

1.1 Introduction

- The network layer is placed between data link layer and transport layer in the TCP/IP network reference model.
- It is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links) i.e. it is responsible for the delivery of individual packets from the source host to the destination host.
- The responsibilities of the network layer are explained below.
 - Logical Addressing:
 - The network layer assigns a unique IP address to each host on the network. This address is used to identify the host and route data packets to it.
 - Routing:
 - The network layer determines the best path for data packets to take from their source to their destination. This is done by using routing protocols, which exchange information about the network topology.
 - Fragmentation and reassembly:
 - The network layer may need to fragment large data packets into smaller packets so that they can be transmitted over a particular link. The packets are then reassembled at the destination host.
 - Error detection and correction:
 - The network layer can detect errors in data packets and request that they be retransmitted. This is done by using checksums, which are calculated over the data packets.
 - Flow control:
 - The network layer can control the flow of data between hosts to prevent one host from overwhelming another. This is done by using flow control mechanisms, such as windowing.
- The different protocols defined at this layer are briefly described below.
 - 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.
 - Some other protocols such as Internet Protocol version 6 (IPv6) are used to overcome the limitations of the IPv4 and Reverse Address Resolution Protocol (RARP) is used to map hardware address to IP address.

1.2 IPv4 Addresses

- The identifier used in the network layer (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.
- An IPv4 address is unique to a single connection to the Internet. A device with two connections to the Internet, via two networks, will have two IPv4 addresses.
- All hosts that want to connect to the Internet must use the IPv4 addressing system.

1.2.1 Address Space

- A protocol like IPv4 that defines addresses has an address space.
- An address space is the total number of addresses that can be used by a protocol.
- The number of bits used to define an address determines the size of the address space.
- For example, IPv4 uses 32 bits to define an address, so its address space is 2^{32} , which is equal to 4,294,967,296 addresses. This means that, in theory, more than 4 billion devices could be connected to the Internet using IPv4 addresses.

Notations

- There are three common notations to represent an IPv4 address:
 - binary notation (base 2)
 - dotted-decimal notation (base 256)
 - hexadecimal notation (base 16)
- Binary notation is the most basic way to represent an IPv4 address. It is a 32-bit number, which can be written as a string of 8 binary digits (bits). Each bit can have a value of 0 or 1. 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.
- Dotted-decimal notation is a more human-readable way to represent an IPv4 address. It is a string of four decimal numbers separated by periods (dot ●). Note that because each byte (octet) is only 8 bits, each number in the dotted-decimal notation is between 0 and 255.
- 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 – 1 shows an IP address in the three discussed notations.

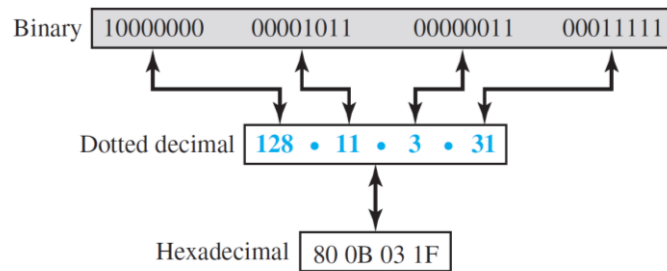


Figure-1 Three different notations in IPv4 addressing.

Example – 1

Change the following IPv4 addresses from binary notation to dotted-decimal notation.

- a. 10000001 00001011 00001011 11101111
- b. 11000001 10000011 00011011 11111111.

Solution

We replace each group of 8 bits with its equivalent decimal numbers and add dots for separation.

- a. 129.11.11.239
- b. 193.131.27.255

Example – 2

Change the following IPv4 addresses from dotted-decimal notations to binary notations.

- a. 111.56.45.78
- b. 221.34.7.82

Solution

We replace each decimal number with its binary equivalent

- a. 01101111 00111000 00101101 01001110
- b. 11011101 00100010 00000111 01010010

Example – 3

Find the error, if any, in the following IPv4 addresses.

- a. 111.56.045.78
- b. 221.34.7.8.20
- c. 75.45.301.14
- d. 11100010.23.14.67

Solution

- a. There must be no leading zero (045)

- b. There can be no more than four numbers in an IPv4 address.
- c. Each number needs to be less than or equal to 255 (301 is outside the range).
- d. A mixture of binary notation and dotted-decimal notation is not allowed.

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.
- A telephone number is divided into the country code, area code and the connection.
- Similarly, a 32-bit IPv4 address is also hierarchical, but divided only into two parts.
 - The first part of the address, called the prefix (Net ID), defines the network.
 - the second part of the address, called the suffix (Host ID), defines the node (connection of a device to the Internet).
- Figure – 2 shows the prefix and suffix of a 32-bit IPv4 address.

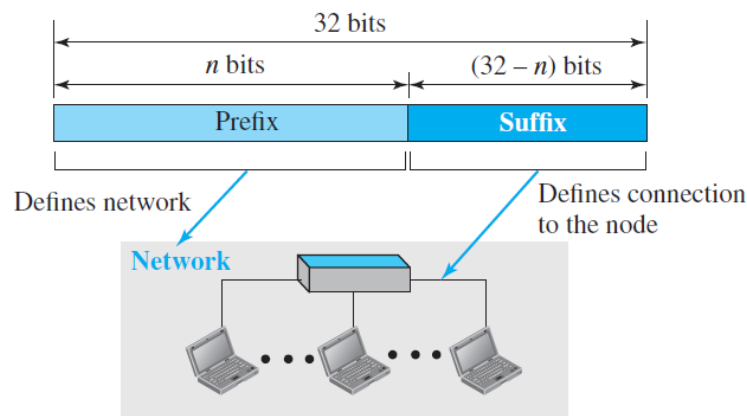


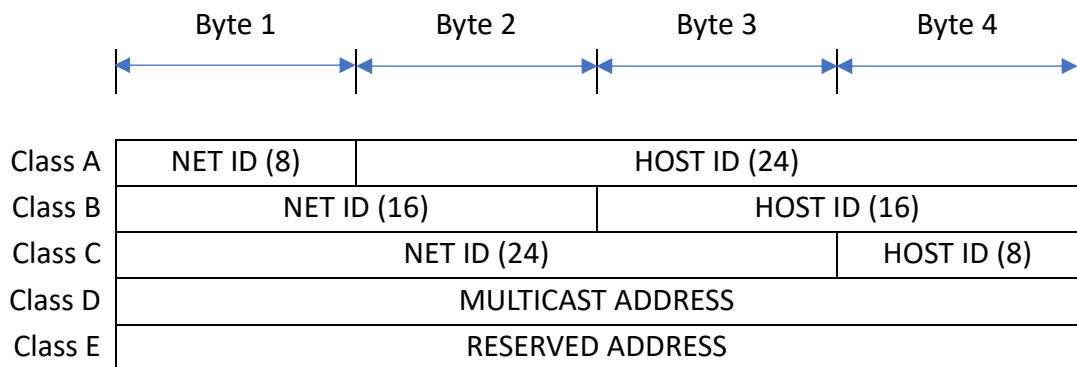
Figure-2 Hierarchy in addressing.

- The prefix length is n bits, and the suffix length is $(32 - n)$ bits. 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.

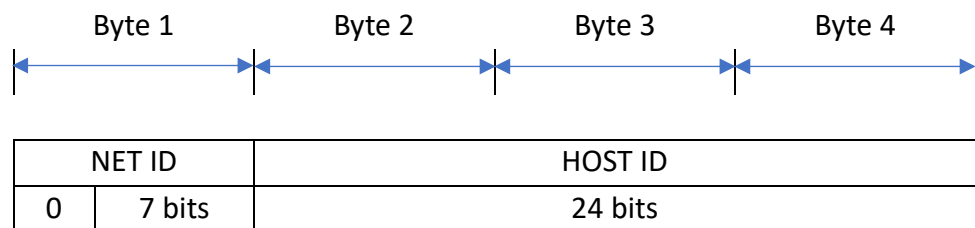
1.2.2 Classful Addressing

- Classful addressing is an IPv4 addressing architecture that divides the address space into five classes: A, B, C, D, and E. The first three classes (A, B, and C) are used for network hosts, while class D is reserved for multicasting and class E is reserved for future use. Each class occupies some part of the address space.
- The order of bits in the first octet of the IPv4 address determines the classes of the IP address.
- The IPv4 address is divided into two parts:

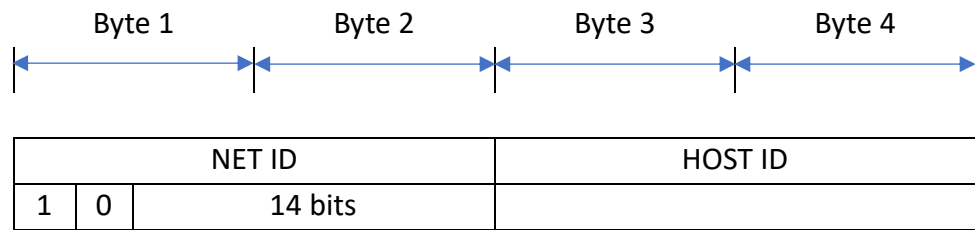
- Net ID (Prefix)
- Host ID (Suffix)
- The network ID identifies the network that the device is connected to, while the host ID identifies the specific device on the network.
- The class of IP address is used to determine the bits used for network ID, the bits used for host ID, the number of total networks and hosts possible in that class.
- Net ID and Host ID bits for each class is shown below.



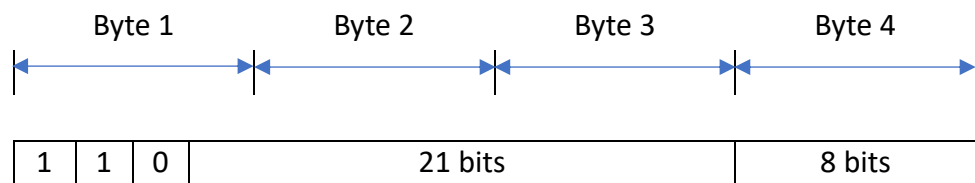
- Class A
 - IP addresses belonging to class A are assigned to the networks that contain a large number of hosts.



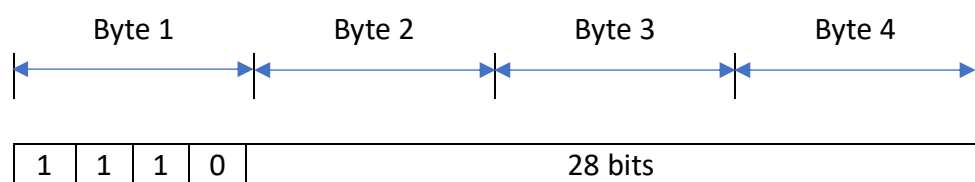
- The Net ID is 8 bits long.
- The Host ID is 24 bits long.
- The higher-order bit of the first octet in class A is always set to 0. The remaining 7 bits in the first octet are used to determine network ID. The 24 bits of host ID are used to determine the host within the network.
- The default subnet mask for Class A is 255.0.0.0.
- class A has a total of
 - $2^7 - 2 = 126$ network ID (Here 2 address is subtracted because 0.0.0.0 and 127.255.255.255 are special address.)
 - $2^{24} - 2 = 16,777,214$ host ID
- IP addresses belonging to class A ranges from 1.0.0.0 – 126.255.255.255
- Class B
 - IP address belonging to class B is assigned to networks that range from medium-sized to large-sized networks.



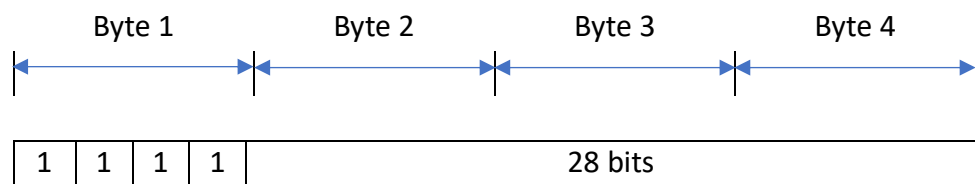
- The Net ID is 16 bits long.
- The Host ID is 16 bits long.
- The higher-order bits of the first octet of IP addresses of class B are always set to 10. The remaining 14 bits are used to determine the network ID. The 16 bits of host ID are used to determine the host in any network.
- The default subnet mask for class B is 255.255.0.0.
- Class B has a total of:
 - $2^{14} = 16384$ network address
 - $2^{16} - 2 = 65534$ host address
- IP addresses belonging to class B ranges from 128.0.0.0 – 191.255.255.255.
- Class C
 - IP addresses belonging to class C are assigned to small-sized networks.



- The Net ID is 24 bits long.
- The Host ID is 8 bits long.
- The higher-order bits of the first octet of IP addresses of class C are always set to 110. The remaining 21 bits are used to determine the network ID. The 8 bits of host ID are used to determine the host in any network.
- The default subnet mask for class C is 255.255.255.0.
- Class C has a total of:
 - $2^{21} = 2097152$ network address
 - $2^8 - 2 = 254$ host address
- IP addresses belonging to class C range from 192.0.0.0 – 223.255.255.255.
- Class D
 - IP address belonging to class D is reserved for multi-casting.



- The higher-order bits of the first octet of IP addresses belonging to class D are always set to 1110. The remaining bits are for the address that interested hosts recognize.
- Class D does not possess any subnet mask.
- IP addresses belonging to class D range from 224.0.0.0 – 239.255.255.255.
- Class E
 - IP addresses belonging to class E are reserved for experimental and research purposes.



- The higher-order bits of the first octet of IP addresses belonging to class E are always set to 1111.
- Class E does not possess any subnet mask.
- IP addresses of class E range from 240.0.0.0 – 255.255.255.254.
- Rules for Assigning HOST ID
 - Host IDs are used to identify a host within a network. The host ID is assigned based on the following rules:
 - Within any network, the host ID must be unique to that network.
 - A host ID in which all bits are set to 0 cannot be assigned because this host ID is used to represent the network ID of the IP address.
 - Host ID in which all bits are set to 1 cannot be assigned because this host ID is reserved as a broadcast address to send packets to all the hosts present on that particular network.
- Rules for Assigning Network ID
 - Hosts that are located on the same physical network are identified by the network ID, as all host on the same physical network is assigned the same network ID. The network ID is assigned based on the following rules:
 - The network ID cannot start with 127 because 127 belongs to the class A address and is reserved for internal loopback functions.
 - All bits of network ID set to 1 are reserved for use as an IP broadcast address and therefore, cannot be used.
 - All bits of network ID set to 0 are used to denote a specific host on the local network and are not routed and therefore, aren't used.
- Summary of Classful Addressing is shown in the Table – 1

Unit-1 Network Layer Protocols (4350706)

Class	Leading Bit	Net ID Bits	Host ID Bits	No. of Networks	Host Addresses Per Network	Start Address	End Address
A	0	8	24	2^7	2^{24}	0.0.0.0	127.255.255.255
B	10	16	16	2^{14}	2^{16}	128.0.0.0	191.255.255.255
C	110	24	8	2^{21}	2^8	192.0.0.0	223.255.255.255
D	1110	Not Defined	Not Defined	Not Defined	Not Defined	224.0.0.0	239.255.255.255
E	1111	Not Defined	Not Defined	Not Defined	Not Defined	240.0.0.0	255.255.255.255

Table – 1 Summary of Classful Addressing

Example – 4

Find the class of each address.

- 00000001 00001011 00001011 11101111
- 11000001 10000011 00011011 11111111
- 14.23.120.8
- 252.5.15.111

Solution

- The first bit is 0. This is class A address.
- The first 2 bits are 1; the third bit is 0. This is class C address.
- The first byte is 14 (between 0 and 127). This is a class A address.
- The first byte is 252 (between 240 and 255). This is a class E address.

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.
- To alleviate address depletion, two strategies were proposed and, to some extent, implemented: subnetting and supernetting.

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 (Net ID) for each class is fixed, we can find the prefix length (Net ID) immediately.
- In other words, the prefix length (Net ID) in classful addressing is inherent in the address; no extra information is needed to extract the prefix Net ID) and the suffix (Host ID).

1.2.3 Classless Addressing

- Subnetting and Supernetting in classful addressing were not enough to solve the address depletion problem. As the Internet grew, it became clear that a larger address space was needed as a long-term solution. However, this would require increasing the length of IP addresses, which would also require changing the format of IP packets.
- The long-term solution to address depletion is IPv6, which has a much larger address space than IPv4. However, IPv6 is not yet widely deployed, so a short-term solution was needed. This solution, called classless addressing, uses the same address space as IPv4 but distributes the addresses more efficiently. Classless addressing does not assign addresses based on classes, but instead allows organizations to get the exact amount of address space they need.
- In other words, classless addressing removes the "class privilege" from the distribution of addresses. This helps to compensate for the address depletion problem and allows more organizations to get the addresses they need.
- In the 1990s, Internet Service Providers (ISPs) became increasingly common. ISPs provide Internet access to individuals, small businesses, and midsize organizations that do not want to create their own Internet sites or provide Internet services to their employees. ISPs are granted a large range of addresses from the Internet authorities, and then subdivide these addresses into smaller blocks for their customers.
- Classful addressing, which was the previous method of assigning IPv4 addresses, was not well-suited for this type of arrangement. Classful addressing divides the IPv4 address space into five classes, each with a fixed number of addresses. This meant that ISPs could only be assigned Class A, B, or C addresses, which were not always the right size for their needs.
- To address this problem, the Internet authorities announced a new architecture called classless addressing in 1996. Classless addressing allows ISPs to be assigned variable-

length blocks of addresses. This means that ISPs can get the exact amount of address space they need, regardless of their size.

- In classless addressing, the whole address space is divided into variable length blocks.
- The prefix in an address defines the block (network); the suffix defines the node (device).
- Theoretically, we can have a block of 2^0 , 2^1 , 2^2 , . . . , 2^{32} addresses. One of the restrictions 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 shows the division of the whole address space into nonoverlapping blocks.

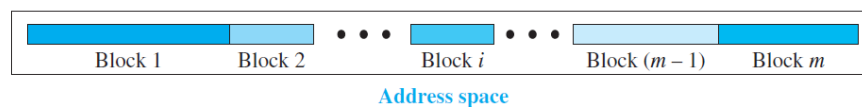


Figure – 3 Variable-length blocks in classless addressing

- 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 – 4.

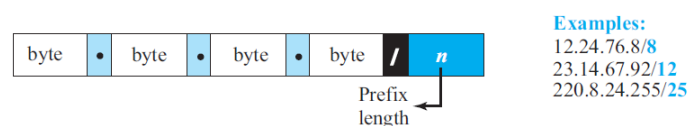


Figure – 4 Slash notation (CIDR)

- 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 18.21.
 - The number of addresses in the block is found as $N = 2^{32-n}$.
 - To find the first address, we keep the n leftmost bits and set the $(32 - n)$ rightmost bits all to 0s.
 - To find the last address, we keep the n leftmost bits and set the $(32 - n)$ rightmost bits all to 1s.

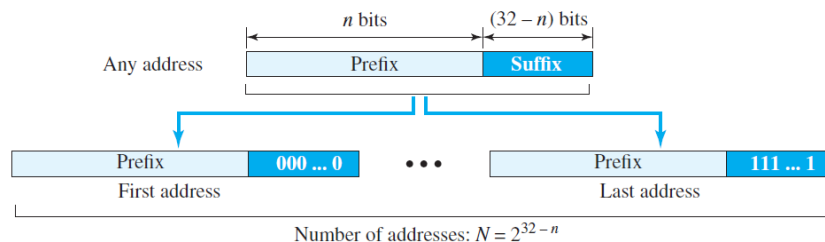


Figure-5 Information extraction in classless addressing

Example – 5

A classless address is given as **167.199.170.82/27**. Find the number of addresses, the first and last address of the network.

Solution

- The number of addresses in the network is $2^{32-n} = 2^{32-27} = 2^5 = 32$ addresses.
- The first address can be found by keeping the first 27 bits of the address as it is and changing the rest of the bits to 0s.

Address: 167.199.170.82/27	10100111 11000111 10101010 01010010
The first Address: 167.199.170.64/27	10100111 11000111 10101010 01000000

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

Address: 167.199.170.82/27	10100111 11000111 10101010 01010010
The first 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 $N = \text{NOT}(\text{mask}) + 1$.
 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))].

Example – 6

A classless address is given as 167.199.170.82/27. Find the address mask, number of addresses, the first and last address of the network.

Solution**1. Address Mask**

11111111 11111111 11111111 11100000	The address given here has a prefix length of 27. Now to find the address mask from it first 27 bits should be set to 1 and remaining 5 bits should be set to 0.
255.255.255.224	Dotted decimal notation of binary notation is written here

2. Number of addresses in the block

To find how many addresses are possible in a block, first perform a logical NOT operation on the address mask and then add the value 1 to it.	
Logical NOT (11111111 11111111 11111111 11000000) + 1	
(00000000 00000000 00000000 00011111) + 1 = 31 + 1	
32 addresses.	

3. The first address in the block

The bitwise AND operation of a given IP address and its subnet mask yields the network address, which is the first address of that block.	
Address: 167.199.170.82/27 Subnet Mask: 255.255.255.224	(10100111 11000111 10101010 01010010) Logical AND (11111111 11111111 11111111 11100000)
The first Address: 167.199.170.64/27	10100111 11000111 10101010 01000000

4. The last address in the block

The bitwise OR operation of a given IP address and the logical NOT of its subnet mask yields the last possible address in that block.	
Address: 167.199.170.82/27 Subnet Mask: 255.255.255.224	(10100111 11000111 10101010 01010010) Logical OR (Logical NOT(11111111 11111111 11111111 11100000))
Address: 167.199.170.82/27 Subnet Mask: 255.255.255.224	(10100111 11000111 10101010 01010010) Logical OR (00000000 00000000 00000000 00011111)
Address: 167.199.170.95/27	10100111 11000111 10101010 01011111

Example – 7

In classless addressing, an address cannot per se define the block the address belongs to. For example, the address 230.8.24.56 can belong to many blocks. Some of them are shown below with the value of the prefix associated with that block.

Prefix Length	Block
/16	230.8.0.0 to 230.8.255.255
/20	230.8.16.0 to 230.8.31.255
/26	230.8.24.0 to 230.8.24.63
/27	230.8.24.32 to 230.8.24.63
/29	230.8.24.56 to 230.8.24.63
/31	230.8.24.56 to 230.8.24.57

Network Address

- The above examples show that, given any address, we can find all information about the block.
- 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.
- Figure – 6 shows the idea.

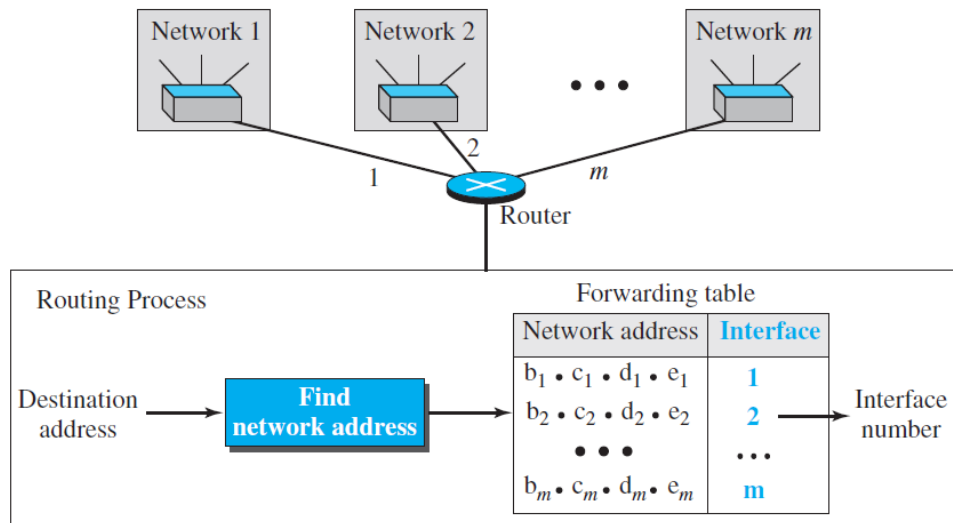


Figure – 6 Network address

- 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.

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 $N = 2^{32-n}$ or $n = 32 - \log_2 N$. 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

$$\text{first address} = (\text{prefix in decimal}) \times 2^{32-n} = (\text{prefix in decimal}) \times N.$$

Example – 8

An ISP has requested a block of 1000 addresses. Since 1000 is not a power of 2, 1024 addresses are granted. The prefix length is calculated as $n = 32 - \log_2 1024 = 22$. An available block, 18.14.12.0/22, is granted to the ISP. It can be seen that the first address in decimal is 302,910,464, which is divisible by 1024.

Subnetting

- Subnetting is the process of dividing a single IP network into smaller subnetworks, or subnets.
- 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
 - a. The subnetworks in a network should be carefully designed to enable the routing of packets.
 - b. 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.
 1. The number of addresses in each subnetwork should be a power of 2.
 2. The prefix length for each subnetwork should be found using the following formula:
$$n_{\text{sub}} = 32 - \log_2 N_{\text{sub}}$$
 3. 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
 - a. 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.

Example – 9

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 – 7 shows the configuration of blocks. We have shown the first address in each block.

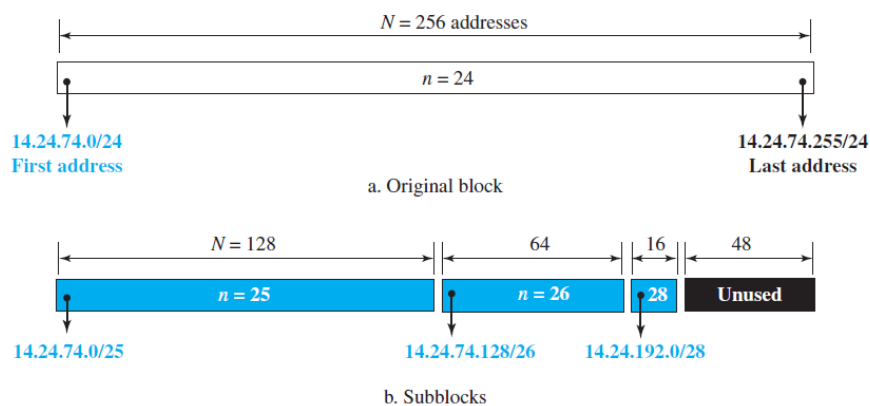


Figure – 7 Configuration of blocks

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.

Example – 10

- Figure – 7 shows how four small blocks of addresses are assigned to four organizations by an ISP. The ISP combines these four blocks into one single block and advertises the larger block to the rest of the world.
- Any packet destined for this larger block should be sent to this ISP.
- It is the responsibility of the ISP to forward the packet to the appropriate organization. This is similar to Forwarding of IP packets routing we can find in a postal network. All packages coming from outside a country are sent first to the capital and then distributed to the corresponding destination.

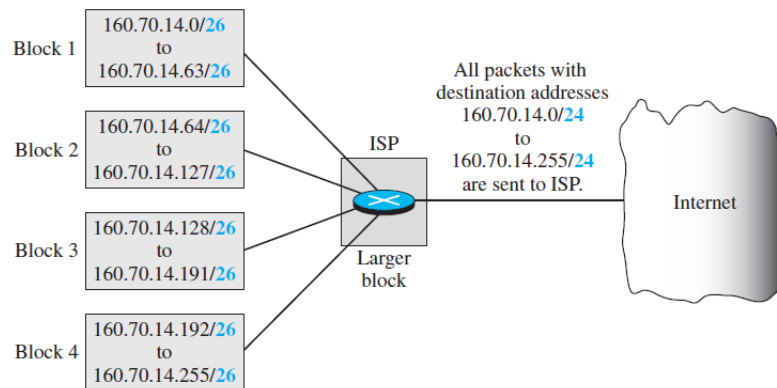


Figure – 7 Example of address aggregation

Special Addresses

- Before finishing the topic of addresses in IPv4, we need to mention five special addresses that are used for special purposes:
 - this-host address,
 - limited-broadcast address,
 - loopback address,
 - private addresses,
 - and multicast addresses.
- This-host Address
 - The only address in the block 0.0.0.0/32 is called the this-host address.
 - It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address.
 - It is used in DHCP.
- Limited-broadcast Address
 - The only address in the block 255.255.255.255/32 is called the limited-broadcast address.
 - It is used whenever a router or a host needs to send a datagram to all devices in a network.
 - The routers in the network, however, block the packet having this address as the destination; the packet cannot travel outside the network.

- Loopback Address
 - The block 127.0.0.0/8 is called the loopback address.
 - A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host.
 - Any address in the block is used to test a piece of software in the machine.
 - For example, we can write a client and a server program in which one of the addresses in the block is used as the server address. We can test the programs using the same host to see if they work before running them on different computers.
- Private Addresses
 - Four blocks are assigned as private addresses: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, and 169.254.0.0/16.
- Multicast Addresses
 - The block 224.0.0.0/4 is reserved for multicast addresses.

1.2.4 Network Address Translation

- The distribution of addresses through ISPs has created a new problem.
 - a. *Assume that an ISP has granted a small range of addresses to a small business or a household. If the business grows or the household needs a larger range, the ISP may not be able to grant the demand because the addresses before and after the range may have already been allocated to other networks.*
- In most situations, however, only a portion of computers in a small network need access to the Internet simultaneously. This means that the number of allocated addresses does not have to match the number of computers in the network.
 - a. *For example, assume that in a small business with 20 computers the maximum number of computers that access the Internet simultaneously is only 4. Most of the computers are either doing some task that does not need Internet access or communicating with each other. This small business can use the TCP/IP protocol for both internal and universal communication. The business can use 20 (or 25) addresses from the private block addresses (discussed before) for internal communication; five addresses for universal communication can be assigned by the ISP.*
- A technology that can provide the mapping between the private and universal addresses is known as Network Address Translation (NAT).
- The technology allows a site to use a set of private addresses for internal communication and a set of global Internet addresses (at least one) for communication with the rest of the world.
- The site must have only one connection to the global Internet through a NAT-capable router that runs NAT software.
- Figure – 8 shows a simple implementation of NAT.

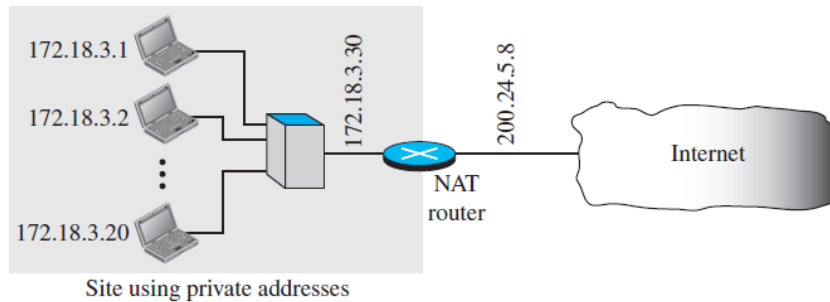


Figure – 8 NAT

- As the figure shows, the private network uses private addresses. The router that connects the network to the global address uses one private address and one global address. The private network is invisible to the rest of the Internet; the rest of the Internet sees only the NAT router with the address 200.24.5.8.

Address Translation

- All of the outgoing packets go through the NAT router, which replaces the source address in the packet with the global NAT address.
- All incoming packets also pass through the NAT router, which replaces the destination address in the packet (the NAT router global address) with the appropriate private address.
- Figure – 9 shows an example of address translation.

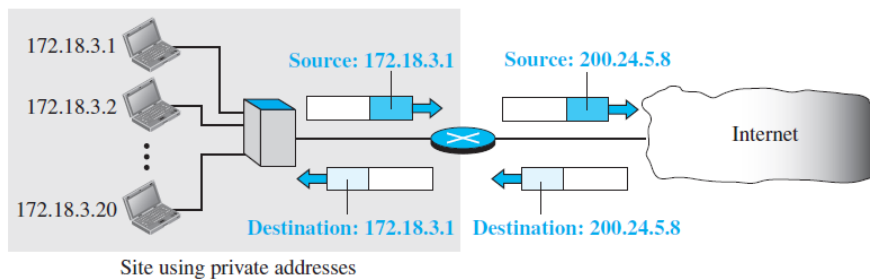


Figure – 9 Address translation

Translation Table

- The reader may have noticed that translating the source addresses for an outgoing packet is straightforward. But how does the NAT router know the destination address for a packet coming from the Internet?
- There may be tens or hundreds of private IP addresses, each belonging to one specific host. The problem is solved if the NAT router has a translation table.

Using One IP Address

- In its simplest form, a translation table has only two columns: the private address and the external address (destination address of the packet).

- When the router translates the source address of the outgoing packet, it also makes note of the destination address— where the packet is going.
- When the response comes back from the destination, the router uses the source address of the packet (as the external address) to find the private address of the packet.
- Figure – 10 shows the idea.

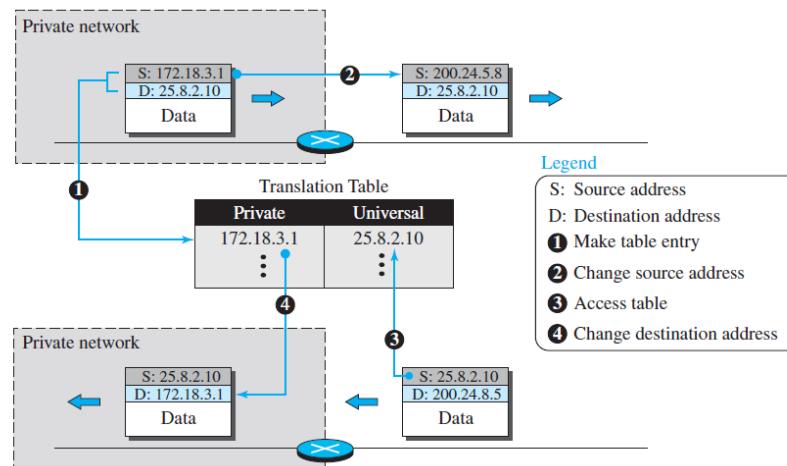


Figure – 10 Translation

- In this strategy, communication must always be initiated by the private network.
- The NAT mechanism described requires that the private network start the communication.
- As we will see, NAT is used mostly by ISPs that assign a single address to a customer. The customer, however, may be a member of a private network that has many private addresses.
- In this case, communication with the Internet is always initiated from the customer site, using a client program such as HTTP, TELNET, or FTP to access the corresponding server program.
- For example, when e-mail that originates from outside the network site is received by the ISP e-mail server, it is stored in the mailbox of the customer until retrieved with a protocol such as POP.

Using a Pool of IP Addresses

- The use of only one global address by the NAT router allows only one private-network host to access a given external host.
- To remove this restriction, the NAT router can use a pool of global addresses. For example, instead of using only one global address (200.24.5.8), the NAT router can use four addresses (200.24.5.8, 200.24.5.9, 200.24.5.10, and 200.24.5.11).
- In this case, four private-network hosts can communicate with the same external host at the same time because each pair of addresses defines a separate connection.
- However, there are still some drawbacks. No more than four connections can be made to the same destination. No private-network host can access two external server

programs (e.g., HTTP and TELNET) at the same time. And, likewise, two private-network hosts cannot access the same external server program (e.g., HTTP or TELNET) at the same time.

Using Both IP Addresses and Port Addresses

- To allow a many-to-many relationship between private-network hosts and external server programs, we need more information in the translation table.
- For example, suppose two hosts inside a private network with addresses 172.18.3.1 and 172.18.3.2 need to access the HTTP server on external host 25.8.3.2. If the translation table has five columns, instead of two, that include the source and destination port addresses and the transport-layer protocol, the ambiguity is eliminated.
- Table – 2 shows an example of such a table.

<i>Private address</i>	<i>Private port</i>	<i>External address</i>	<i>External port</i>	<i>Transport protocol</i>
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
⋮	⋮	⋮	⋮	⋮

Table -2 Five-column translation table

- Note that when the response from HTTP comes back, the combination of source address (25.8.3.2) and destination port address (1401) defines the private network host to which the response should be directed. Note also that for this translation to work, the ephemeral port addresses (1400 and 1401) must be unique.

1.3 Forwarding of IP packets

- A forwarding means to place the packet in its route to its destination.
- Since the Internet today is made of a combination of links (networks), forwarding means to deliver the packet to the next hop (which can be the final destination or the intermediate connecting device).
- Although the IP protocol was originally designed as a connectionless protocol, today the tendency is to change it to a connection-oriented protocol.
- We discuss both cases. When IP is used as a connectionless protocol, forwarding is based on the destination address of the IP datagram; when the IP is used as a connection-oriented protocol, forwarding is based on the label attached to an IP datagram.
- We will discuss only forwarding based on the destination address here.

1.3.1. Forwarding based on Destination Address

- This is a traditional approach, which is prevalent today.
- In this case, forwarding requires a host or a router to have a forwarding table.
- When a host has a packet to send or when a router has received a packet to be forwarded, it looks at this table to find the next hop to deliver the packet to.
- In classless addressing, the whole address space is one entity; there are no classes. This means that forwarding requires one row of information for each block involved. The table needs to be searched based on the network address (first address in the block). Unfortunately, the destination address in the packet gives no clue about the network address. To solve the problem, we need to include the mask (/n) in the table.
- In other words, a classless forwarding table needs to include four pieces of information: the mask, the network address, the interface number, and the IP address of the next router.
- However, we often see in the literature that the first two pieces are combined. For example, if n is 26 and the network address is 180.70.65.192, then one can combine the two as one piece of information: 180.70.65.192/26.
- Figure – 11 shows a simple forwarding module and forwarding table for a router with only three interfaces.

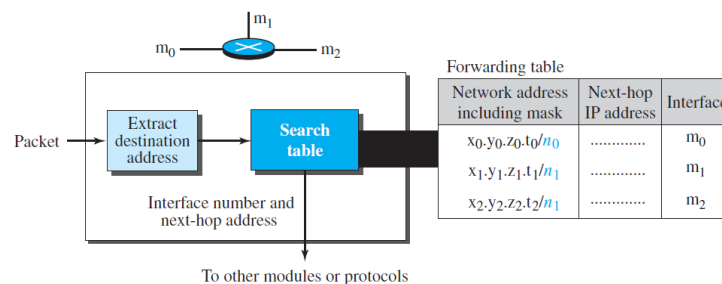


Figure – 11 Simplified forwarding module in classless address

- The job of the forwarding module is to search the table, row by row. In each row, the n leftmost bits of the destination address (prefix) are kept and the rest of the bits (suffix) are set to 0s. If the resulting address (which we call the network address), matches with the address in the first column, the information in the next two columns is extracted; otherwise, the search continues.
- Normally, the last row has a default value in the first column (not shown in the figure), which indicates all destination addresses that did not match the previous rows.
- Sometimes, the literature explicitly shows the value of the n leftmost bits that should be matched with the n leftmost bits of the destination address. The concept is the same, but the presentation is different. For example, instead of giving the address-mask combination of 180.70.65.192/26, we can give the value of the 26 leftmost bits as shown below. 10110100 01000110 01000001 11 Note that we still need to use an algorithm to find the prefix and compare it with the bit pattern. In other words, the algorithm is still needed, but the presentation is different.

Example - 11

Make a forwarding table for router R1 using the configuration in Figure 18.33.

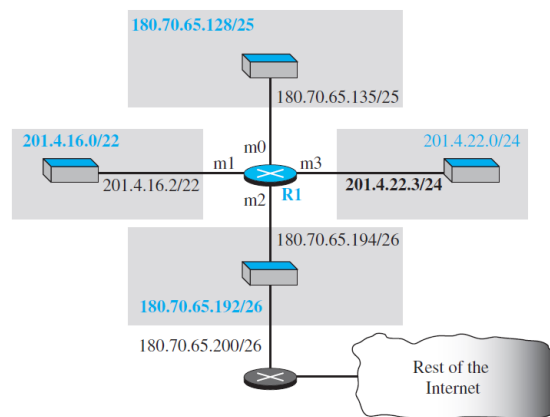


Figure – 12 Configuration for Example 18.7

Network address/mask	Next hop	Interface
180.70.65.192/26	-----	m2
180.70.65.128/25	-----	m0
201.4.22.0/24	-----	m3
201.4.16.0/22	-----	m1
Default	180.70.65.200/26	m2

Table – 4 Forwarding Table in dotted-decimal notation

Leftmost bits in the destination address	Next hop	Interface
10110100 01000110 01000001 11	-----	m2
10110100 01000110 01000001 1	-----	m0
11001001 00000100 00011100	-----	m3
11001001 00000100 000100	-----	m1
Default	180.70.65.200/26	m2

Table – 5 Forwarding table in binary notation

- When a packet arrives whose leftmost 26 bits in the destination address match the bits in the first row, the packet is sent out from interface m2.
- When a packet arrives whose leftmost 25 bits in the address match the bits in the second row, the packet is sent out from interface m0, and so on.
- The table clearly shows that the first row has the longest prefix and the fourth row has the shortest prefix.
- The longer prefix means a smaller range of addresses; the shorter prefix means a larger range of addresses.

Example - 12

Show the forwarding process if a packet arrives at R1 in Figure – 12 with the destination address 180.70.65.140.

Solution

The router performs the following steps:

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

Address Aggregation

- When we use classful addressing, there is only one entry in the forwarding table for each site outside the organization. The entry defines the site even if that site is subnetted. When a packet arrives at the router, the router checks the corresponding entry and forwards the packet accordingly.
- When we use classless addressing, it is likely that the number of forwarding table entries will increase. This is because the intent of classless addressing is to divide up the whole address space into manageable blocks. The increased size of the table results in an increase in the amount of time needed to search the table.
- To alleviate the problem, the idea of address aggregation was designed.
- In Figure – 13 we have two routers. R1 is connected to networks of four organizations that each use 64 addresses. R2 is somewhere far from R1. R1 has a longer forwarding table because each packet must be correctly routed to the appropriate organization. R2, on the other hand, can have a very small forwarding table. For R2, any packet with destination 140.24.7.0 to 140.24.7.255 is sent out from interface m0 regardless of the organization number. This is called address aggregation because the blocks of addresses for four organizations are aggregated into one larger block. R2 would have a longer forwarding table if each organization had addresses that could not be aggregated into one block.

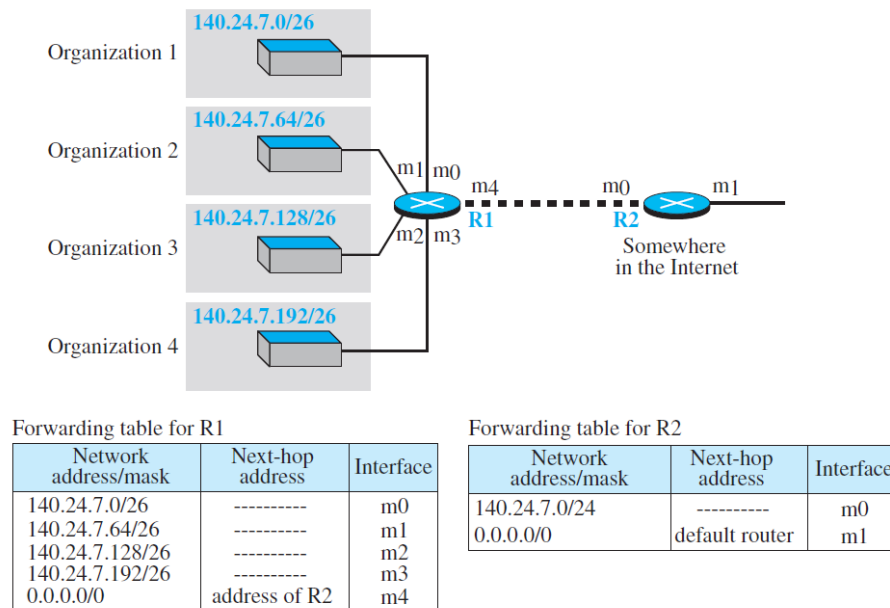


Figure – 13 Address aggregation

Longest Match Making

- What happens if one of the organizations in the previous figure is not geographically close to the other three?
- For example, if organization 4 cannot be connected to router R1 for some reason, can we still use the idea of address aggregation and still assign block 140.24.7.192/26 to organization 4?
- The answer is yes, because routing in classless addressing uses another principle, longest mask matching.
- This principle states that the forwarding table is sorted from the longest mask to the shortest mask. In other words, if there are three masks, /27, /26, and /24, the mask /27 must be the first entry and /24 must be the last.
- Let us see if this principle solves the situation in which organization 4 is separated from the other three organizations.
- Figure – 14 shows the situation. Suppose a packet arrives at router R2 for organization 4 with destination address 140.24.7.200. The first mask at router R2 is applied, which gives the network address 140.24.7.192. The packet is routed correctly from interface m1 and reaches organization 4. If, however, the forwarding table was not stored with the longest prefix first, applying the /24 mask would result in the incorrect routing of the packet to router R1.

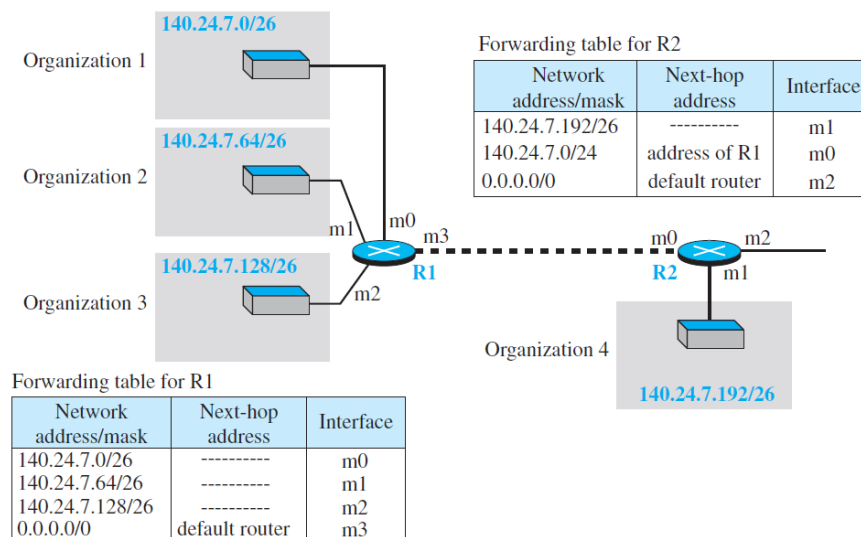


Figure – 14 Longest mask matching

Hierarchical Routing

- To solve the problem of gigantic forwarding tables, we can create a sense of hierarchy in the forwarding tables.
- The Internet today has a sense of hierarchy. That the Internet is divided into backbone and national ISPs. National ISPs are divided into regional ISPs, and regional ISPs are divided into local ISPs.
- If the forwarding table has a sense of hierarchy like the Internet architecture, the forwarding table can decrease in size.
- Let us take the case of a local ISP. A local ISP can be assigned a single, but large, block of addresses with a certain prefix length. The local ISP can divide this block into smaller blocks of different sizes, and assign these to individual users and organizations, both large and small.
- If the block assigned to the local ISP starts with a.b.c.d/n, the ISP can create blocks starting with e.f.g.h/m, where m may vary for each customer and is greater than n. How does this reduce the size of the forwarding table?
- The rest of the Internet does not have to be aware of this division. All customers of the local ISP are defined as a.b.c.d/n to the rest of the Internet.
- Every packet destined for one of the addresses in this large block is routed to the local ISP.
- There is only one entry in every router in the world for all of these customers. They all belong to the same group. Of course, inside the local ISP, the router must recognize the subblocks and route the packet to the destined customer. If one of the customers is a large organization, it also can create another level of hierarchy by subnetting and dividing its subblock into smaller subblocks (or sub-subblocks).
- In classless routing, the levels of hierarchy are unlimited as long as we follow the rules of classless addressing.

Example – 13

As an example of hierarchical routing, let us consider Figure – 15. A regional ISP is granted 16,384 addresses starting from 120.14.64.0. The regional ISP has decided to divide this block into four subblocks, each with 4096 addresses. Three of these subblocks are assigned to three local ISPs; the second subblock is reserved for future use. Note that the mask for each block is /20 because the original block with mask /18 is divided into 4 blocks.

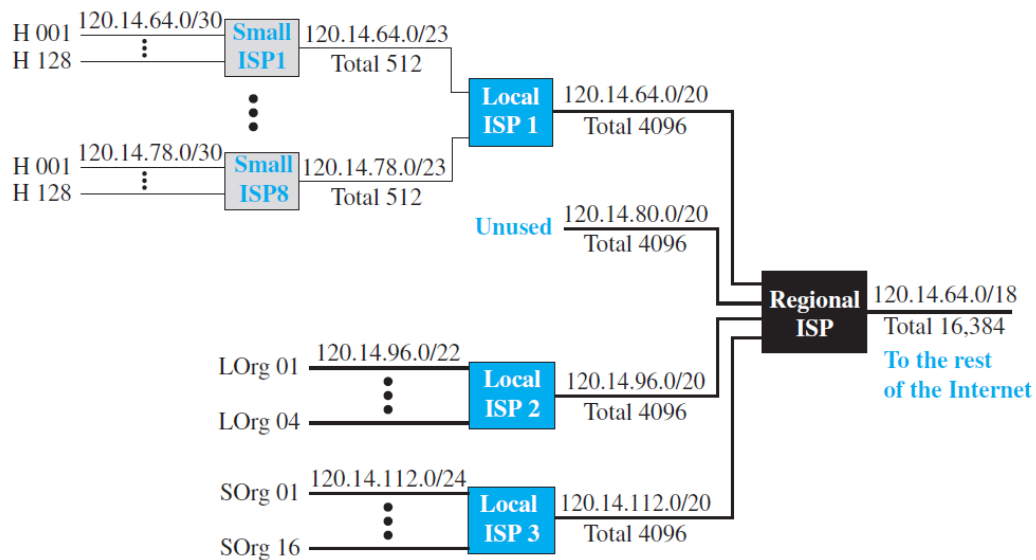


Figure – 15 Hierarchical routing with ISPs

- The first local ISP has divided its assigned subblock into 8 smaller blocks and assigned each to a small ISP. Each small ISP provides services to 128 households (H001 to H128), each using four addresses.
- Note that the mask for each small ISP is now /23 because the block is further divided into 8 blocks.
- Each household has a mask of /30, because a household has only 4 addresses ($2^{32} - 30 = 4$).
- The second local ISP has divided its block into 4 blocks and has assigned the addresses to 4 large organizations (LOrg01 to LOrg04). Note that each large organization has 1024 addresses and the mask is /22.
- The third local ISP has divided its block into 16 blocks and assigned each block to a small organization (SOrg01 to SOrg16). Each small organization has 256 addresses and the mask is /24.
- There is a sense of hierarchy in this configuration.
 - All routers in the Internet send a packet with destination address 120.14.64.0 to 120.14.127.255 to the regional ISP.
 - The regional ISP sends every packet with destination address 120.14.64.0 to 120.14.79.255 to Local ISP1.

- Local ISP1 sends every packet with destination address 120.14.64.0 to 120.14.64.3 to H001.

Geographical Routing

- To decrease the size of the forwarding table even further, we need to extend hierarchical routing to include geographical routing.
- We must divide the entire address space into a few large blocks.
- We assign a block to America, a block to Europe, a block to Asia, a block to Africa, and so on.
- The routers of ISPs outside of Europe will have only one entry for packets to Europe in their forwarding tables.
- The routers of ISPs outside of America will have only one entry for packets to America in their forwarding tables, and so on.

Forwarding Table Search Algorithm

- In classless addressing, there is no network information in the destination address.
- The simplest, but not the most efficient, search method is called the longest prefix match (as we discussed before).
- The forwarding table can be divided into buckets, one for each prefix.
- The router first tries the longest prefix. If the destination address is found in this bucket, the search is complete. If the address is not found, the next prefix is searched, and so on.
- It is obvious that this type of search takes a long time. One solution is to change the data structure used for searching. Instead of a list, other data structures (such as a tree or a binary tree) can be used. One candidate is a tree (a special kind of tree). However, this discussion is beyond the scope of this book.

1.4 Internet Protocol version 4

- 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 – 16 shows the positions of these four protocols in the TCP/IP protocol suite.

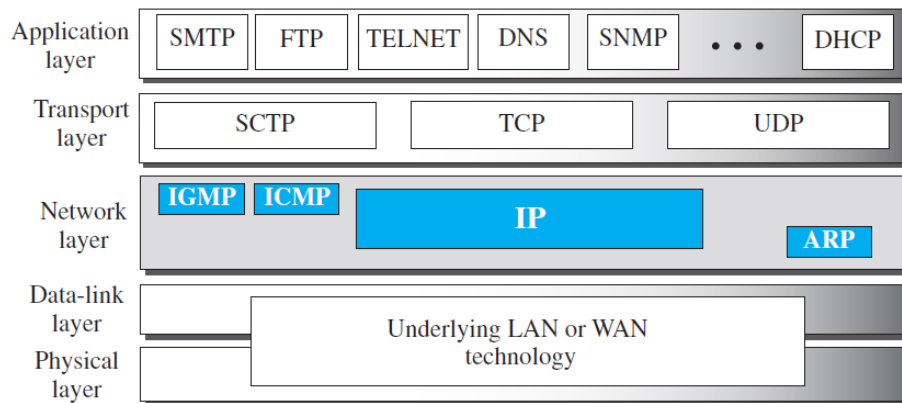


Figure – 16 Position of IP and other network-layer protocols in TCP/IP protocol suite

- 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.

1.4.1 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 - 17 shows the IPv4 datagram format.

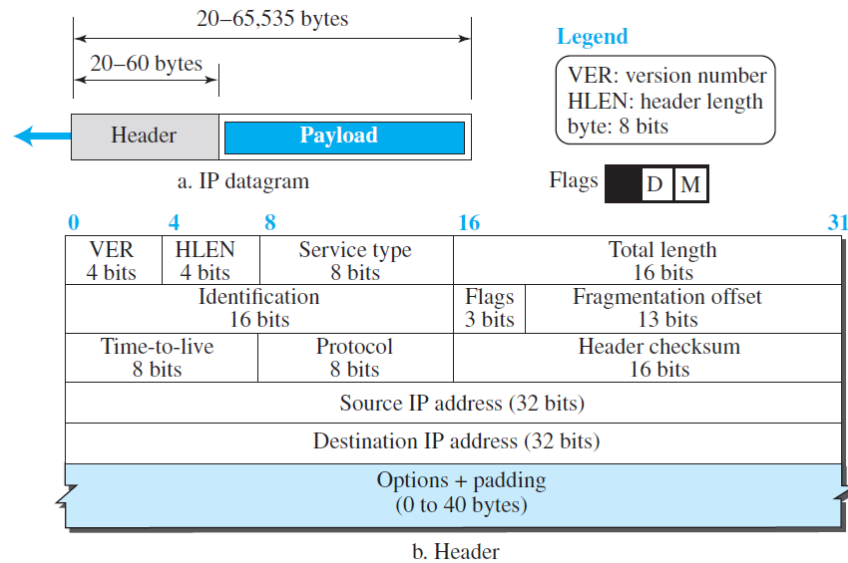
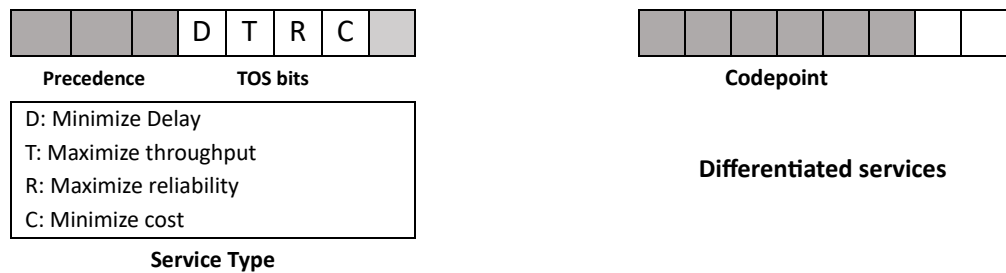


Figure – 17 IP Datagram

- 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 byte sections.
- 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.
 - This field tells the IPv4 software running in the processing machine that datagram has the format of version 4. All fields must be interpreted as specified in the fourth version of the protocol.
- **Header Length.**
 - This 4-bit header length (HLEN) field defines the total length of the datagram header in 4-byte words.
 - This field is needed because the length of the header is variable (between 20 bytes and 60 bytes).
 - When there are no options, the header length is 20 bytes, and the value of this field is 5 ($5 \times 4 = 20$).
 - When the option field is as its maximum size, the value of this field is 15 ($15 \times 4 = 60$).
- **Service Type.**
 - IETF has changed the interpretation and name of this 8-bit field. This field, previously called service type, is now called differentiated services.
 - Both interpretations are shown in the figure below.



1. Service Type

- In this interpretation, the first 3 bits are called precedence bits. The next 4 bits are called type of service (TOS) bits, and the last bit is not used.

a. Precedence

- It is a 3-bit subfield ranging from 0 (000 in binary) to 7 (111 binary).
- The precedence defines the priority of the datagram in issues such as congestion.
- If a router is congested and needs to discard some datagrams, those datagrams with lowest precedence are discarded first. Some datagrams in the Internet are more important than others. For example, a datagram used for network management is much more urgent and important than a datagram containing optional information for a group.

b. TOS bits

- It is a 4-bit subfield with each bit having a special meaning.
- Although a bit can be either 0 or 1, one and only one of the bits can have the value 1 in each datagram. With only 1 bit set at a time, we can have five different types of services.
- The bit patterns and their interpretations are given in the table below.

TOS bits	Description
0000	Normal (Default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimum delay

- Application programs can request a specific type of service. The default for some applications are shown in the table below.

Protocol	TOS bits	Description
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost

IGP	0010	Minimize cost
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Maximize throughput
SMTP (data)	0100	Maximize throughput
DNS (UDP Query)	1000	Minimize delay
DNS (TCP Query)	0000	Normal
DNS (zone)	0100	Maximize throughput

- It is clear from the above table that interactive activities, activities requiring immediate response need minimum delay.
- Those activities that send bulk data require maximum throughput.
- Management activities need maximum reliability.
- Background activities need minimum cost.

2. Differentiated Services

- In this interpretation, the first 6 bits make up the codepoint subfield, and the last 2 bits are not used. The subfield can be used in two different ways.
 - a. Three rightmost bits are 0s
 - When the 3 rightmost bits are 0s, the 3 leftmost bits are interpreted the same as the precedence bits in the service type interpretation.
 - In other words, it is compatible with the old interpretation.
 - b. Three rightmost bits are not all 0s.
 - When the 3 rightmost bits are not all 0s, the 6 bits define 64 services based on the priority assignment by the Internet or local authorities according to table shown below.

Category	Codepoint	Assigning Authority
1	XXXXX0	Internet
2	XXXX11	Local
3	XXXX01	Temporary Or Experimental

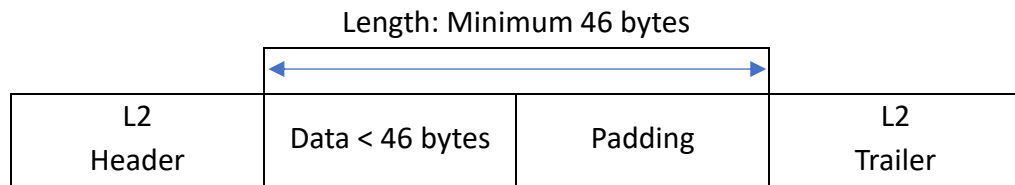
- The first category contains 32 service types; the second and third contain 16.
- The first category (numbers 0, 2, 4, ..., 62) is assigned by the Internet authorities (IETF).

- The second category (3,7,11,15,...,63) can be used by local authorities.
 - The third category(1,5,9,...,61) is temporary and can be used for experimental purposes.
 - Note that numbers are not contiguous. If they were, the first category would range from 0 to 31, the second category would range from 32 to 47, and the third category would range from 48 to 63.
 - This would be incompatible with the TOS interpretation because XXX000(which includes 0,8,16,24,32,40,48 and 56) would fall into all three categories. Instead in this assignment method all these services belong to category 1. Note that these assignments have not yet been finalized.
- **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.

$$\text{Length of data} = \text{total length} - \text{header length}$$

- Since the field length is 16 bits, the total length of the IPv4 datagram is limited 65,535($2^{16} - 1$) bytes, of which 20 to 60 bytes are the header and rest is data from the upper layer.
- 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 as shown the following figure.



- **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.**
 - A datagram has limited lifetime in its travel through an internet.
 - This field was originally designed to hold a timestamp, which was decremented by each visited router. The datagram was discarded when the value became zero.
 - However, for this scheme, all the machines must have synchronized clocks and must know how long it takes for a datagram to go from one machine to another.
 - Today, this field is used mostly 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.
 - This field is needed because routing tables in the Internet can become corrupted.
 - A datagram may travel between two or more routers for a long time without ever getting delivered to the destination host. This field limits the lifetime of a datagram.
 - Another use of this field is to intentionally limit the journey of the packet. For example, if the source wants to confine the packet to the local network, it can store 1 in this field. When a packet arrive at the first router, this value is decremented to 0, and the datagram is discarded.
- **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 – 18.
- Note that 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.

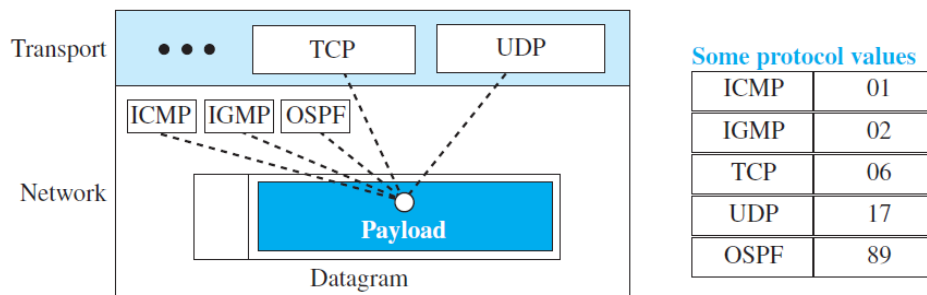


Figure – 18 Multiplexing and demultiplexing using the value of the protocol field.

- **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.

- The 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.
- **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.

Example – 14

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

Solution

- There is an error in this packet. The 4 leftmost bits $(0100)_2$ show the version, which is correct.
- The next 4 bits $(0010)_2$ show an invalid header length ($2 \times 4 = 8$). The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

Example – 15

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

Solution

- The HLEN value is 8, which means the total number of bytes in the header is 8×4 , or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options.

Example – 16

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

Solution

- The HLEN value is 5, which means the total number of bytes in the header is 5×4 , or 20 bytes (no options).
- The total length is $(0028)_{16}$ or 40 bytes, which means the packet is carrying 20 bytes of data ($40 - 20$).

Example – 17

An IPv4 packet has arrived with the first few hexadecimal digits as shown.

$(45000028000100000102...)_{16}$

How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution

- To find the time-to-live field, we skip 8 bytes (16 hexadecimal digits).
- The time-to-live field is the ninth byte, which is $(01)_{16}$. This means the packet can travel only one hop.
- The protocol field is the next byte $(02)_{16}$, which means that the upper-layer protocol is IGMP.

Example – 18

Figure – 19 shows an example of a checksum calculation for an IPv4 header without options. The header is divided into 16-bit sections. All the sections are added and the sum is complemented after wrapping the leftmost digit. The result is inserted in the checksum field.

Solution

4	5	0	28
49.153		0	0
4	17	0	↑
10.12.14.5			
12.6.7.9			
4, 5, and 0	→	4	5 0 0
28	→	0	0 1 C
1	→	C	0 0 1
0 and 0	→	0	0 0 0
4 and 17	→	0	4 1 1
0	→	0	0 0 0
10.12	→	0	A 0 C
14.5	→	0	E 0 5
12.6	→	0	C 0 6
7.9	→	0	7 0 9
Sum	→	1	3 4 4 E
Wrapped sum	→	3	4 4 F
Checksum	→	C	B B 0

Replaces 0

Figure – 19 Example of checksum calculation in IP

- Note that the calculation of wrapped sum and checksum can also be done as follows in hexadecimal:
 - Wrapped Sum = Sum mod FFFF
 - Checksum = FFFF – Wrapped Sum

1.4.2 Fragmentation

- A datagram can travel through different networks.
- Each router decapsulates the IP datagram from the frame it receives, processes it, and then encapsulates it in another frame.
- The format and size of the received frame depend on the protocol used by the physical network through which the frame has just travelled.
- The format and size of the sent frame depend on the protocol used by the physical network through which the frame is going to travel.
- For example, if a router connects a LAN to a WAN, it receives a frame in the LAN format and sends a frame in the WAN format.

Maximum Transfer Unit (MTU)

- Each link-layer protocol has its own frame format. One of the features of each format is the maximum size of the payload that can be encapsulated.
- In other words, when a datagram is encapsulated in a frame, the total size of the datagram must be less than this maximum size, which is defined by the restrictions imposed by the hardware and software used in the network (see Figure – 20).

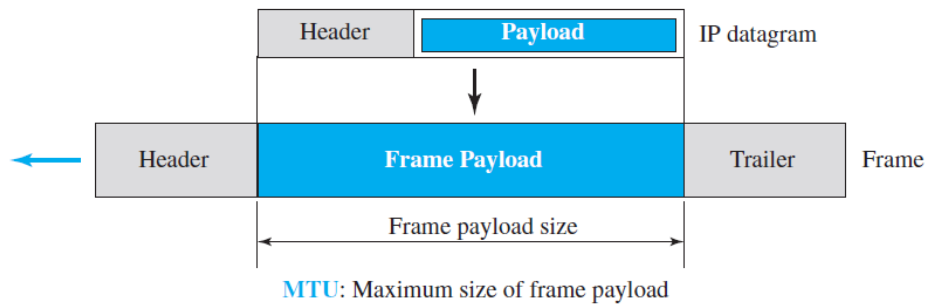


Figure – 20 Maximum transfer unit (MTU)

- The value of the MTU differs from one physical network protocol to another.
- For example, the value for a LAN is normally 1500 bytes, but for a WAN it can be larger or smaller.
- In order to make the IP protocol independent of the physical network, the designers decided to make the maximum length of the IP datagram equal to 65,535 bytes. This makes transmission more efficient if one day we use a link-layer protocol with an MTU of this size.
- However, for other physical networks, we must divide the datagram to make it possible for it to pass through these networks. This is called fragmentation.
- When a datagram is fragmented, each fragment has its own header with most of the fields repeated, but some have been changed.
- A fragmented datagram may itself be fragmented if it encounters a network with an even smaller MTU.
- In other words, a datagram may be fragmented several times before it reaches the final destination.
- A datagram can be fragmented by the source host or any router in the path. The reassembly of the datagram, however, is done only by the destination host, because each fragment becomes an independent datagram.
- Whereas the fragmented datagram can travel through different routes, and we can never control or guarantee which route a fragmented datagram may take, all of the fragments belonging to the same datagram should finally arrive at the destination host.
- So it is logical to do the reassembly at the final destination.
- An even stronger objection for reassembling packets during the transmission is the loss of efficiency it incurs.
- When we talk about fragmentation, we mean that the payload of the IP datagram is fragmented.
- However, most parts of the header, with the exception of some options, must be copied by all fragments.

- The host or router that fragments a datagram must change the values of three fields: flags, fragmentation offset, and total length. The rest of the fields must be copied. Of course, the value of the checksum must be recalculated regardless of fragmentation.

Fields Related to Fragmentation

- We mentioned before that three fields in an IP datagram are related to fragmentation:
 - a. identification,
 - b. flags,
 - c. and fragmentation offset.
- Let us explain these fields now. The 16-bit identification field identifies a datagram originating from the source host. The combination of the identification and source IP address must uniquely define a datagram as it leaves the source host.
- To guarantee uniqueness, the IP protocol uses a counter to label the datagrams. The counter is initialized to a positive number. When the IP protocol sends a datagram, it copies the current value of the counter to the identification field and increments the counter by one. As long as the counter is kept in the main memory, uniqueness is guaranteed.
- When a datagram is fragmented, the value in the identification field is copied into all fragments. In other words, all fragments have the same identification number, which is also the same as the original datagram.
- The identification number helps the destination in reassembling the datagram. It knows that all fragments having the same identification value should be assembled into one datagram.
- The 3-bit flags field defines three flags. The leftmost bit is reserved (not used). The second bit (D bit) is called the do not fragment bit. If its value is 1, the machine must not fragment the datagram. If it cannot pass the datagram through any available physical network, it discards the datagram and sends an ICMP error message to the source host. If its value is 0, the datagram can be fragmented if necessary. The third bit (M bit) is called the more fragment bit. If its value is 1, it means the datagram is not the last fragment; there are more fragments after this one. If its value is 0, it means this is the last or only fragment. The 13-bit fragmentation offset field shows the relative position of this fragment with respect to the whole datagram. It is the offset of the data in the original datagram measured in units of 8 bytes.
- Figure – 21 shows a datagram with a data size of 4000 bytes fragmented into three fragments. The bytes in the original datagram are numbered 0 to 3999. The first fragment carries bytes 0 to 1399. The offset for this datagram is $0/8 = 0$. The second fragment carries bytes 1400 to 2799; the offset value for this fragment is $1400/8 = 175$. Finally, the third fragment carries bytes 2800 to 3999. The offset value for this fragment is $2800/8 = 350$.

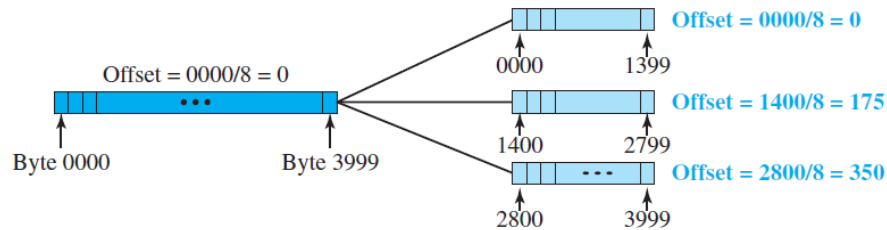


Figure – 21 Fragmentation example

- Remember that the value of the offset is measured in units of 8 bytes. This is done because the length of the offset field is only 13 bits long and cannot represent a sequence of bytes greater than 8191. This forces hosts or routers that fragment datagrams to choose the size of each fragment so that the first byte number is divisible by 8.
- Figure – 22 shows an expanded view of the fragments in the previous figure.
- The original packet starts at the client; the fragments are reassembled at the server.
- The value of the identification field is the same in all fragments, as is the value of the flags field with the more bit set for all fragments except the last. Also, the value of the offset field for each fragment is shown.
- Note that although the fragments arrived out of order at the destination, they can be correctly reassembled.

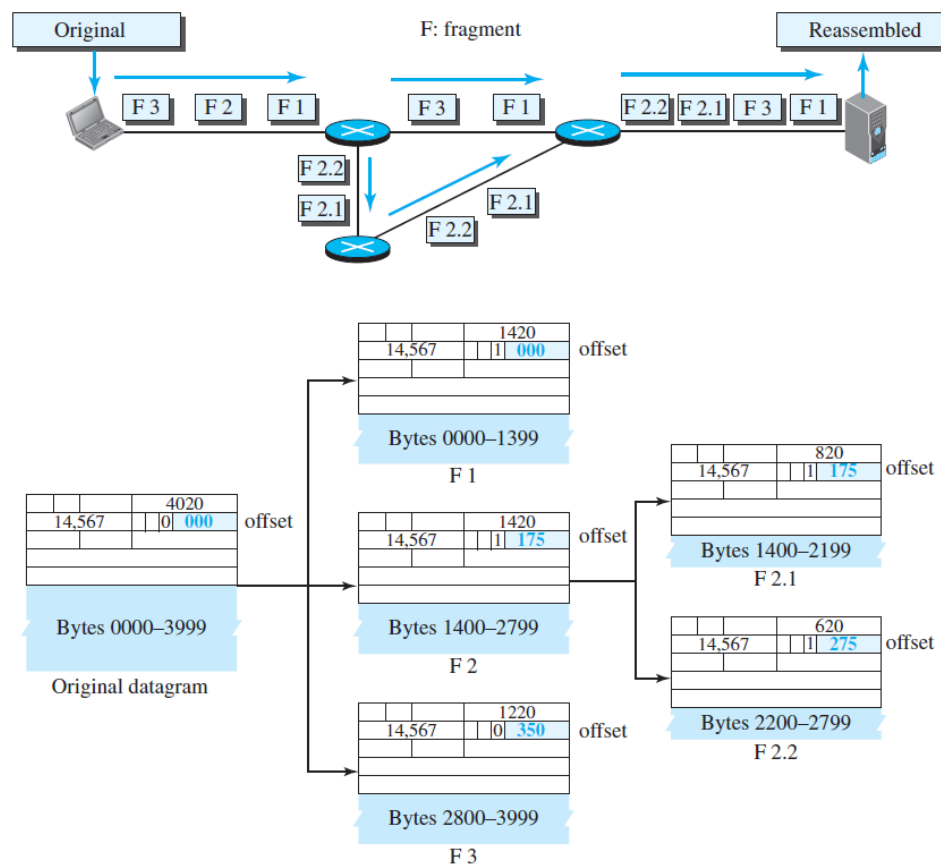


Figure – 22 Detailed fragmentation example

- The figure also shows what happens if a fragment itself is fragmented.
- In this case the value of the offset field is always relative to the original datagram. For example, in the figure, the second fragment is itself fragmented later into two fragments of 800 bytes and 600 bytes, but the offset shows the relative position of the fragments to the original data. It is obvious that even if each fragment follows a different path and arrives out of order, the final destination host can reassemble the original datagram from the fragments received (if none of them is lost) using the following strategy:
 - a. The first fragment has an offset field value of zero.
 - b. Divide the length of the first fragment by 8. The second fragment has an offset value equal to that result.
 - c. Divide the total length of the first and second fragment by 8. The third fragment has an offset value equal to that result.
 - d. Continue the process. The last fragment has its M bit (more bit) set to 0.

Example – 19

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

Solution

- If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A non fragmented packet is considered the last fragment.

Example – 21

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

Solution

- If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

Example – 22

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

Solution

- Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment.

Example – 23

A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?

Solution

- To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length of the data.

Example – 24

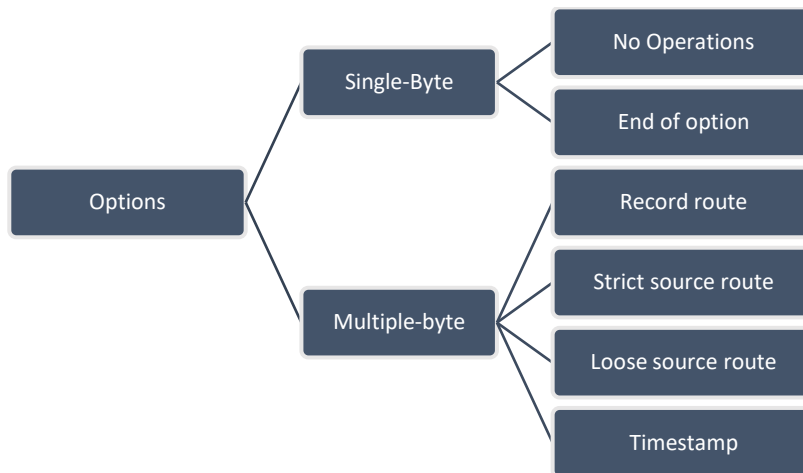
A packet has arrived in which the offset value is 100, the value of HLEN is 5, and the value of the total length field is 100. What are the numbers of the first byte and the last byte?

Solution

- The first byte number is $100 \times 8 = 800$. The total length is 100 bytes, and the header length is 20 bytes (5×4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.

1.4.3 Options

- The header of the IPv4 datagram is made of two parts: a fixed part and a variable part.
- The fixed part is 20 bytes long and was discussed in the previous section.
- The variable part comprises the options that can be a maximum of 40 bytes (in multiples of 4-bytes) to preserve the boundary of the header.
- Options, as the name implies, are not required for a datagram. They can be used for network testing and debugging. Although options are not a required part of the IPv4 header, option processing is required of the IPv4 software. This means that all implementations must be able to handle options if they are present in the header.
- Options are divided into two broad categories: single-byte options and multiple-byte options.
- Taxonomy of options in IPv4 is shown in the figure below.



Single -Byte Options

- There are two single-byte options.
- No Operation
 - A no-operation option is a 1-byte option used as a filler between options.
- End of Option
 - An end-of-option option is a 1-byte option used for padding at the end of the option field. It, however, can only be used as the last option.

Multiple-Byte Options

- There are four multiple-byte options.
- Record Route
 - A record route option is used to record the Internet routers that handle the datagram. It can list up to nine router addresses.
 - It can be used for debugging and management purposes.
- Strict Source Route
 - A strict source route option is used by the source to predetermine a route for the datagram as it travels through the Internet.
 - Dictation of a route by the source can be useful for several purposes.
 - The sender can choose a route with a specific type of service, such as minimum delay or maximum throughput. Alternatively, it may choose a route that is safer or more reliable for the sender's purpose. For example, a sender can choose a route so that its datagram does not travel through a competitor's network.
 - If a datagram specifies a strict source route, all the routers defined in the option must be visited by the datagram. A router must not be visited if its IPv4 address is not listed in the datagram. If the datagram visits a router that is not on the list, the datagram is discarded, and an error message is issued. If the datagram arrives at the destination and some of the entries were not visited, it will also be discarded and an error message issued.

- Loose Source Route
 - A loose source route option is similar to the strict source route, but it is less rigid.
 - Each router in the list must be visited, but the datagram can visit other routers as well.
- Timestamp
 - A timestamp option is used to record the time of datagram processing by a router.
 - The time is expressed in milliseconds from midnight, Universal time or Greenwich mean time.
 - Knowing the time a datagram is processed can help users and managers track the behaviour of the routers in the Internet.
 - We can estimate the time it takes for a datagram to go from one router to another. We say estimate because, although all routers may use Universal time, their local clocks may not be synchronized.

1.4.4 Security of IPv4 Datagrams

- The IPv4 protocol, as well as the whole Internet, was started when the Internet users trusted each other. No security was provided for the IPv4 protocol.
- Today, however, the situation is different; the Internet is not secure anymore.
- There are three security issues that are particularly applicable to the IP protocol:
 1. packet sniffing,
 2. packet modification,
 3. and IP spoofing.
- Packet Sniffing
 - An intruder may intercept an IP packet and make a copy of it.
 - Packet sniffing is a passive attack, in which the attacker does not change the contents of the packet. This type of attack is very difficult to detect because the sender and the receiver may never know that the packet has been copied.
 - Although packet sniffing cannot be stopped, encryption of the packet can make the attacker's effort useless.
 - The attacker may still sniff the packet, but the content is not detectable.
- Packet Modification
 - The second type of attack is to modify the packet. The attacker intercepts the packet, changes its contents, and sends the new packet to the receiver. The receiver believes that the packet is coming from the original sender.
 - This type of attack can be detected using a data integrity mechanism.
 - The receiver, before opening and using the contents of the message, can use this mechanism to make sure that the packet has not been changed during the transmission.

- IP Spoofing
 - An attacker can masquerade as somebody else and create an IP packet that carries the source address of another computer.
 - An attacker can send an IP packet to a bank pretending that it is coming from one of the customers. This type of attack can be prevented using an origin authentication mechanism.
- IPSec
 - The IP packets today can be protected from the previously mentioned attacks using a protocol called IPSec (IP Security).
 - This protocol, which is used in conjunction with the IP protocol, creates a connection-oriented service between two entities in which they can exchange IP packets without worrying about the three attacks discussed above.
 - IPSec provides the following four services:
 - Defining Algorithms and Keys.
 - The two entities that want to create a secure channel between themselves can agree on some available algorithms and keys to be used for security purposes.
 - Packet Encryption.
 - The packets exchanged between two parties can be encrypted for privacy using one of the encryption algorithms and a shared key agreed upon in the first step. This makes the packet sniffing attack useless.
 - Data Integrity.
 - Data integrity guarantees that the packet is not modified during the transmission. If the received packet does not pass the data integrity test, it is discarded. This prevents the second attack, packet modification, described above.
 - Origin Authentication.
 - IPSec can authenticate the origin of the packet to be sure that the packet is not created by an imposter. This can prevent IP spoofing attacks as described above.

1.5 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 payload is an ICMP message.

1.5.1 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 neighbours. 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 – 23 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. The rest of the header is specific for each message type.

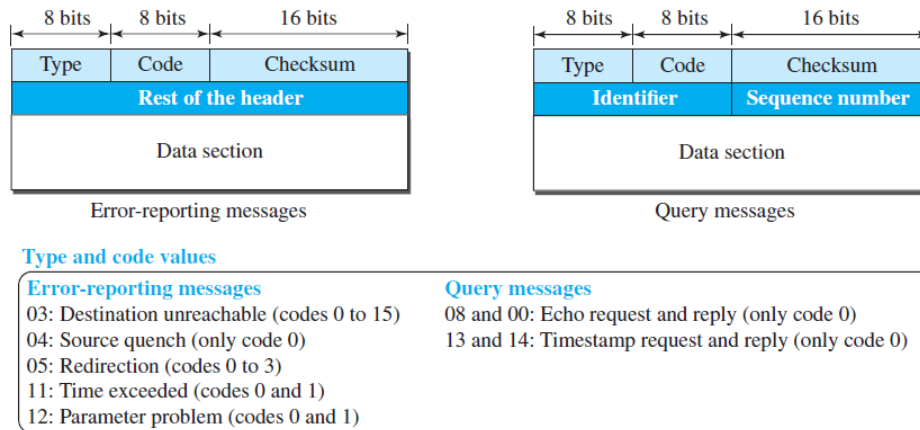


Figure – 23 General format of ICMP messages

- The data section in error messages carries information for finding the original packet that had the error.
- The data section in query messages, 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.

- ICMP forms an error packet, which is then encapsulated in an IP datagram (see Figure – 24).

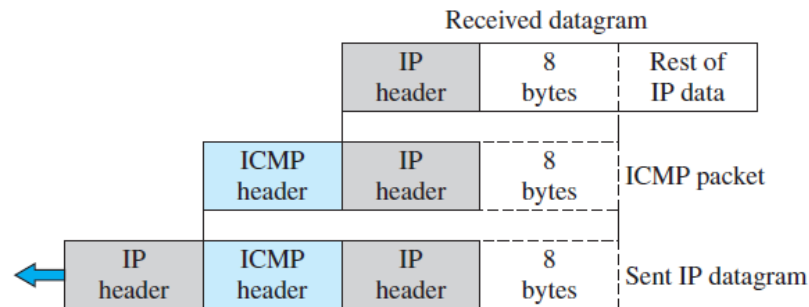


Figure – 24 Contents of data field for the error messages

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.

Time Exceed 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 liveness 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
 - Echo request and echo 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 liveness 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.
 - 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).

1.5.2 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 – 25, 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.

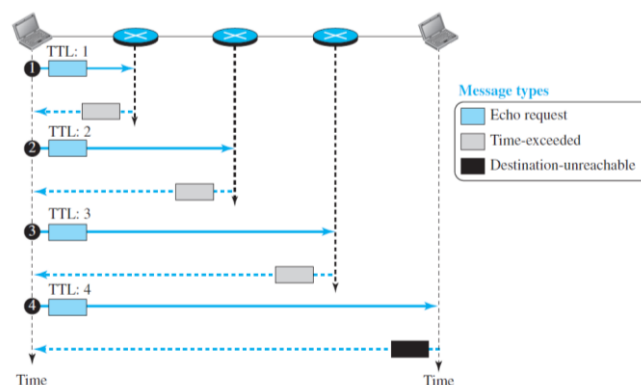


Figure – 25 Use of ICMPv4 in traceroute

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

1.5.3 ICMP Checksum

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

Example – 25

Figure – 26 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.

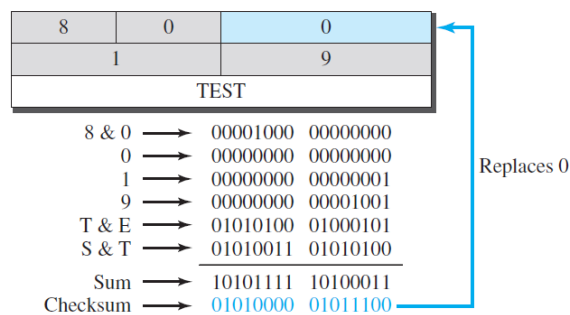


Figure – 26 Example of checksum calculation

References:

1. Data Communication and Networking, Fifth Edition, BEHROUZ A. FOROUZAN
2. Data Communication and Networking, Fourth Edition, BEHROUZAN A. FOROUZA
3. <https://www.geeksforgeeks.org/introduction-of-classful-ip-addressing/>