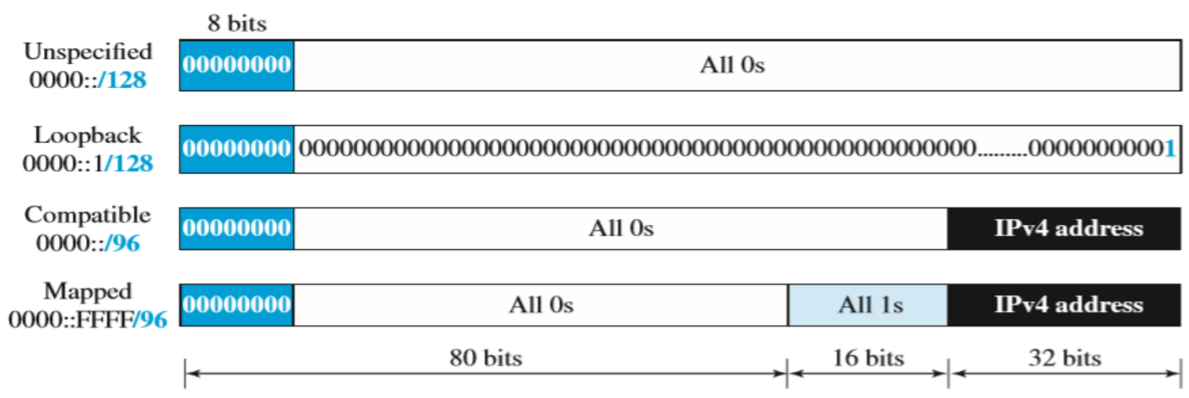


Figure 3.16: Special addresses

The unspecified address is a subblock containing only one address, which is used during bootstrap when a host does not know its own address and wants to send an inquiry to find it (see DHCP section).

The loopback address also consists of one address. We discussed loopback addresses for IPv4 before. In IPv4 the block is made of a range of addresses; in IPv6, the block has only a single address in it.

During the transition from IPv4 to IPv6, hosts can use their IPv4 addresses embedded in IPv6 addresses. Two formats have been designed for this purpose: compatible and mapped. A compatible address is an address of 96 bits of zero followed by 32 bits of IPv4 address. It is used when a computer using IPv6 wants to send a message to another computer using IPv6. A mapped address is used when a computer already migrated to version 6 wants to send an address to a computer still using version 4. A very interesting point about mapped and compatible addresses is that they are designed such that, when calculating the checksum, one can use either the embedded address or the total address because extra 0s or 1s in multiples of 16 do not have any effect in checksum calculation. This is important for UDP and TCP, which use a pseudoheader to calculate the checksum, because the checksum calculation is not affected if the address of the packet is changed from IPv6 to IPv4 by a router.

Other Assigned Blocks

IPv6 uses two large blocks for private addressing and one large block for multicasting, as shown in Figure 3.17. A subblock in a unique local unicast block can be privately created and used by a site. The packet carrying this type of address as the destination address is not expected to be routed. This type of address has the identifier 1111 110, the next bit can be 0 or 1 to define how the address is selected (locally or by an authority). The next 40 bits are selected by the site using a randomly generated number of length 40 bits. This means that the total of 48 bits defines a subblock that looks like a global unicast address. The 40-bit random number makes the probability of duplication of the address extremely small. Note the similarity between the format of these addresses and the global unicast. The second block, designed for private addresses, is the link local block. A subblock in this block can be used as a private address in a network. This type of address has the block identifier 1111111010. The next 54 bits are set to zero. The last 64 bits can be changed to define the interface for each computer. Note the similarity between the format of these addresses and the global unicast address.

Multicast addresses are used to define a group of hosts instead of just one. In IPv6 a large block of addresses are assigned for multicasting. All these addresses use the prefix 11111111. The second field is a flag that defines the group address as either permanent or transient. A permanent group address is defined by the Internet authorities and can be accessed at all times. A transient group address, on the other hand, is used only temporarily. Systems engaged in a teleconference, for example, can use a transient group address. The third field defines the scope of the group address. Many different scopes have been defined, as shown in the figure.

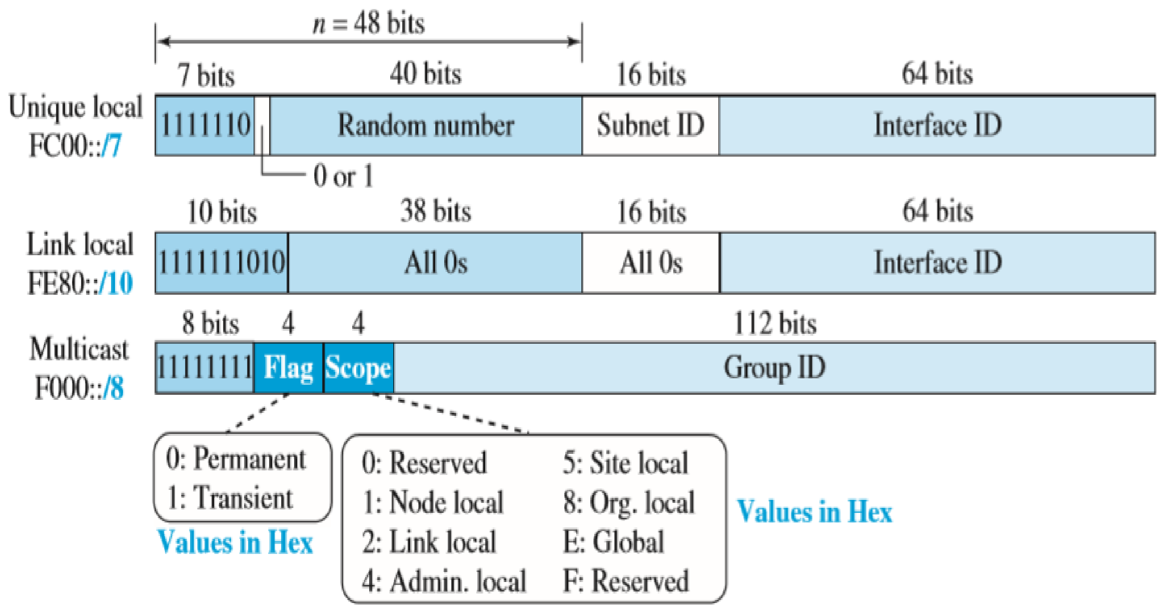


Figure 3.17: Unique local unicast block.

3.1.9.4 Autoconfiguration

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.

When a host in IPv6 joins a network, it can configure itself using the following process:

1. The host first creates a link local address for itself. This is done by taking the 10-bit link local prefix (1111 1110 10), adding 54 zeros, and adding the 64-bit interface identifier, which any host knows how to generate from its interface card. The result is a 128-bit link local address.

2. The host then tests to see if this link local address is unique and not used by other hosts. Since the 64-bit interface identifier is supposed to be unique, the link local address generated is unique with a high probability. However, to be sure, the host sends a neighbour solicitation message (see Chapter 28) and waits for a neighbour advertisement message. If any host in the subnet is using this link local address, the process fails and the host cannot autoconfigure itself; it needs to use other means such as DHCP for this purpose.

3. If the uniqueness of the link local address is passed, the host stores this address as its link local address (for private communication), but it still needs a global unicast address. The host then sends a router solicitation message (discussed later in the chapter) to a local router. If there is a router running on the network, the host receives a router advertisement message that includes the global unicast prefix and the subnet prefix that the host needs to add to its interface identifier to generate its global unicast address. If the router cannot help the host with the configuration, it informs the host in the router advertisement message (by setting a flag). The host then needs to use other means for configuration.

3.1.9.5 Renumbering

To allow sites to change the service provider, renumbering of the address prefix (n) was built into IPv6 addressing. As we discussed before, each site is given a prefix by the service provider to which it is connected. If the site changes the provider, the address prefix needs to be changed. A router to which the site is connected can advertise a new prefix and let the site use the old prefix for a short time before disabling it. In other words, during the transition period, a site has two prefixes. The main problem in using the renumbering mechanism is the support of the DNS, which needs to propagate the new addressing associated with a domain name. A new protocol for DNS, called Next Generation DNS, is under study to provide support for this mechanism.

3.1.10 Data encapsulation

- Delivery of data over IPv6 internetworks is accomplished by encapsulating higher-layer data into IPv6 datagrams.
- IPv6 datagrams use a structure that includes a regular header and optionally, one or more extension headers. This regular header is like the header of IPv4 datagrams, though it has a different format, as we will see shortly. The standards don't give this header a name; it is just "the IPv6 header"
- The IPv6 main header is required for every datagram. It contains addressing and control information that are used to manage the processing and routing of the datagram.

3.1.11 The IPv4 Datagram Format

Refer to IP part.

3.1.12 THE IPv6 PROTOCOL

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.

3.1.12.1 Packet Format

The IPv6 packet is shown in Figure 3.18. 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.

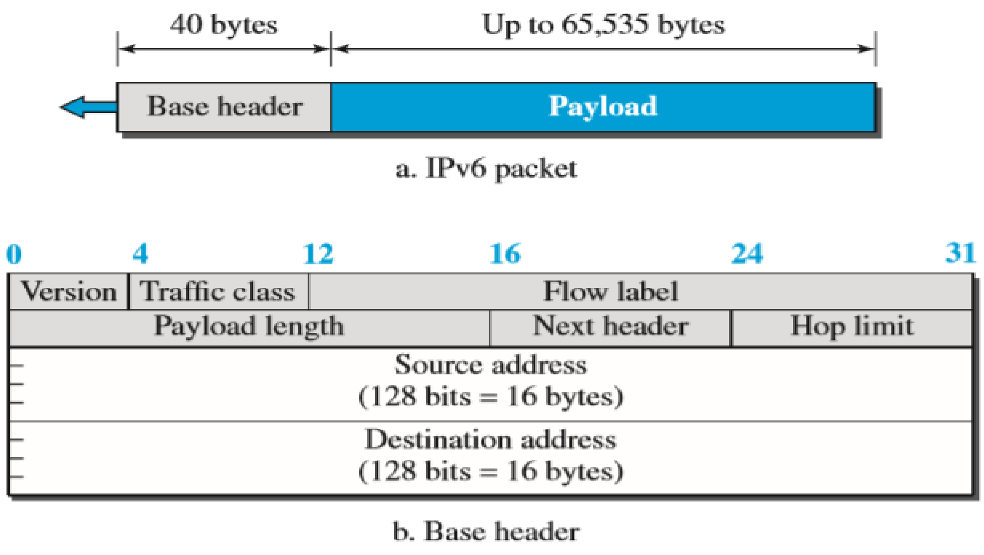


Figure 3.18: IPv6 datagram.

❑ **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 delivery requirements. 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 base header 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, but we talk more about it when we discuss the payload.

❑ **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 3.19.

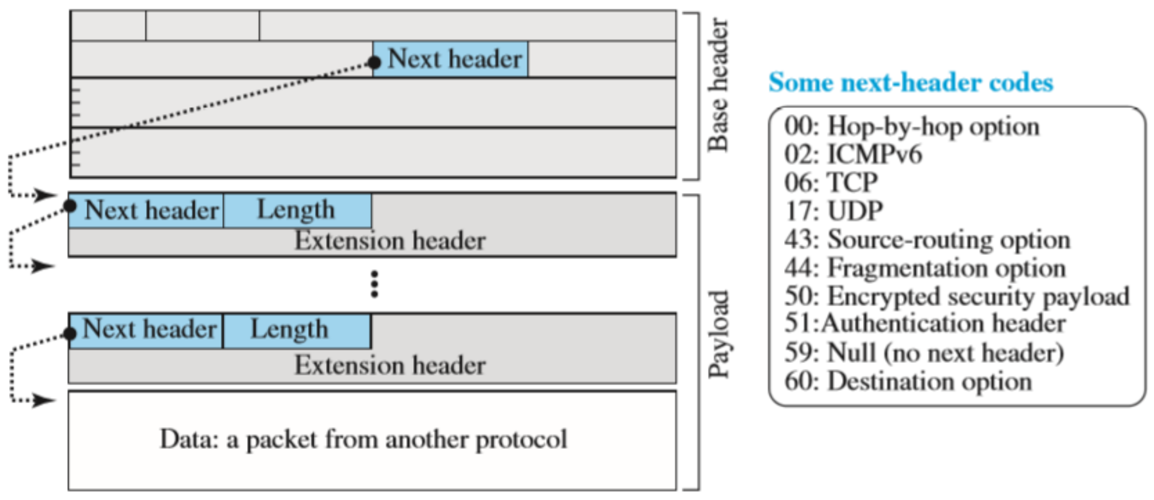


Figure 3.19: Payload in an IPv6 datagram.

The payload in IPv6 means a combination of zero or more extension headers (options) followed by the data from other protocols (UDP, TCP, and so on). In IPv6, options, which are part of the header in IPv4, are designed as extension headers. The payload can have as many extension headers as required by the situation. Each extension header has two mandatory fields, next header and the length, followed by information related to the particular option. Note that each next header field value (code) defines the type of the next header (hop-by-hop option, source-routing option, . . .); the last next header field defines the protocol (UDP, TCP, . . .) that is carried by the datagram.

3.1.12.2 Extension Header

An IPv6 packet is made of a base header and some extension headers. The length of the base header is fixed at 40 bytes. However, to give more functionality to the IP datagram, the base header can be followed by up to six extension headers. Many of these headers are options in IPv4. Six types of extension headers have been defined. These are hop-by-hop option, source routing, fragmentation, authentication, encrypted security payload, and destination option.

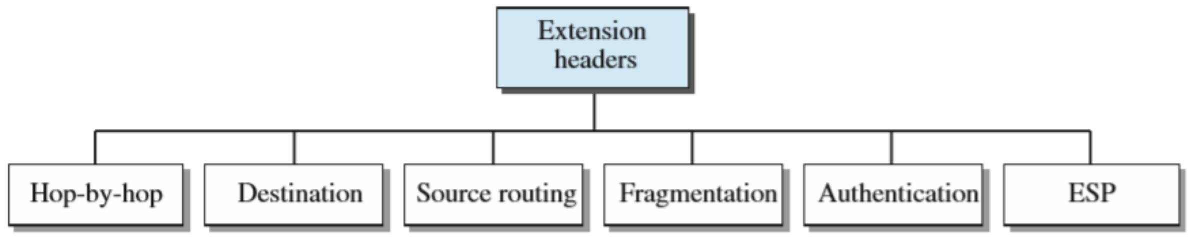


Figure 3.20: Extension header types.

Hop-by-Hop Option: The hop-by-hop option is used when the source needs to pass information to all routers visited by the datagram. For example, perhaps routers must be informed about certain management, debugging, or control functions. Or, if the length of the datagram is more than the usual 65,535 bytes, routers must have this information. So far, only three hop-by-hop options have been defined: Pad1, PadN, and jumbo payload.

❑ **Pad1:** This option is 1 byte long and is designed for alignment purposes. Some options need to start at a specific bit of the 32-bit word. If an option falls short of this requirement by exactly one byte, Pad1 is added.

❑ **PadN:** PadN is similar in concept to Pad1. The difference is that PadN is used when 2 or more bytes are needed for alignment.

❑ **Jumbo payload:** Recall that the length of the payload in the IP datagram can be a maximum of 65,535 bytes. However, if for any reason a longer payload is required, we can use the jumbo payload option to define this longer length.

Destination Option: The destination option is used when the source needs to pass information to the destination only. Intermediate routers are not permitted access to this information. The format of the destination option is the same as the hop-by-hop option. So far, only the Pad1 and PadN options have been defined.

Source Routing: The source routing extension header combines the concepts of the strict source route and the loose source route options of IPv4.

Fragmentation: The concept of fragmentation in IPv6 is the same as that in IPv4. However, the place where fragmentation occurs differs. In IPv4, the source or a router is required to fragment if the size of the datagram is larger than the MTU of the network over which the datagram travels. In IPv6, only the original source can fragment. A source must use a Path MTU Discovery technique to find the smallest MTU supported by any network on the path. The source then fragments using this knowledge.

If the source does not use a Path MTU Discovery technique, it fragments the datagram to a size of 1280 bytes or smaller. This is the minimum size of MTU required for each network connected to the Internet.

Authentication: The authentication extension header has a dual purpose: it validates the message sender and ensures the integrity of data. The former is needed so the receiver can be sure that a message is from the genuine sender and not from an imposter. The latter is needed to check that the data is not altered in transition by some hacker.

Encrypted Security Payload: The encrypted security payload (ESP) is an extension that provides confidentiality and guards against eavesdropping.

Comparison of Options between IPv4 and IPv6

The following shows a quick comparison between the options used in IPv4 and the options used in IPv6 (as extension headers).

❑ The no-operation and end-of-option options in IPv4 are replaced by Pad1 and PadN options in IPv6.

❑ The record route option is not implemented in IPv6 because it was not used.

❑ The timestamp option is not implemented because it was not used.

❑ The source route option is called the source route extension header in IPv6.

❑ The fragmentation fields in the base header section of IPv4 have moved to the fragmentation extension header in IPv6.

❑ The authentication extension header is new in IPv6.

❑ The encrypted security payload extension header is new in IPv6.

3.1.13 INTERNET CONTROL MESSAGE PROTOCOL (ICMP)

Datagram service is the best-effort service. To enhance the reliability, Internet Control Message Protocol (ICMP) is used, which provides information about errors, loss of packets, unavailable destinations, etc. It is documented as RFC 792. It is mandatory that every device that implements IP, must also implement ICMP. In ICMP, any destination or router that detects any problem in handling a received IP packet, generates an ICMP message addressed to the originating station of the IP packet. ICMP messages can be analyzed by network management systems to generate network reports for the network administrators. ICMP messages are sent as IP packets. The protocol field of IP header is set to 0x01 to indicate that this packet contains ICMP message (Figure 3.21).

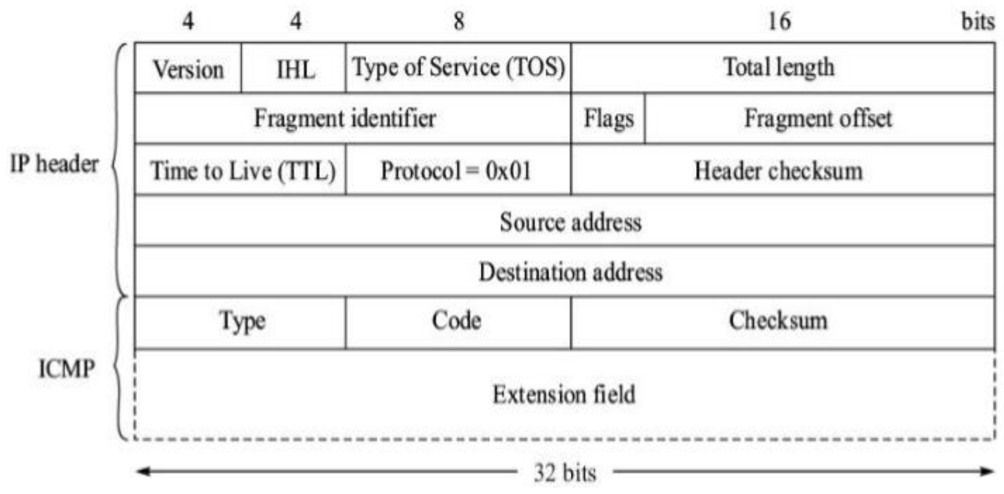


Figure 3.21: IP packet containing ICMP.

3.1.14 ICMPv4

The IPv4 has no error-reporting or error-correcting mechanism. What happens if something goes wrong? What happens if a router must discard a datagram because it cannot find a route to the final destination, or because the time-to-live field has a zero value? What happens if the final destination host must discard the received fragments of a datagram because it has not received all fragments within a predetermined time limit? These are examples of situations where an error has occurred and the IP protocol has no built-in mechanism to notify the original host.

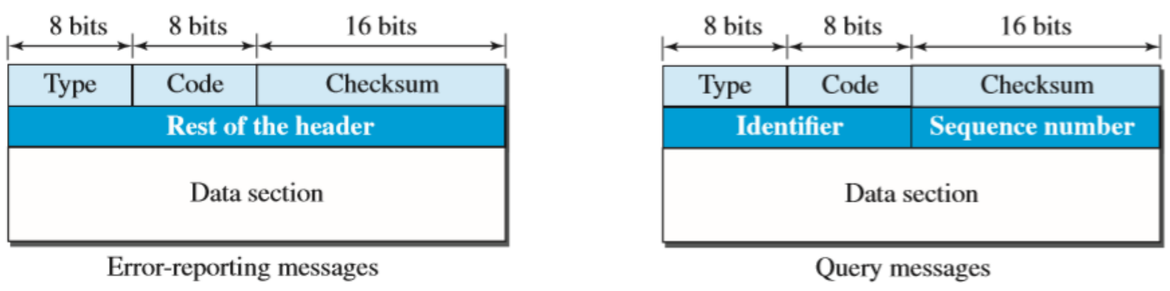
The IP protocol also lacks a mechanism for host and management queries. A host sometimes needs to determine if a router or another host is alive. And sometimes a network manager needs information from another host or router.

The Internet Control Message Protocol version 4 (ICMPv4) has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol. ICMP itself is a network-layer protocol. However, its messages are not passed directly to the data-link layer as would be expected. Instead, the messages are first encapsulated inside IP datagrams before going to the lower layer. When an IP datagram encapsulates an ICMP message, the value of the protocol field in the IP datagram is set to 1 to indicate that the IP payroll is an ICMP message.

3.1.14.1 MESSAGES

ICMP messages are divided into two broad categories: error-reporting messages and query messages. The error-reporting messages report problems that a router or a host (destination) may encounter when it processes an IP packet. The query messages, which occur in pairs, help a host or a network manager get specific information from a router or another host. For example, nodes can discover their neighbors. Also, hosts can discover and learn about routers on their network and routers can help a node redirect its messages.

An ICMP message has an 8-byte header and a variable-size data section. Although the general format of the header is different for each message type, the first 4 bytes are common to all. As Figure 3.22 shows, the first field, ICMP type, defines the type of the message. The code field specifies the reason for the particular message type. The last common field is the checksum field (to be discussed later in the chapter). The rest of the header is specific for each message type.



Type and code values

Error-reporting messages	Query messages
03: Destination unreachable (codes 0 to 15)	08 and 00: Echo request and reply (only code 0)
04: Source quench (only code 0)	13 and 14: Timestamp request and reply (only code 0)
05: Redirection (codes 0 to 3)	
11: Time exceeded (codes 0 and 1)	
12: Parameter problem (codes 0 and 1)	

Figure 3.22: General format of ICMP messages.

The data section in error messages carries information for finding the original packet that had the error. In query messages, the data section carries extra information based on the type of query.

- **Error Reporting Messages**

  Since IP is an unreliable protocol, one of the main responsibilities of ICMP is to report some errors that may occur during the processing of the IP datagram. ICMP does not correct errors, it simply reports them. Error correction is left to the higher-level protocols. Error messages are always sent to the original source because the only information available in the datagram about the route is the source and destination IP addresses. ICMP uses the source IP address to send the error message to the source (originator) of the datagram. To make the error-reporting process simple, ICMP follows some rules in reporting messages. First, no error message will be generated for a datagram having a multicast address or special address (such as this host or loopback). Second, no ICMP error message will be generated in response to a datagram carrying an ICMP error message. Third, no ICMP error message will be generated for a fragmented datagram that is not the first fragment.

  Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram. The original datagram header is added to give the original source, which receives the error message, information about the datagram itself. The 8 bytes of data are included because the first 8 bytes provide information about the port numbers (UDP and TCP) and sequence number (TCP). This information is needed so the source can inform the protocols (TCP or UDP) about the error.

The following are important points about ICMP error messages:

❑ No ICMP error message will be generated in response to a datagram carrying an ICMP error message.

❑ No ICMP error message will be generated for a fragmented datagram that is not the first fragment.

❑ No ICMP error message will be generated for a datagram having a multicast address.

❑ No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram. The original datagram header is added to give the original source, which receives the error message, information about the datagram itself. The 8 bytes of data are included because, on UDP and TCP protocols, the first 8 bytes provide information about the port numbers (UDP and TCP) and sequence number (TCP). This information is needed so the source can inform the protocols (TCP or UDP) about the error. ICMP forms an error packet, which is then encapsulated in an IP datagram.

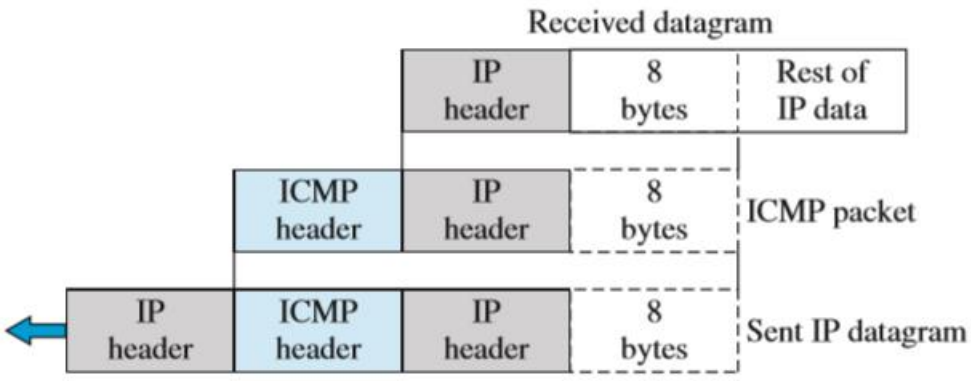


Figure 3.23: Contents of data field for the error messages.

- **Destination Unreachable**

The most widely used error message is the destination unreachable (type 3). This message uses different codes (0 to 15) to define the type of error message and the reason why a datagram has not reached its final destination. For example, code 0 tells the source that a host is unreachable. This may happen, for example, when we use the HTTP protocol to access a web page, but the server is down. The message “destination host is not reachable” is created and sent back to the source.

- **Source Quench**

  Another error message is called the source quench (type 4) message, which informs the sender that the network has encountered congestion and the datagram has been dropped; the source needs to slow down sending more datagrams. In other words, ICMP adds a kind of congestion control mechanism to the IP protocol by using this type of message.

- **Redirection Message**

  The redirection message (type 5) is used when the source uses a wrong router to send out its message. The router redirects the message to the appropriate router, but informs the source that it needs to change its default router in the future. The IP address of the default router is sent in the message.

  When the TTL value becomes 0, the datagram is dropped by the visiting router and a time exceeded message (type 11) with code 0 is sent to the source to inform it about the situation. The time-exceeded message (with code 1) can also be sent when not all fragments of a datagram arrive within a predefined period of time.

- **Parameter Problem**

  A parameter problem message (type 12) can be sent when either there is a problem in the header of a datagram (code 0) or some options are missing or cannot be interpreted (code 1).

- **Query Messages**

  Interestingly, query messages in ICMP can be used independently without relation to an IP datagram. Of course, a query message needs to be encapsulated in a datagram, as a carrier. Query messages are used to probe or test the liveliness of hosts or routers in the Internet, find the one-way or the round-trip time for an IP datagram between two devices, or even find out whether the clocks in two devices are synchronized. Naturally, query messages come in pairs: request and reply. The echo request (type 8) and the echo reply (type 0) pair of messages are used by a host or a router to test the liveliness of another host or router. A host or router sends an echo request message to another host or router; if the latter is alive, it responds with an echo reply message. We shortly see the applications of this pair in two debugging tools: ping and traceroute.

The timestamp request (type 13) and the timestamp reply (type 14) pair of messages are used to find the round-trip time between two devices or to check whether the clocks in two devices are synchronized. The timestamp request message sends a 32-bit number, which defines the time the message is sent. The timestamp reply resends that number, but also includes two new 32-bit numbers representing the time the request was received and the time the response was sent. If all timestamps represent Universal time, the sender can calculate the one-way and round-trip time.

- **Deprecated Messages**

  Three pairs of messages are declared obsolete by IETF:

  1. Information request and replay messages are not used today because their duties are done by the Address Resolution Protocol (ARP).

  2. Address mask request and reply messages are not used today because their duties are done by the Dynamic Host Configuration Protocol (DHCP).

  3. Router solicitation and advertisement messages are not used today because their duties are done by the Dynamic Host Configuration Protocol (DHCP).

3.1.14.2 Debugging Tools

There are several tools that can be used in the Internet for debugging. We can determine the viability of a host or router. We can trace the route of a packet. We introduce two tools that use ICMP for debugging: ping and traceroute.

- **Ping**

  We can use the ping program to find if a host is alive and responding. We use ping here to see how it uses ICMP packets. The source host sends ICMP echo-request messages; the destination, if alive, responds with ICMP echo-reply messages. The ping program sets the identifier field in the echo-request and echo-reply message and starts the sequence number from 0; this number is incremented by 1 each time a new message is sent. Note that ping can calculate the round-trip time. It inserts the sending time in the data section of the message. When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (RTT).

- **Traceroute or Tracert**

  The traceroute program in UNIX or tracert in Windows can be used to trace the path of a packet from a source to the destination. It can find the IP addresses of all the routers that are visited along the path. The program is usually set to check for the maximum of

30 hops (routers) to be visited. The number of hops in the Internet is normally less than this. Since these two programs behave differently in Unix and Windows, we explain them separately.

Traceroute

The traceroute program is different from the ping program. The ping program gets help from two query messages; the traceroute program gets help from two error-reporting messages: time-exceeded and destination-unreachable. The traceroute is an application layer program, but only the client program is needed, because, as we can see, the client program never reaches the application layer in the destination host. In other words, there is no traceroute server program. The traceroute application program is encapsulated in a UDP user datagram, but traceroute intentionally uses a port number that is not available at the destination. If there are n routers in the path, the traceroute program sends (n + 1) messages. The first n messages are discarded by the n routers, one by each router; the last message is discarded by the destination host. The traceroute client program uses the (n + 1) ICMP error-reporting messages received to find the path between the routers. We will show shortly that the traceroute program does not need to know the value of n; it is found automatically. In Figure 3.24, the value of n is 3.

The first traceroute message is sent with time-to-live (TTL) value set to 1; the message is discarded at the first router and a time-exceeded ICMP error message is sent, from which the traceroute program can find the IP address of the first router (the source IP address of the error message) and the router name (in the data section of the message). The second traceroute message is sent with TTL set to 2, which can find the IP address and the name of the second router. Similarly, the third message can find the information about router 3. The fourth message, however, reaches the destination host. This host is also dropped, but for another reason. The destination host cannot find the port number specified in the UDP user datagram. This time ICMP sends a different message, the destination-unreachable message with code 3 to show the port number is not found. After receiving this different ICMP message, the traceroute program knows that the final destination is reached. It uses the information in the received message to find the IP address and the name of the final destination.

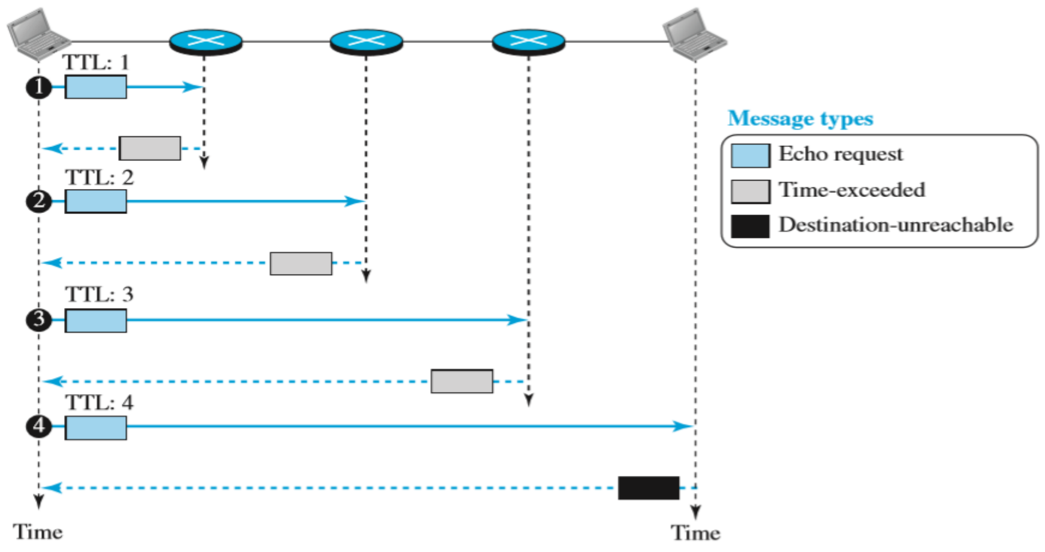


Figure 3.24: Use of ICMPv4 in traceroute.

The traceroute program also sets a timer to find the round-trip time for each router and the destination. Most traceroute programs send three messages to each device, with the same TTL value, to be able to find a better estimate for the round-trip time.

Tracert

The tracert program in windows behaves differently. The tracert messages are encapsulated directly in IP datagrams. The tracert, like traceroute, sends echo-request messages. However, when the last echo request reaches the destination host, an echo-replay message is issued.

3.1.14.3 ICMP Checksum

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

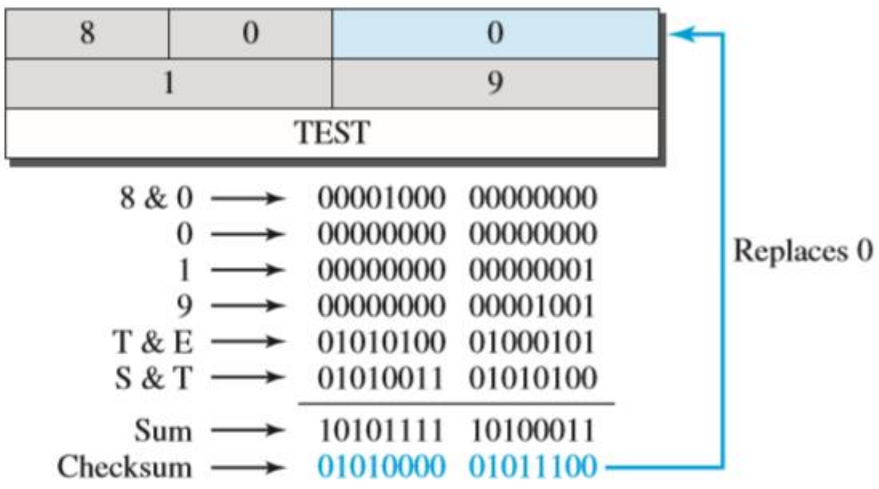


Figure 3.25: Example of checksum calculation.

Figure 3.25 shows an example of checksum calculation for a simple echo-request message. We randomly chose the identifier to be 1 and the sequence number to be 9. The message is divided into 16-bit (2-byte) words. The words are added and the sum is complemented. Now the sender can put this value in the checksum field.

3.1.15 THE ICMPv6 PROTOCOL

Another protocol that has been modified in version 6 of the TCP/IP protocol suite is ICMP. This new version, Internet Control Message Protocol version 6 (ICMPv6), follows the same strategy and purposes of version 4. ICMPv6, however, is more complicated than ICMPv4: some protocols that were independent in version 4 are now part of ICMPv6 and some new messages have been added to make it more useful. Figure 3.26 compares the network layer of version 4 to that of version 6. The ICMP, ARP and IGMP protocols in version 4 are combined into one single protocol, ICMPv6.

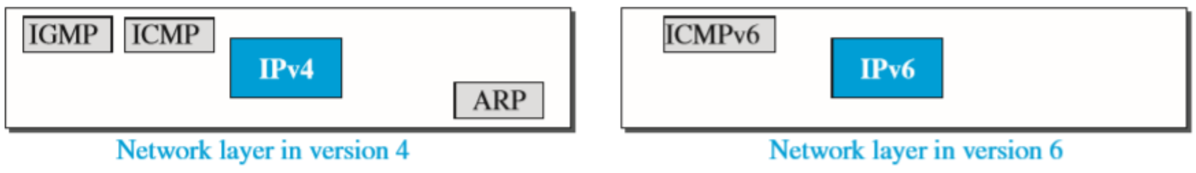


Figure 3.26: Comparison of network layer in version 4 and version 6.

The messages in ICMPv6 can be divided into four groups: error-reporting messages, informational messages, neighbour-discovery messages, and group-membership messages, as shown in Figure 3.27.

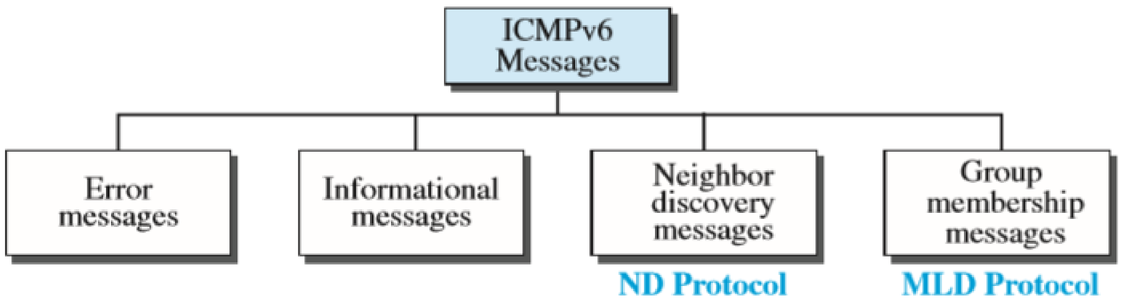


Figure 3.27: Categories of ICMPv6 messages.

3.1.15.1 Error-Reporting Messages

One of the main responsibilities of ICMPv6 is to report errors. Four types of errors are handled: destination unreachable, packet too big, time exceeded, and parameter problems. Note that the source-quenched message, which is used to control congestion in version 4, is eliminated in this version because the priority and flow label fields in IPv6 are supposed to take care of congestion. The redirection message has moved from the error-reporting category to the neighbor-discovery category, so we discuss it as part of the neighbor-discovery messages.

ICMPv6 forms an error packet, which is then encapsulated in an IPv6 datagram. This is delivered to the original source of the failed datagram.

Destination-Unreachable Message

The concept of the destination unreachable message is the same as described for ICMPv4. When a router cannot forward a datagram or a host cannot deliver the content of the datagram to the upper layer protocol, the router or the host discards the datagram and sends a destination-unreachable error message to the source host.

Packet-Too-Big Message

This is a new type of message added to version 6. Since IPv6 does not fragment at the router, if a router receives a datagram that is larger than the maximum transmission unit (MTU) size of the network through which the datagram should pass, two things happen. First, the router discards the datagram. Second, an ICMP error packet—a packet-too-big message—is sent to the source.

Time-Exceeded Message

A time-exceeded error message is generated in two cases: when the time to live value becomes zero and when not all fragments of a datagram have arrived in the time limit. The format of the time-exceeded message in version 6 is similar to the one in version 4. The only difference is that the type value has changed to 3.

Parameter-Problem Message

Any ambiguity in the header of the datagram can create serious problems as the datagram travels through the Internet. If a router or the destination host discovers any ambiguous or missing value in any field, it discards the datagram and sends a parameter-problem message to the source. The message in ICMPv6 is similar to its version 4 counterpart.

3.1.15.2 Informational Messages

Two of the ICMPv6 messages can be categorized as informational messages: echo request and echo reply messages. The echo-request and echo-reply messages are designed to check whether two devices in the Internet can communicate with each other. A host or router can send an echo-request message to another host; the receiving computer or router can reply using the echo-reply message.

Echo-Request Message

The idea and format of the echo-request message is the same as the one in version 4.

Echo-Reply Message

The idea and format of the echo-reply message is the same as the one in version 4.

3.1.15.3 Neighbor-Discovery Messages

Several messages in ICMPv4 have been redefined in ICMPv6 to handle the issue of neighbor discovery. Some new messages have also been added to provide extension. The most important issue is the definition of two new protocols that clearly define the functionality of these group messages: the Neighbor-Discovery (ND) protocol and the Inverse-Neighbor-Discovery (IND) protocol. These two protocols are used by nodes (hosts or routers) on the same link (network) for three main purposes:

1. Hosts use the ND protocol to find routers in the neighborhood that will forward packets for them.

2. Nodes use the ND protocol to find the link-layer addresses of neighbors (nodes attached to the same network).

3. Nodes use the IND protocol to find the IPv6 addresses of neighbors.

Router-Solicitation Message

The idea behind the router-solicitation message is the same as in version 4. A host uses the router-solicitation message to find a router in the network that can forward an IPv6 datagram for the host. The only option that is so far defined for this message is the inclusion of the physical (data-link layer) address of the host to make the response easier for the router.

Router-Advertisement Message

The router-advertisement message is sent by a router in response to a router solicitation message.

Neighbor-Solicitation Message

The network layer in version 4 contains an independent protocol called Address Resolution Protocol (ARP). In version 6, this protocol is eliminated, and its duties are included in ICMPv6. The neighbor solicitation message has the same duty as the ARP request message. This message is sent when a host or router has a message to send to a neighbor. The sender knows the IP address of the receiver, but needs the data-link address of the receiver. The data-link address is needed for the IP datagram to be encapsulated in a frame. The only option announces the sender data-link address for the convenience of the receiver. The receiver can use the sender data-link address to send a unicast response.

Neighbor-Advertisement Message

The neighbor-advertisement message is sent in response to the neighbor-solicitation message.

Redirection Message

The purpose of the redirection message is the same as described for version 4. However, the format of the packet now accommodates the size of the IP address in version 6. Also, an option is added to let the host know the physical address of the target router.

Inverse-Neighbor-Solicitation Message

The inverse-neighbor-solicitation message is sent by a node that knows the link-layer address of a neighbor, but not the neighbor’s IP address. The message is encapsulated in an IPv6 datagram using an all-node multicast address. The sender must send the following two pieces of information in the option field: its link-layer address and the link-layer address of the target node. The sender can also include its IP address and the MTU value for the link.

Inverse-Neighbor-Advertisement Message

The inverse-neighbor-advertisement message is sent in response to the inverse-neighbor-discovery message. The sender of this message must include the link-layer address of the sender and the link-layer address of the target node in the option section.

3.1.15.4 Group Membership Messages

The management of multicast delivery handling in IPv4 is given to the IGMPv3 protocol. In IPv6, this responsibility is given to the Multicast Listener Delivery protocol. MLDv1 is the counterpart to IGMPv2; MLDv2 is the counterpart to IGMPv3. The material discussed in this section is taken from RFC 3810. The idea is the same as we discussed in IGMPv3, but the sizes and formats of the messages have been changed to fit the larger multicast address size in IPv6. Like IGMPv3, MLDv2 has two types of messages: membership-query message and membership-report message. The first type can be divided into three subtypes: general, group-specific, and group-and-source specific.

Membership-Query Message

A membership-query message is sent by a router to find active group members in the network. The fields are almost the same as the ones in IGMPv3 except that the size of the multicast address and the source address has been changed from 32 bits to 128 bits. Another noticeable change in the field size is in the maximum response code field, in which the size has been changed from 8 bits to 16 bits. We will discuss this field shortly. Also note that the format of the first 8 bytes matches the format for other ICMPv6 packets because MLDv2 is considered to be part of ICMPv6.

Membership-Report Message

The format of the membership report in MLDv2 is exactly the same as the one in IGMPv3 except that the sizes of the fields are changed because of the address size. In particular, the record type is the same as the one defined for IGMPv3 (types 1 to 6).

3.1.16 IGMP

The protocol that is used today for collecting information about group membership is the Internet Group Management Protocol (IGMP). IGMP is a protocol defined at the network layer; it is one of the auxiliary protocols, like ICMP, which is considered part of the IP. IGMP messages, like ICMP messages, are encapsulated in an IP datagram.

3.1.16.1 Messages

There are only two types of messages in IGMP version 3, query and report messages, as shown in Figure 3.28. A query message is periodically sent by a router to all hosts attached to it to ask them to report their interests about membership in groups. A report message is sent by a host as a response to a query message.

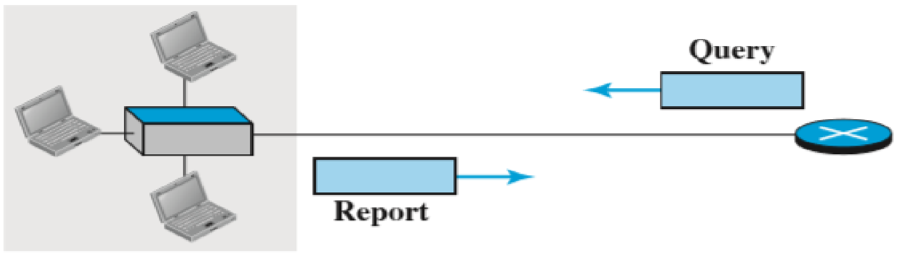


Figure 3.28: IGMP operation.

Query Message

The query message is sent by a router to all hosts in each interface to collect information about their membership. There are three versions of query messages, as described below:

a. A general query message is sent about membership in any group. It is encapsulated in a datagram with the destination address 224.0.0.1 (all hosts and routers). Note that all routers attached to the same network receive this message to inform them that this message is already sent and that they should refrain from resending it.

b. A group-specific query message is sent from a router to ask about the membership related to a specific group. This is sent when a router does not receive a response about a specific group and wants to be sure that there is no active member of that group in the network. The group identifier (multicast address) is mentioned in the message. The message is encapsulated in a datagram with the destination address set to the corresponding multicast address. Although all hosts receive this message, those not interested drop it.

c. A source-and-group-specific query message is sent from a router to ask about the membership related to a specific group when the message comes from a specific source or sources. Again, the message is sent when the router does not hear about a specific group related to a specific host or hosts. The message is encapsulated in a datagram with the destination address set to the corresponding multicast address. Although all hosts receive this message, those not interested drop it.

Report Message

A report message is sent by a host as a response to a query message. The message contains a list of records in which each record gives the identifier of the corresponding group (multicast address) and the addresses of all sources that the host is interested in receiving messages from (inclusion). The record can also mention the source addresses from which the host does not desire to receive a group message (exclusion). The message is encapsulated in a datagram with the multicast address 224.0.0.22 (multicast address assigned to IGMPv3). In IGMPv3, if a host needs to join a group, it waits until it receives a query message and then sends a report message. If a host needs to leave a group, it does not respond to a query message. If no other host responds to the corresponding message, the group is purged from the router database.

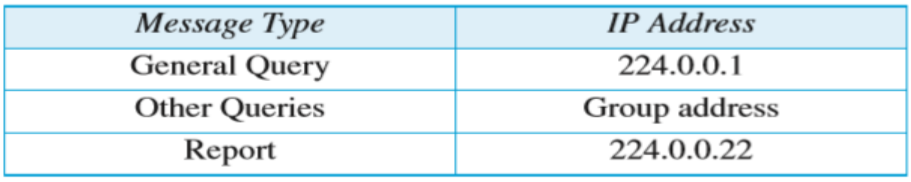
3.1.16.2 Propagation of Membership Information

After a router has collected membership information from the hosts and other routers at its own level in the tree, it can propagate it to the router located in a higher level of the tree. Finally, the router at the tree root can get the membership information to build the multicast tree.

3.1.16.3 Encapsulation

The IGMP message is encapsulated in an IP datagram with the value of the protocol field set to 2 and the TTL field set to 1. The destination IP address of the datagram, however, depends on the type of message, as shown in Table 3.2.

Table 3.2: Destination IP Addresses



Message Type	*IP Address*
General Query	224.0.0.1
Other Queries	Group address
Report	224.0.0.22

3.1.17 Introduction to Routing and Switching concepts Router

Router

A router is a physical or virtual appliance that passes information between two or more packet-switched computer networks. A router inspects a given data packet's destination Internet Protocol address (IP address), calculates the best way for it to reach its destination and then forwards it accordingly. A router is a common type of gateway. It is positioned where two or more networks meet at each point of presence on the internet.

Working of a router

An IP datagram is routed hop-by-hop across a network to the destination using forwarding tables that are stored in the routers in advance. A forwarding table in a router contains <destination network prefix, next hop> associations (Figure 3.29). When an IP packet is received by a router, its Destination Address (DA) is matched with one of the entries in the forwarding table and the IP packet is forwarded through the interface indicated in the forwarding table. The entire process described above is the forwarding or packet switching process. Routing is the process of creating and maintaining the forwarding tables.

The routing instance of a forwarding table (or simply a route) can be created manually. Such routes are called static routes. A forwarding table can have all static routes in it. The network administrator updates the static routes as and when required.

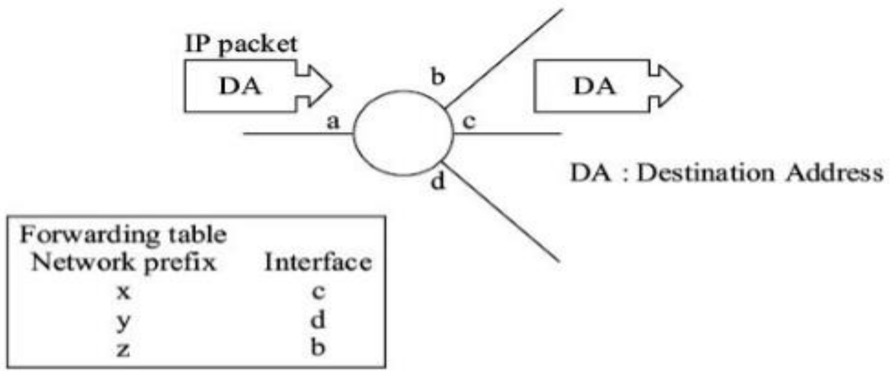
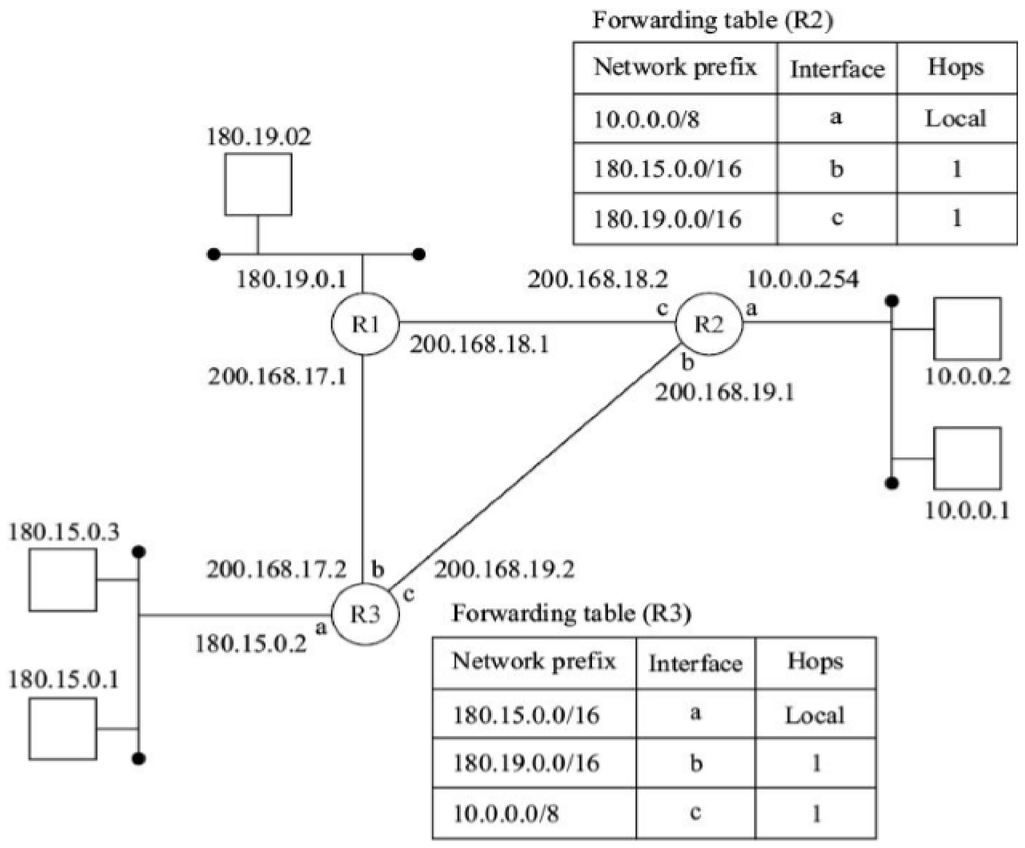


Figure 3.29: Forwarding table.

Procedure of packets getting scan in router to enter a network (Complete Picture of IP Packet Delivery)

The process of IP packet delivery from a host to another host interconnected through an IP network is described below. Let us assume that host with IP address 180.15.0.1 wants to send an IP packet to host having IP address 10.0.0.1 (Figure 3.30). We assume that routers R1, R2, and R3 are interconnected using a data link protocol (e.g. HDLC) which is already in data transfer mode. The local area network connecting the hosts and routers is ethernet. The process is as follows:

- Host 180.15.0.1 sends ARP-request packet using the target IP address (180.15.0.2) of its default gateway router R3. The packet is encapsulated in ethernet frame with destination MAC address as broadcast address.



Forwarding table (R2)

Network prefix	Interface	Hops
10.0.0.0/8	a	Local
180.15.0.0/16	b	1
180.19.0.0/16	c	1

Forwarding table (R3)

Network prefix	Interface	Hops
180.15.0.0/16	a	Local
180.19.0.0/16	b	1
10.0.0.0/8	c	1

Figure 3.30: Forwarding of IP packets.

- R3 replies with ARP-response indicating its MAC address. ARP-response is encapsulated in ethernet frame with MAC addresses of R3 and the host.
- Host (180.15.0.1) sends the IP packet addressed to the host 10.0.0.1. The IP packet is encapsulated in the ethernet frame with the MAC address of R3.
- R3 receives the IP packet, consults its forwarding table, which indicates that the packet should be forwarded through its interface c. It encapsulates the packet in the data link frame (e.g. HDLC) and sends the frame to R2.
- R2 receives the frame and the encapsulated IP packet. It consults its forwarding table and determines that the destination IP address is on the local network. It sends an ARP-request packet with target IP address 10.0.0.1 to find the MAC address of the destination host. The ARP-request is sent using ethernet encapsulation and broadcast MAC address.
- Host 10.0.0.1 sends its ARP-response to R2 containing its MAC address. It uses ethernet encapsulation with MAC address of R2.

- R2 sends the IP packet received from host 180.15.0.1 to the host 10.0.0.1. The packet is sent in an ethernet frame with MAC address of the destination host.

The subsequent IP packets from the host 180.15.0.1 to the host 10.0.0.1 do not invoke ARP since the ARP cache contains the required MAC addresses.

Advantages of Router:

a) Router can function on LAN & WAN.

b) Router can connect different media & architectures.

c) Router can determine best path/route for data to reach

the destination.

Disadvantages of Router:

a) Router only works with routable protocol.

b) Routing updates consume bandwidth.

c) Increase latency due to greater degree of packet filtering.

3.2 Transport Layer

The transport layer is situated above the network layer in an end system. While the network layer and the other lower layers reside in all the end systems and intermediate systems (network nodes, routers), the transport layer is implemented only in the end systems (Figure 20.1). The interactions between peer transport layer entities are, therefore, end-to-end and these are made possible by the data transfer service provided by the network layer.

Figure 3.31 shows the two important layered network architectures—OSI and TCP/IP suite. In the OSI architecture, the transport layer is sandwiched between the session and network layers. The network layer of OSI can be X.25 or ISO CLNP.

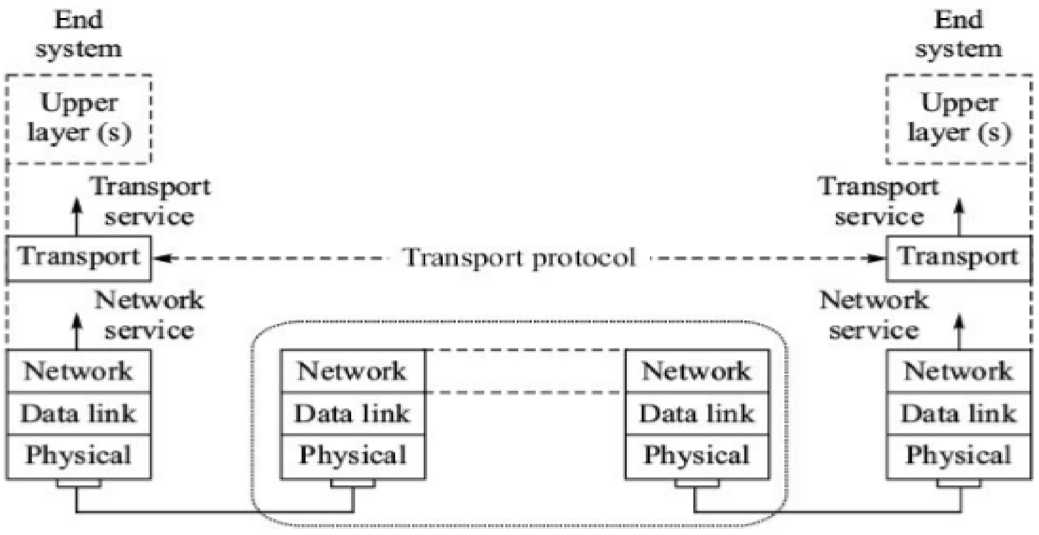


Figure 3.31: Transport layer.

In the TCP/IP suite, the transport layer has two different protocols—TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) as shown in Figure 3.32. The upper layer is application layer and the layer below is IP layer. TCP and UDP are the two most widely used transport layer protocols which are described in detail in this chapter. OSI transport layer protocol is well documented but there are few implementations.

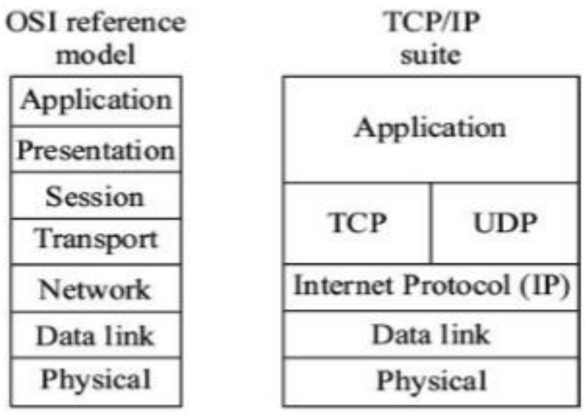


Figure 3.32: Layered architecture of OSI and TCP/IP suite.

Purpose of Transport Layer

The goal of the transport layer is to abstract the structure and function of the network so that the upper layer can communicate without needing to consider the network technology. The transport layer provides transparent, reliable, and cost effective transfer of data units between the upper layer entities in the end systems. Transparency implies that the transport layer does not place any restriction on the content of the data units received from the user entities.

Reliable, in this context, implies that the data units are delivered to the user entities as they are received from them. If an error occurs, the same is either corrected or reported to the user. Cost effectiveness implies that the underlying network service is utilized in the most optimum way to provide the quality of service requested by the users.

Types of Transport Service

Two basic types of transport service are possible:

- Connection-oriented transport service
- Connectionless transport service.

Connection-oriented transport service: In the connection-oriented service, a transport connection is established between the user entities on their request. Transfer of the user data units takes place on this connection. The following capabilities are offered to the connection-oriented transport service users.

Transport connections establishment and release. The transport service user can establish and release a transport connection with another transport service user. The users can request and negotiate a certain quality of transport service parameters. Either user can unconditionally release the connection. The connection may also be released by the transport layer.

Normal data transfer. The users can request for transfer of user data having any integral number of octets. The mode of data transfer can be two-way simultaneous. The connection-oriented transport service maintains integrity of user data in the sense that content errors, disordering of sequence, duplication of user data octets are taken care of.

Urgent data. The user can mark a segment of data as urgent. The transport entity delivers the segment to receiving user alerting him that data being received is urgent.

Forced data delivery. Usually the transport entity does not recognize the boundaries of user data segments and transports user data as a continuous stream of octets. But if required, a user may force the transport entity for delivery of a segment of user data or even a single octet of user data.

Quality of service. The transport layer allows the user entity to specify quality of service parameters.

Connectionless transport service: Because of the connectionless operation, there is no connection establishment or release phase. The service is unreliable in the sense that user data may be lost, corrupted or disordered. This is consistent with the connectionless-mode of operation.

3.2.1 Transmission Control Protocol (TCP)

TCP is a connection oriented layer-4 protocol and it provides reliable service between computer processes (applications) that reside in two different end systems. It uses services of underlying IP layer which provides connectionless and unreliable service. TCP, therefore, has inbuilt mechanisms for error control, flow control, and sequenced delivery of the user data octets. TCP is specified in RFC 793. Its basic operational features are as follows:

- It has three phases of operation connection—establishment, data transfer, and disconnection phase.
- A data unit at TCP layer is called TCP segment, which is equivalent to a frame at layer 2 and packet at layer 3. A TCP segment contains a header and user data octets.
- Continuous repeat request with sliding window flow control is used.
- The window contains user data octets, not full TCP segments. Thus flow control is based on volume of user data octets, not on number of TCP segments that can be transmitted.
- Window size is dynamic and is controlled by the receiver at the other end.
- All user data octets are acknowledged but acknowledgements may not be individual. It has inbuilt timeout mechanism for retransmission of unacknowledged TCP segments. Acknowledgements are piggybacked on a TCP data segment.
- Mode of communication is two-way simultaneous.

3.2.2 User Datagram Protocol (UDP)

UDP is a connectionless transport protocol that operates on IP layer. It is specified in RFC 768. Its main features are:

- UDP provides connectionless service to the application layer. Being connectionless, its service is unreliable.
- Each UDP datagram is transported from source to the destination independent of others. The delivery of user data may not be sequenced.
- There is no flow control or acknowledgement mechanism.
- There is optional checksum field in UDP datagram. It provides end-to-end error detection capability in the user data.

- It is less complex than TCP and easy to implement.

UDP is used where the overhead of a connection-oriented service is undesirable. Examples of such applications are given below:

- Periodic collection of data from sensors, transmission of alarms from security devices.
- Outward transmission of short broadcast messages.

UDP is also used where retransmission of lost segments makes no sense. For example, UDP is used for transmission of digital voice over IP. Retransmission of a lost voice packet does not make sense.

Format of UDP Datagram

There is only one type of UDP datagram.

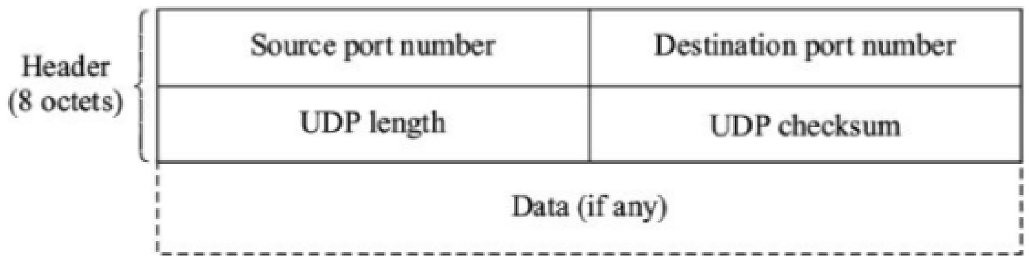


Figure 3.33: Format of UDP datagram.

- **Source port number (2 octets)**: Port number of the source process.
- **Destination port number (2 octets)**: Port number of the destination process.
- **UDP length (2 octets)**: This field indicates length of UDP datagram (header plus data) in octets.
- **UDP checksum (2 octets)**: This field contains 1's complement of the sum of all the 1's complements of 16-bit words in the UDP segment including user data field and pseudo IP header. If the computed value UDP checksum comes to all zeroes, all checksum bits are set to 1. UDP checksum is optional. If it is not used, it is set to all zeroes.

Pseudo IP header enables detection of wrong delivery of UDP datagrams by the IP layer. The format of UDP pseudo header is shown in Figure 3.34. The protocol type code is 17 for UDP. The UDP segment length includes number of octets in the UDP datagram (excluding pseudo header).

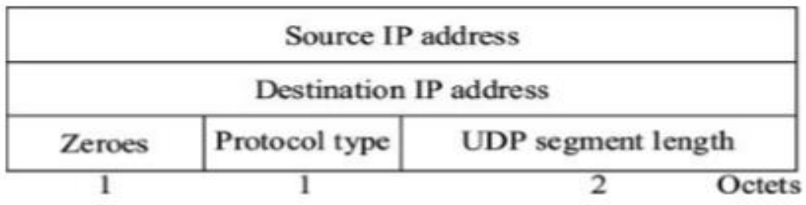


Figure 3.34: Pseudo UDP header.

UDP Operation

- UDP operation is very simple. There is no connection establishment phase. UDP is always in data transfer phase. As and when user data is handed over by the application layer, it sends it to the destination as an independent data unit.
- As a transport layer protocol, UDP adds little but needed capability to the IP service. It enables port addressing capability. IP addressing scheme enables delivery of a datagram to the end systems. There can be multiple applications in an end system. UDP supports simultaneous communications with these applications and delivers the user data to respective applications in an end system.
- There is no flow control in UDP. As regards the error control, recall that IP does not compute checksum over the payload portion of the IP packet. Therefore, content errors in the payload of an IP packet go unchecked. UDP checksum is, therefore, the only way to guarantee that the user data is intact. UDP datagram received with content error are simply discarded.
- The checksum also ensures delivery of a datagram to its correct destination. At the destination, the IP layer hands over the UDP datagram along with the source and destination IP addresses. The receiving UDP entity reconstructs pseudo header and then verifies the checksum. Wrong deliveries are discarded.

3.2.3 Overview of Ports & Sockets

TCP provides service to the higher layer through a port. A port is similar to Service Access Point (SAP) in OSI. Each communicating computer process is assigned a port number. By using different port numbers, a TCP layer can simultaneously serve multiple processes, e.g. e-mail, FTP, etc. as shown in Figure 3.35. Port number range 0–1023 is reserved for common applications. These are assigned by IANA (Internet Assigned Numbers Authority). Some well-known ports are listed below:

7	Echo	20	File Transfer Protocol (FTP)
23	Telnet	69	Trivial File Transfer Protocol (TFTP)
25	Simple Mail Transfer Protocol (SMTP)	80	Hyper Text Transfer Protocol (HTTP)

The combination of IP address and port is called a socket. Each socket pair uniquely identifies a connection. The connection between processes P1 and P4 in end systems A and B respectively can be written as <socket (33, 10.0.0.1), socket (67, 10.0.0.3)>.

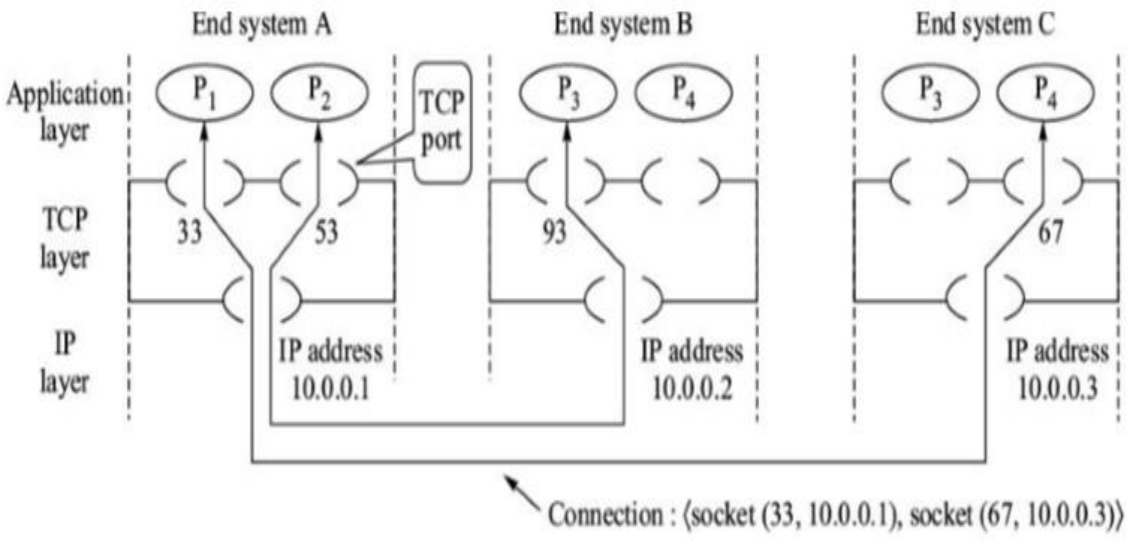


Figure 3.35: TCP ports and connections.

In a client-server environment, a server process may need to maintain several simultaneous sessions with clients on different end systems. Thus a single port supports multiple virtual connections (Figure 3.36).

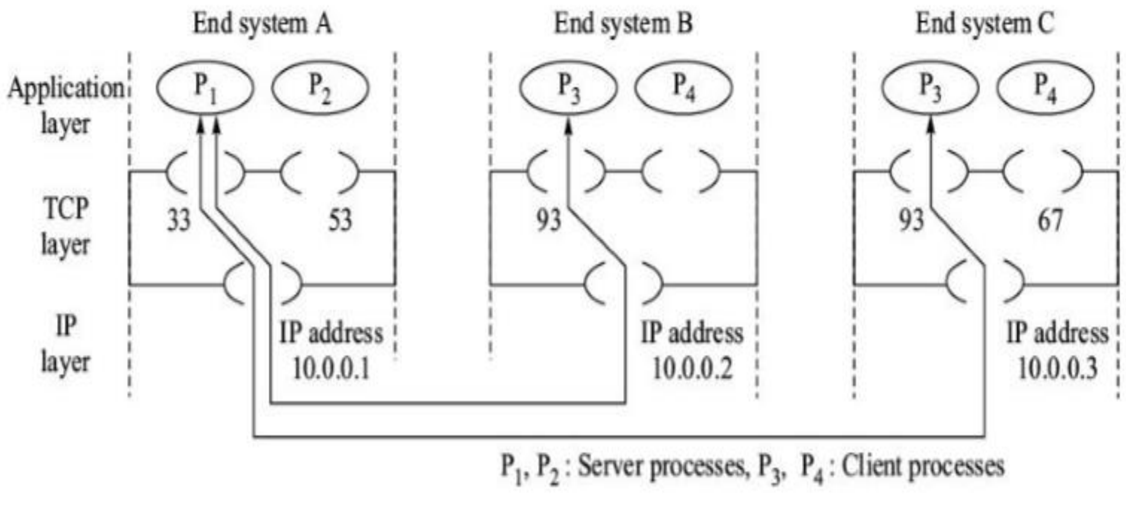


Figure 3.36: Multiple TCP connections through a port.

3.3 Application Layer

The whole Internet, hardware and software, was designed and developed to provide services at the application layer. The fifth layer of the TCP/IP protocol suite is where these services are provided for Internet users. The other four layers are there to make these services possible.

The application layer provides services to the user. Communication is provided using a logical connection, which means that the two application layers assume that there is an imaginary direct connection through which they can send and receive messages.

TCP/IP APPLICATION PROTOCOLS

Unlike the OSI reference model, TCP/IP applications run directly on the TCP or UDP layer. The functions of the session and the presentation layers of OSI reference model are integrated in the applications. A selection of the important network applications of the TCP/IP suite consists of (Figure 3.37):

- Bootstrapping Protocol (BOOTP)
- Dynamic Host Configuration Protocol (DHCP)
- File Transfer Protocol (FTP)
- Simple Mail Transfer Protocol (SMTP)
- Telnet Hyper Text Transfer Protocol (HTTP)
- Trivial File Transfer Protocol (TFTP)
- Simple Network Management Protocol (SNMP)
- Domain Name System (DNS)

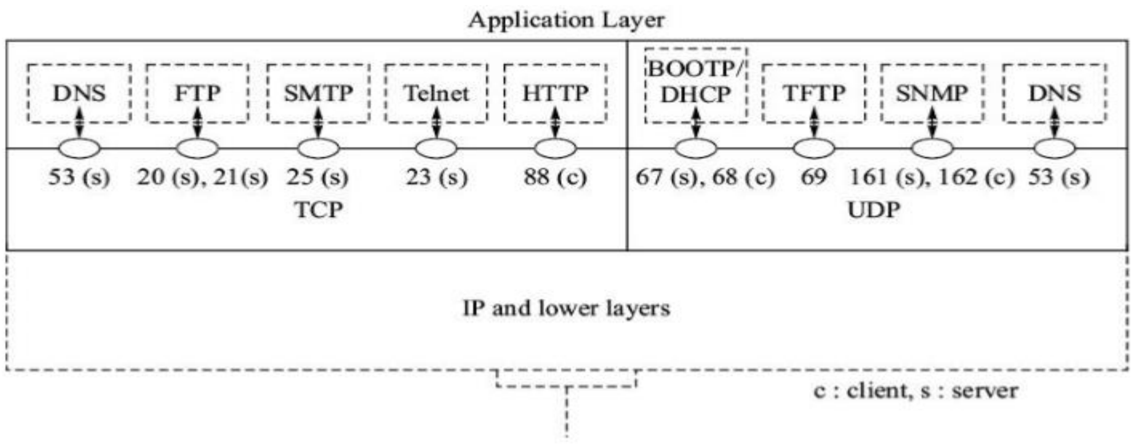


Figure 3.37: Application layer of TCP/IP suite.

These applications usually form part of a larger application or are used by a larger application. For example, e-mail is a very popular distributed application and Simple Mail Transfer Protocol (SMTP) is that part of the email application that resides in the application layer. These applications use either the TCP port or the UDP port for transfer of information and have been assigned specific port numbers. Note that some of the applications use two ports. The interaction between two communicating application entities (e.g. between two SMTP entities in two end systems) is based on client-server model which we discuss next. The server ports are always specified. The client ports, if not specified, can be any port having port number > 1023.

3.3.1 DYNAMIC HOST CONFIGURATION PROTOCOL (DHCP)

BOOTP is suitable for static environment where the client's machine does not move from one network to another. This is so because the network manager creates in the server BOOTP file that contains the configuration parameters for each host. As and when a request comes, the parameters are sent to the requesting client based on the BOOTP file.

With the advent of the portable clients (laptops, wireless networking), the clients can move from one network to another and therefore the configuration file needs dynamic updating. A new protocol called Dynamic Host Configuration Protocol (DHCP) was designed to handle such situation. A DHCP server has an address pool and when a request comes, it allocates an address from the pool. Thus, DHCP extends the functionality of BOOTP to include mechanisms to allocate host (client's) addresses dynamically.

DHCP is based on client-server mode of operation and uses the same UDP ports as BOOTP.

3.3.1.1 Configurable Parameters

DHCP has three modes of assignment of IP addresses:

1. **Automatic**: DHCP assigns the address on permanent basis.

2. **Dynamic**: The address is assigned (leased) for a limited period.

3. **Manual**: The address assignment is manually configured as in BOOTP.

In addition to assignment of IP address, DHCP allows number of parameters to be configured. A client can ask for subnet mask, DNS server, default TTL value, MTU size, source routing option, maximum fragment size, ARP cache timeout, etc.

3.3.1.2 Message Format

The message format of DHCP is same as that of BOOTP (Figure 22.8) with the following changes:

- The options area is at least 312 octets instead of 64 octets. All the DHCP messages (described later) are appended here.
- The reserved field is called flag field in DHCP. The left-most bit of the field is set to 1 by a client if it wants the response from the server in broadcast mode. Other bits of the field are set to zero.

3.3.1.3 Message Types

The following types of messages are sent by the client and by the server. These messages are contained in the options area and are identified by option code 53 and type field which contains the type value.

DHCPDISCOVER (Type 1): Broadcast from client to find DHCP server(s).

DHCPOFFER (Type 2): Response from the server to DHCPDISCOVER offering IP and other parameters.

DHCPREQUEST (Type 3): Message from the client to indicate selection of the parameters offered by a server. It is also indication to other servers declining their offers. The client also requests extension of lease time using this message.

DHCPDECLINE (Type 4): Message from the client to the server indicating an error. **DHCPACK (Type 5)**: Acknowledgement from the server to the client.

DHCPNACK (Type 6): Negative acknowledgement from the server to client if the IP address requested by the client is invalid or the lease is expired.

DHCPRELEASE (Type 7): Message from the client to the server canceling remainder of the lease and releasing the IP address.

DHCPINFORM (Type 8): Message from the client that it already has a configured IP address but needs additional configuration parameters.

Figure 3.38 shows a typical exchange of DHCP messages between a client and a server for IP address lease. The client sends IP lease request using DHCPDISCOVER message. The servers sends its offer using DHCPOFFER message. The client indicates its selection of lease to the server by sending DHCPREQUEST message. The server accepts the request by sending DHCPACK message.

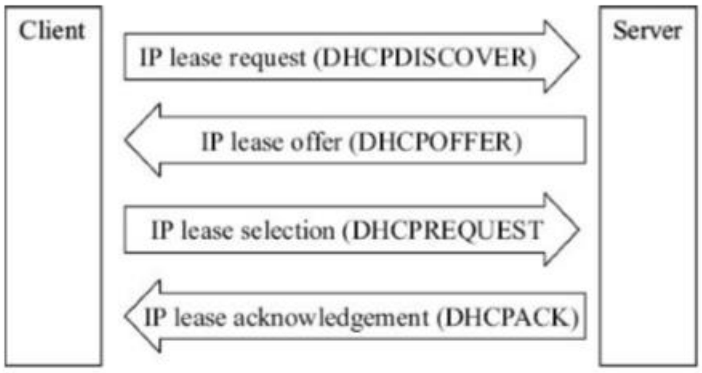

Client
Server
IP lease request (DHCPDISCOVER)
IP lease offer (DHCPOFFER)
IP lease selection (DHCPREQUEST
IP lease acknowledgement (DHCPACK)

Figure 3.38: Typical exchange of DHCP messages.

3.3.2 DOMAIN NAME SYSTEM (DNS)

Users are more comfortable with names than with digits. Thus we need a system that maps addresses to names and vice versa. The naming system used by TCP/IP is called Domain Name System (DNS). This system was developed in 1984 by P. Mokapetris of IAB. The RFCs associated with DNS are 1032, 1033, 1034, and 1035.

3.3.2.1 Hierarchical Naming System

IP address space consists of 2^{32} = 4294967296 addresses theoretically. This is rather a large number for a naming directory. Assigning mnemonic names for such a large namespace requires a naming system that is easy to use, scalable, and manageable. The Internet used flat namespace initially. Flat namespace is not scalable due to administrative reasons. Therefore, a hierarchical naming system is used.

A hierarchical naming system has vertical and horizontal partitions similar to that of an organization. Consider for example an educational institution UNIV (Figure 3.39). It has computer science (CS), telecommunication (TEL), and administration (ADM) departments. The computer science department has two divisions—software engineering (SW) and hardware engineering (HW). Each engineering division has a laboratory (LAB). If we use a naming scheme that reflects the lineage, a possible name for the software laboratory can be LAB.SW.CS. UNIV. The name LAB.SW.CS.UNIV is readily understood and unique. The naming scheme is scalable. If computer science department introduces another division, network engineering (NE), the naming scheme readily accommodates the change without having any impact on the rest of the existing name structure.

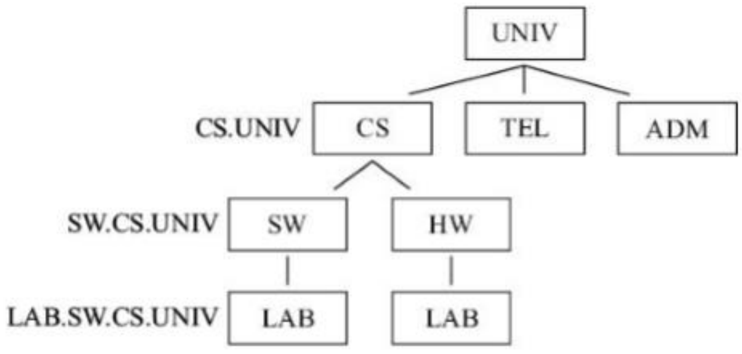


Figure 3.39: Hierarchical naming.

3.3.2.2 Internet Naming System

The Internet uses a hierarchical naming system called Domain Name System (DNS). Its tree like structure is similar to the example given above. It maps the IP address of the hosts to a human readable format. It is implemented as distributed database worldwide on DNS 'name servers'. When an IP host wants to get the IP address of a name, its client process communicates with the name servers to get the IP address.

Domain: It is a complete sub-tree under a particular point in the naming tree structure (Figure 3.40). Note the emphasis on naming tree structure. A domain has nothing to do with geographic distribution of hosts.

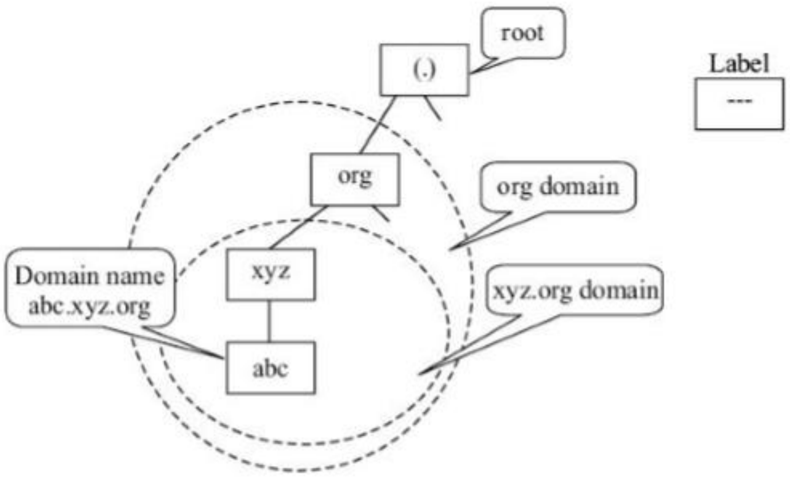


Figure 3.40: DNS terminology.

Label: It is a component of a domain name and identifies a local entity at a particular hierarchy level. For example, SW is the label assigned to the software engineering division of the computer science department. Note that SW in itself is not a fully qualified name. A label need not be unique across the tree. It needs only to be unique at the particular point in the tree. In the last example, LAB was used as a label both for software and hardware laboratories without any conflict in the names.

Domain name: It is the name given to a node in the naming tree. It consists of all the labels from the root to the node, listed from right to left and separated by dots (.). It has the following characteristics:

- A domain name may consist of maximum 255 characters.
- Domain names are not case sensitive. For example, www.nic.org is same as WWW. NIC.ORG.
- Due to SMTP restrictions, domain names can contain only the characters a to z (lower or upper case), numerals 1 to 9 and symbol (-).
- A Fully Qualified Domain Name (FQDN) is concatenation of all the labels up to the root of the naming tree.
- Hosts with multiple IP addresses can be assigned a single domain name.
- Hosts with single IP address can have multiple domain names depending on applications, e.g. the domain names can be www. abc, mail. abc, etc. to differentiate several services.

The domain name database consists of master files. A master file is associated with a domain and contains name-address mappings of lower domains and the hosts in the domain.

The root of the DNS tree is represented as a '.'. The root is implemented by several name servers at the highest level. The lower hierarchy levels are called top level, second level, and so on. The top level contains seven three-character generic domains as well as two-character country-specific domains:

1. edu (Education)
2. com (Commercial)
3. gov (US government)
4. mil (US military)

5. org (Organizations other than above)

6. int (International organizations)

7. net (Network providers)

8. Country specific, e.g. fr — France, in — India.

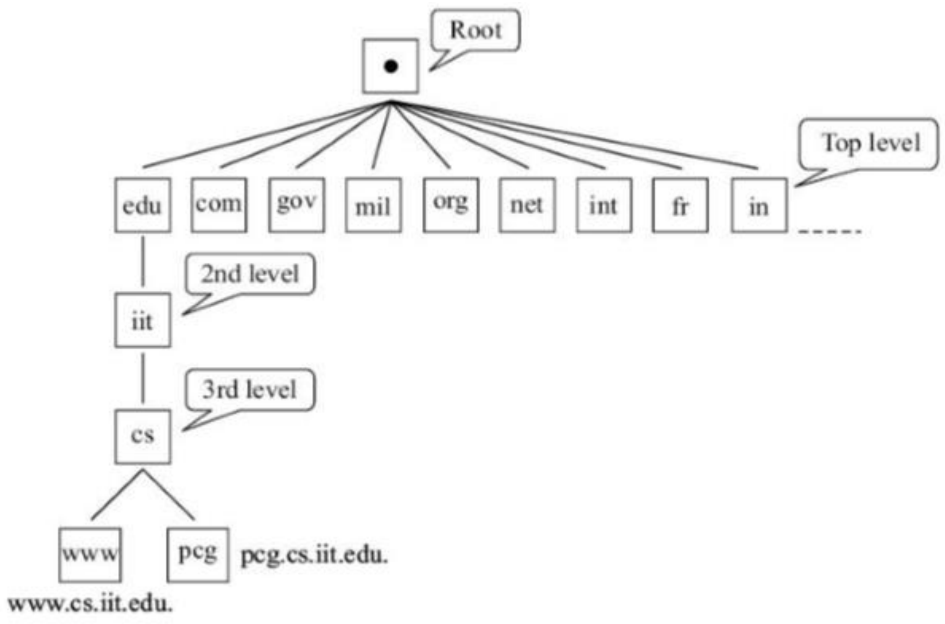


Figure 3.41: DNS naming system in the Internet.

Domain name registration is totally independent of IP address allotment. Domain names are registered with InterNIC (USA), RIPE (Europe), and APNIC (Asia).

3.3.2.3 DNS Protocol

DNS is based on the client-server model. DNS messages are of two types— queries (and replies) and zone transfers. Queries use UDP transport protocol and zone transfers use TCP transport protocol. DNS queries are limited to 512 bytes. The well-known port for the DNS server is 53. Basic structure of DNS message is depicted in Figure 3.42. It consists of several sections. Header is always present whilst other sections may be empty.

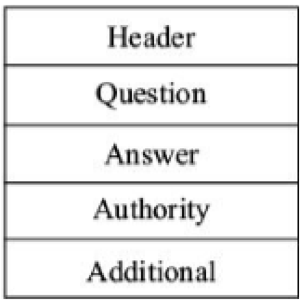


Figure 3.42: Structure of DNS message.

Header: It contains identifiers that associate queries with responses. It indicates the type of message (query or response), and the number of queries and records present in the message. There are several other fields.

Question: It contains the queries for the DNS server. Answer. It contains the Resource Records (RR) that answer the queries.

Authority: It contains the list of authorities (name servers) that serve a particular domain. **Additional**: This section contains additional information that does not form part of the sections above.

Figure 3.43 shows a typical exchange of DNS messages to find IP address of the name www.abc.xyz.org.

1. The client in the local host has pre-configured address of the default name server (NS). It sends the query to the default server.

2. The default server sends query to the root server which gives list of org name servers.

3. Default NS sends query to a org server which gives the list of xyz.org name servers.
4. Finally, xyz.org server gives the required IP address to the default name server which sends it to the local host.

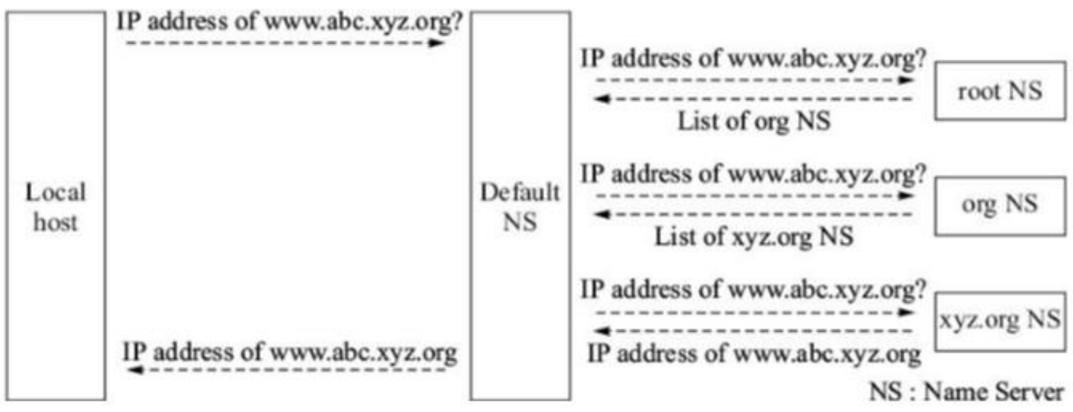

Local host
IP address of www.abc.xyz.org?
IP address of www.abc.xyz.org
Default NS
IP address of www.abc.xyz.org?
List of org NS
root NS
IP address of www.abc.xyz.org?
List of xyz.org NS
org NS
IP address of www.abc.xyz.org?
IP address of www.abc.xyz.org
xyz.org NS
NS : Name Server

Figure 3.43: Typical exchange of DNS messages.

3.3.3 HYPERTEXT TRANSFER PROTOCOL (HTTP)/HTTPS

HTTP is used for transporting WWW documents between the client (Web browser) and server (Web server). HTTP is described in RFCs (1945, 2616 and 2817), the current version being used is version 1.1.

HTTP runs on the well-known TCP port 80 for the server (Figure 3.44). The basic operation consists of three steps:

1. The client opens a TCP connection and sends request for a document.

2. The server responds with the document. 3. The server closes the connection.

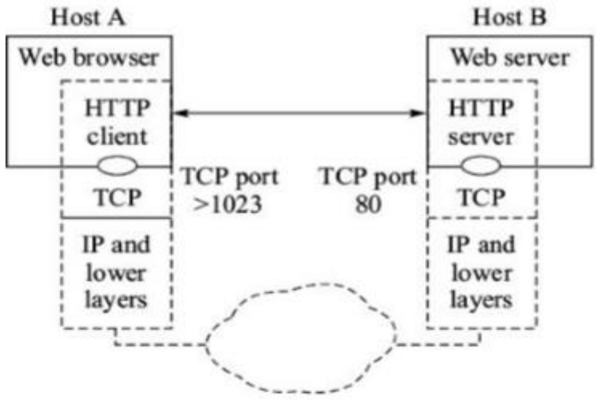


Figure 3.44: HTTP architecture.

HTTP messages from the client to the server are called HTTP request messages. HTTP messages from the server to the client are called response messages. These messages consist of a header and a body as described below. The body contains data described by a MIME header. The data can be HTML document (Web page), graphic, video or sound.

There are several types HTTP-Methods. The important ones are described below:

GET: GET is a request that allows the client to retrieve the information as identified in the URL. The requested information is returned in the body of the response message of the server. GET request consists of the header only.

HEAD: This request is same as GET except that the server's response consists only of the header as if body were present. This enables the client to get the resource information without transferring the actual body information. HEAD request consists of the header only.

POST: POST request consists of a header and a body. It is a request to the server to accept the attached entity in the body of the request as a new subordinate to the identified URL.

PUT: It is a request to store the attached entity in the body of the request under the supplied URL. It can be a new URL or an existing URL. In the later case the entity replaces the existing contents of the URL. Note the difference between POST and PUT. In POST request the new contents becomes subordinate, and it does not replace the existing content.

DELETE: DELETE requests the server to delete the resource identified by the URL.

TRACE: TRACE allows the client to see how the message was received and retrieved at the other end. The server returns whatever it receives in the body of the TRACE request from the client. It acts like an application layer loop-back. It is used for diagnostic purposes.

COPY: It is a request to copy the resource identified by the URL in the request line to the locations indicated in the URL-Header field that follows the request line of the message.

MOVE: It requests the server to move the resource identified by the URL in the request line to the locations given in the URL-Header field.

3.3.4 FILE TRANSFER PROTOCOL (FTP)

The way files containing information are stored in a system depends on the architecture of the system which includes the hardware, the operating system, data type, coding styles, file organization, and the access methods. To exchange these files between two different systems requires a common protocol that is acceptable to the two systems. There are two approaches for exchange of the files:

- Virtual files
- Reduction approach

In virtual file approach, (just like NVT), a virtual file structure is defined. The real files are translated to the virtual files by the one end system. The virtual file is transmitted to the other end where they are translated to the local file system. This approach is quite complex because all the possible representations and file structures must be considered for implementing translators from the real to the virtual file system. An example of virtual file system protocols is FTAM (File Transfer, Access, and Management) protocol of ISO.

The second approach is to reduce all the file systems to a file system having a common minimal set of fundamental properties. Only views and operations as defined by the common set of properties are possible. No translation is carried out as in case of virtual files. File Transfer Protocol (FTP) is based on this concept. FTP enables transfer of a copy of a file from one end system to another. The original remains unchanged and in the original location.

The basic features of FTP are:

- **Data representation**: FTP handles three types of data representations— ASCII (7 bit), EBCDIC (8-bit), and 8-bit binary data.
- **File organization**: FTP supports unstructured and structured files. An unstructured file contains string of bytes, end-marked by EOF (End of File). A structured file contains a list of records and each record is delimited by EOR (End of Record). EOF and EOR are represented as a sequence of two bytes, 0xFF + 0x01 (EOR), 0xFF + 0x02 (EOF) and 0xFF + 0x03 (EOR + EOF).
- **Transmission mode**: In stream mode, the file is transmitted as continuous bit stream without any modification. EOF and EOR, where applicable, are inserted as the two byte sequence. In block mode, the data is divided into blocks with EOR after every block and EOF at the end of the file. In compressed mode, a sequence of same characters is transmitted only once with a replication counter that enables the receiver to restore the compressed data.
- **Error control**: Since TCP is used for data transfer no additional error recovery mechanism is required.
- **Access control**: File access protection is done using login procedure with login name and password.

3.3.5 TRIVIAL FILE TRANSFER PROTOCOL (TFTP)

TFTP is suited for applications that do not require rather complex procedures of FTP (File Transfer Protocol) and do not have enough resources (RAM, ROM) for this purpose. For example, FTP requires multiple concurrent session which may not be possible on small diskless machines or network devices such as bridges, routers, etc. The code size of TFTP is very small and can be easily fit into bootstrap-ROMs of such machines and devices. Typical applications of TFTP include loading the image on diskless machine and upgrading the operating system in network devices such as routers. The main features of TFTP are:

- TFTP is based on client-server principle and uses well-known UDP port number 69 for the TFTP server.
- TFTP is unsecured protocol and does not support authentication.
- TFTP incorporates idle-RQ (stop and wait) error recovery mechanism.
  – Every TFTP data unit bears a sequence number.
  – Each data unit is individually acknowledged. After receiving the acknowledgement the next data unit is sent.
  – Error recovery is by retransmission after timeout. TFTP uses adaptive timeout with exponential back-off algorithm.

TFTP Message Formats

There are four types of TFTP messages: Read request (Type 1). This command is used by the client to get a copy of a file from the server. Write request (Type 2). This command is used by the client to write a file into the server. Data (Type 3). This TFTP message contains blocks of data (portions of the file being copied). Acknowledgement (Type 4). This is used by the client and the server to acknowledge the received data units. Figure 3.45 shows the format of these messages. The first two octets indicate the type of message. Mode field defines the type of data (ASCII, binary, mail). The file-name and mode fields are delimited using an all zeroes octet. Type 3 message contains the data blocks of fixed size of 512 octets. The session is terminated if a data message arrives with data octets less than 512 octets. The last data message can have data block with EOF having size less than 512 octets. Type 4 message is used for acknowledgement.

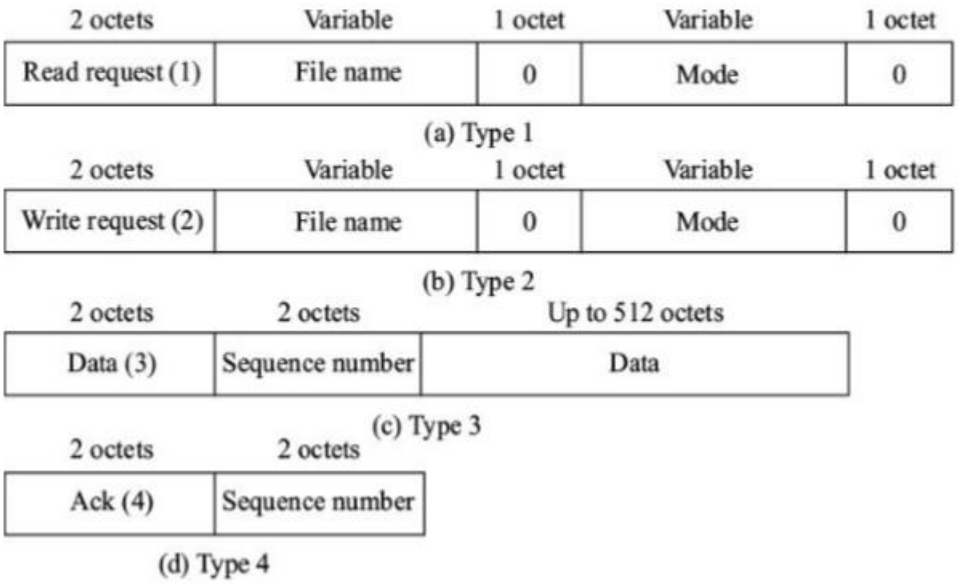


Figure 3.45: Types of TFTP messages.

3.3.6 Secure File Transfer Protocol (SFTP)

Secure File Transfer Protocol (SFTP) is a secure version of File Transfer Protocol (FTP), which facilitates data access and data transfer over a Secure Shell (SSH) data stream. It is part of the SSH Protocol. This term is also known as SSH File Transfer Protocol. SFTP was designed by the Internet Engineering Task Force (IETF) as an extended version of SSH 2.0, allowing file transfer over SSH and use with Transport Layer Security (TLS) and VPN applications. Both the commands and data are encrypted in order to prevent passwords and other sensitive information from being transferred over the network. The functionality of SFTP is similar to that of FTP. However, SFTP uses SSH to transfer files. SFTP requires that the client user must be authenticated by the server and the data transfer must take place over a secure channel (SSH).

3.3.7 Telnet

The main task of the Internet is to let users have access to remote hosts running different applications. One possible way is to write client-server program for each application. But this is not a practical solution. Alternative is to have a general-purpose client-server program that lets a user access any application running on a remote host. Telnet is a general purpose client-server application that achieves this objective.

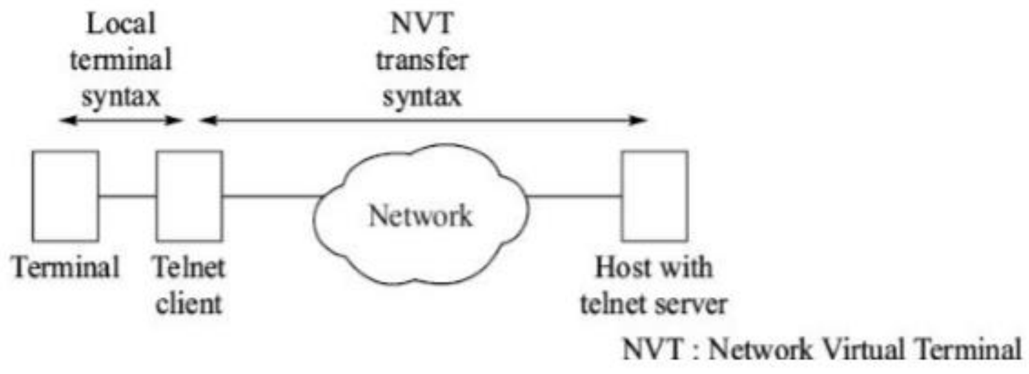


Figure 3.46: Telnet client and server.

Telnet is a connection-oriented protocol and uses TCP at the transport layer. The telnet server is connected to the well-known TCP port 23. The telnet client can be on any TCP port > 1023.

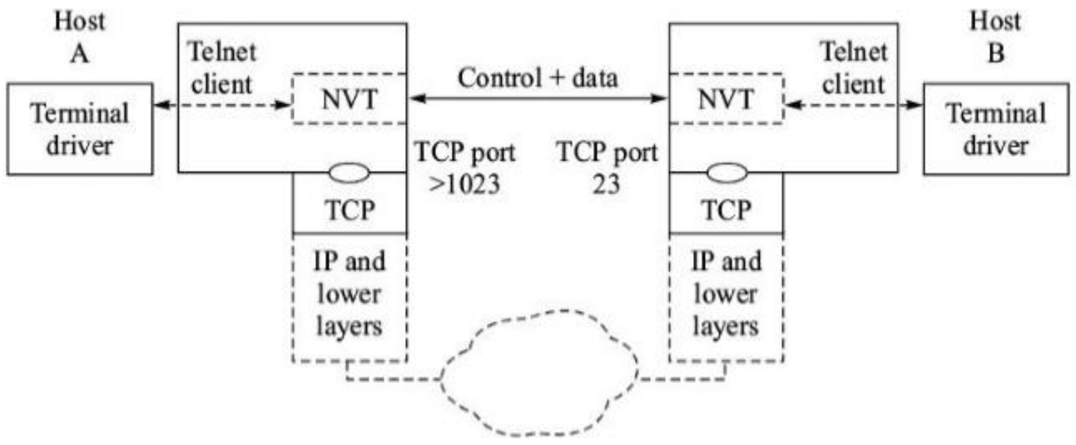


Figure 3.47: Telnet architecture.

A telnet client can emulate the behaviour of a wide range of well-known terminals. Internally the telnet client communicates with the telnet server through a canonical terminal representation called Network Virtual Terminal (NVT) as shown in Figure 3.47. Thus, the local terminal device characteristics are mapped to NVT capabilities. The user's keystrokes received by the local terminal driver are sent to the telnet client which transforms these to a universal character set of the NVT. The process is reversed at the other end.

3.3.8 Email: SMTP

Mail Format

Mail format is defined by RFC 822 with extension MIME. It consists of two parts—an envelope or header and a body. Both parts are represented in ASCII. The body is assumed to be simple text, although it may be encoded image or voice/video clip or data.

Header: The header contains information necessary for transmission and delivery of the mail. It also includes the sender's identification. It consists of a series of lines each terminated with a pair of ASCII characters, CR (carriage return) and LF (line feed). Each line has a type field followed by a colon and the value. The header starts with FROM in the first line. For example, FROM: prakash <prakash@xyz.com> identifies the sender. The header contains information about the sender, receiver, date-time, subject, etc.

Body: The body of the mail is separated from the header by a blank line. It contains user's message. RFC 822 restricts the maximum size of the body to 1000 characters. But MIME extends it further in certain cases. The body always contains character codes from ASCII. The body is followed by a signature separated from the body by two dashes '--'. It contains personal information of the sender, keys, etc.

Simple Mail Transfer Protocol (SMTP)

SMTP is based on client-server model and it uses well-known TCP port number 25 for the server. All the commands, responses, and messages are in ASCII format (printable characters with binary values from 33 to 126, CR and LF).

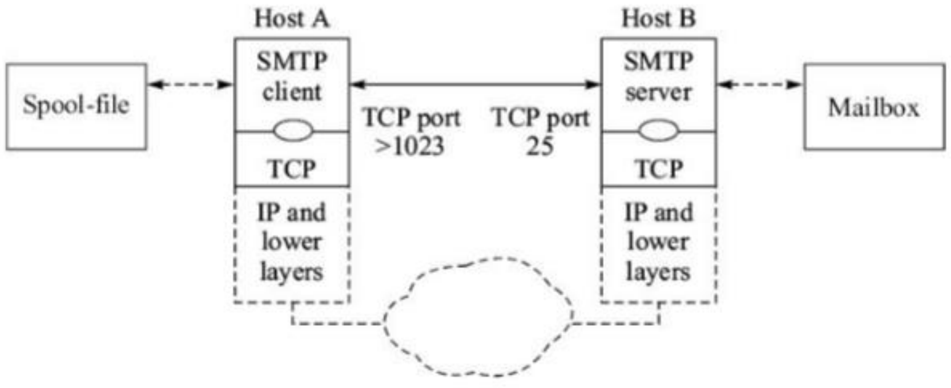


Figure 3.48: SMTP architecture.

3.3.9 Post Office Protocol (POP3) and Internet Message Access Protocol (IMAP4)

Very often a user writes and reads his e-mails on local machine and has his mailbox on a server that runs SMTP for receiving e-mails (and probably SMTP client for sending e-mails). The server is permanently connected to the Internet. Post Office Protocol, version 3 (POP3) and IMAP are the protocols that let a user fetch his e-mail from its remote mailbox on the server. The user's machine and the mail server run the client and server processes respectively. POP3 is described in RFC 1939 and it runs on well-known TCP port 110. The communication procedure is similar to SMTP and uses ASCII characters. IMAP is described in RFC 2060. It is more sophisticated than POP3 and allows a client to access and manipulate e-mails and mailboxes. It allows:

- a client to create, delete, and rename mailboxes,
- a client to selectively fetch message attributes such as all, body, envelope or flags,
- searching the mailbox for given match criteria,
- maintaining several flags such as seen, answered, draft, and deleted.

IMAP runs on the well-known TCP port 143.

3.3.10 NTP

Network Time Protocol (NTP) is an application layer protocol used for clock synchronization between hosts on a TCP/IP network. The goal of NTP is to ensure that all computers on a network agree on the time, since even a small difference can create problems. For example, if there is more than 5 minutes difference on your host and the Active Directory domain controller, we will not be able to login into our AD domain.

Network Time Protocol (NTP) is a protocol used to synchronize computer clock times in a network. It belongs to and is one of the oldest parts of the TCP/IP protocol suite. The term NTP applies to both the protocol and the client-server programs that run on computers.

Working of NTP

The NTP client initiates a time-request exchange with the NTP server. As a result of this exchange, the client is able to calculate the link delay and its local offset, and adjust its local clock to match the clock at the server's computer. As a rule, six exchanges over a period of about five to 10 minutes are required to initially set the clock.

Once synchronized, the client updates the clock about once every 10 minutes, usually requiring only a single message exchange. In addition to client-server synchronization. This transaction occurs via the User Datagram Protocol on port 123. NTP also supports broadcast synchronization of peer computer clocks.