National Institute of Technology, Silchar

Department of Computer Science and Engineering

CS1508 – Internet Protocols

Report on IPV4 Protocol

By

ADITYA KUMAR MOHANTY (2022108)

MTech - First Year

aditya_pg@cse.nits.ac.in

7008940269

Course Instructor: Dr. Umakanta Majhi

TABLE OF CONTENTS

1.Declaration

I affirm that the report related to IPV4 Protocol being submitted in partial fulfilment for the completion of the course work is the original survey done by me with the help of various online resources. This has not formed the part of any other project report on the basis of which a degree or course completion was conferred on an earlier occasion on this or any other candidate.

I hereby, declare that this report is solely done by me, to the best of my knowledge and belief.

2.Acknowledgment

I take immense pleasure to express my heartfelt thanks to our Course Instructor**, Dr. Umakanta Majhi Sir** for the valuable guidance throughout this course. Sir has provided constant support for all my queries during the course and during the development of the survey report as well. This inspired me to explore more about the topic.

I would also like to acknowledge the contribution of all Research Scholars for their wonderful research papers, online publications and resources, which helped me for the successful completion of this survey report.

Name : ADITYA KUMAR MOHANTY
Scholar No: 2022108
Date: 13-05-2021

IPV4

What is Network?

A Network in the world of computers is said to be a collection of interconnected hosts, via some shared media which can be wired or wireless. A computer network enables its hosts to share and exchange data and information over the media. Network can be a Local Area Network spanned across an office or Metro Area Network spanned across a city or Wide Area Network which can be spanned across cities and provinces.

A computer network can be as simple as two PCs connected together via a single copper cable or it can be grown up to the complexity where every computer in this world is connected to every other, called the Internet. A network then includes more and more components to reach its ultimate goal of data exchange. Below is a brief description of the components involved in computer network:

- **Hosts -** Hosts are said to be situated at ultimate end of the network, i.e. a host is a source of information and another host will be the destination. Information flows end to end between hosts. A host can be a user's PC, an internet Server, a database server etc.
- **Media -** If wired, then it can be copper cable, fiber optic cable, and coaxial cable. If wireless, it can be free-to-air radio frequency or some special wireless band. Wireless frequencies can be used to interconnect remote sites too.
- **Hub -** A hub is a multiport repeater and it is used to connect hosts in a LAN segment. Because of low throughputs hubs are now rarely used. Hub works on Layer-1 (Physical Layer) of OSI Model.
- **Switch -** A Switch is a multiport bridge and is used to connect hosts in a LAN segment. Switches are much faster than Hubs and operate on wire speed. Switch works on Layer-2 (Data Link Layer), but Layer-3 (Network Layer) switches are also available.
- **Router -** A router is Layer-3 (Network Layer) device which makes routing decisions for the data/information sent for some remote destination. Routers make the core of any interconnected network and the Internet.
- **Gateways -** A software or combination of software and hardware put together, works for exchanging data among networks which are using different protocols for sharing data.
- **Firewall -** Software or combination of software and hardware, used to protect users' data from unintended recipients on the network/internet.

All components in a network ultimately serve the hosts.

Host Addressing

Communication between hosts can happen only if they can identify each other on the network. In a single collision domain (where every packet sent on the segment by one host is heard by every other host) hosts can communicate directly via MAC address.

MAC address is a factory coded 48-bits hardware address which can also uniquely identify a host. But if a host wants to communicate with a remote host, i.e. not in the same segment or logically not connected, then some means of addressing is required to identify the remote host uniquely. A logical address is given to all hosts connected to Internet and this logical address is called Internet Protocol Address.

Hardware Addressing

A hardware address is used to uniquely identify a host within a local network. Hardware addressing is a function of the Data-Link layer of the OSI model (Layer-2).

Ethernet utilizes the 48-bit MAC address as its hardware address. The MAC address is often hardcoded on physical network interfaces, though some interfaces support changing the MAC address using special utilities. In virtualization environments, dynamically assigning MAC addresses is very common.

A MAC address is most often represented in hexadecimal, using one of two accepted formats:

00:43:AB:F2:32:13

0043.ABF2.3213

The first six hexadecimal digits of a MAC address identify the manufacturer of the physical network interface. This is referred to as the OUI (Organizational Unique Identifier). The last six digits uniquely identify the host itself, and are referred to as the host ID.

The MAC address has one shortcoming – it contains no hierarchy. MAC addresses provide no mechanism to create boundaries between networks. There is no method to distinguish one network from another.

This lack of hierarchy poses significant difficulties to network scalability. If only Layer-2 hardware addressing existed, all hosts would technically exist on the same network. Internetworks like the Internet could not exist, as it would be impossible to separate my network from your network. Imagine if the entire Internet existed purely as a single Layer-2 switched network. Switches, as a rule, will forward a broadcast out every port. With billions of hosts on the Internet, the resulting broadcast storms would be devastating. The Internet would simply collapse.

The scalability limitations of Layer-2 hardware addresses are mitigated using logical addresses, covered in great detail in this guide.

Logical Addressing

Logical addressing is a function of the Network layer of the OSI Model (Layer-3), and provides a hierarchical structure to separate networks. Logical addresses are never hardcoded on physical network interfaces, and can be dynamically assigned and changed freely.

A logical address contains two components:

- **Network ID –** identifies which network a host belongs to.
- **Host ID –** uniquely identifies the host on that network.

Examples of logical addressing protocols include Internetwork Packet Exchange (IPX) and Internet Protocol (IP). IPX was predominantly used on Novell networks, but is now almost entirely deprecated. IP is the most widely-used logical address, and is the backbone protocol of the Internet.

Internet Protocol (IP)

In the 1970's, the Department of Defence developed the Transmission Control Protocol (TCP), to provide both Network and Transport layer functions. When this proved to be an inflexible solution, those functions were separated - with the Internet Protocol (IP) providing Network layer services, and TCP providing Transport layer services.

Together, TCP and IP provide the core functionality for the TCP/IP or Internet protocol suite.

IP provides two fundamental Network layer services:

- **Logical addressing –** provides a unique address that identifies both the host, and the network that host exists on.
- **Routing –** determines the best path to a particular destination network, and then routes data accordingly.

IP was originally defined in RFC 760, and has been revised several times. IP Version 4 (IPv4) was the first version to experience widespread deployment, and is defined in RFC 791. IPv4 will be the focus of this guide.

IPv4 employs a 32-bit address, which limits the number of possible addresses to 4,294,967,296. IPv4 will eventually be replaced by IP Version 6 (IPv6), due to a shortage of available IPv4 addresses. IPv6 is covered in great detail in another guide.

IPv4 Addressing

A core function of IP is to provide logical addressing for hosts. An IP address provides a hierarchical structure to both uniquely identify a host, and what network that host exists on.

An IP address is most often represented in decimal, in the following format:

158.80.164.3

An IP address is comprised of four octets, separated by periods:

First Octet	Second Octet	Third Octet	Fourth Octet
158	80	164	3

Each octet is an 8-bit number, resulting in a 32-bit IP address. The smallest possible value of an octet is 0, or 00000000 in binary. The largest possible value of an octet is 255, or 11111111 in binary.

The above IP address represented in binary would look as follows:

First Octet	Second Octet	Third Octet	Fourth Octet
10011110	01010000	10100100	00000011

Decimal to Binary Conversion

The simplest method of converting between decimal and binary is to remember the following table:

128	64	32	16	8	4	2	1

To convert a decimal number of 172 to binary, start with the leftmost column. Since 172 is greater than 128, that binary bit will be set to 1. Next, add the value of the next column (128 + 64 = 192). Since 172 is less than 192, that binary bit will be set to 0.

Again, add the value of the next column (128 + 32 = 160). Since 172 is greater than 160, that binary bit will be set to 1. Continue this process until the columns with binary bits set to 1 add up to 172:

Decimal	128	64	32	16	8	4	2	1
Binary	1	0	1	0	1	1	0	0

Binary to Decimal Conversion

Converting from binary back to decimal is even simpler. Apply the binary number to the conversion table, and then add up any columns with binary bits set to 1.

For example, consider the binary number of 11110001:

Decimal	128	64	32	16	8	4	2	1
Binary	1	1	1	1	0	0	0	1

By adding 128 + 64 + 32 + 16+ 1, it can be determined that 11110001 equals 241.

The Subnet Mask

Part of an IP address identifies the network. The other part of the address identifies the host. A subnet mask is required to provide this distinction:

158.80.164.3

255.255.0.0

The above IP address has a subnet mask of 255.255.0.0. The subnet mask follows two rules:

- If a binary bit is set to a 1 (or on) in a subnet mask, the corresponding bit in the address identifies the network.
- If a binary bit is set to a 0 (or off) in a subnet mask, the corresponding bit in the address identifies the host.

Looking at the above address and subnet mask in binary:

IP Address: 10011110.01010000.10100100.00000011

Subnet Mask: 11111111.11111111.00000000.00000000

The first 16 bits of the subnet mask are set to 1. Thus, the first 16 bits of the address (158.80) identify the network. The last 16 bits of the subnet mask are set to 0. Thus, the last 16 bits of the address (164.3) identify the unique host on that network.

The network portion of the subnet mask must be contiguous. For example, a subnet mask of 255.0.0.255 is not valid.

Hosts on the same logical network will have identical network addresses, and can communicate freely. For example, the following two hosts are on the same network:

Host A: 158.80.164.100

Subnet mask:255.255.0.0

Host B: 158.80.164.101

255.255.0.0

Both share the same network address (158.80), which is determined by the 255.255.0.0 subnet mask. Hosts that are on different networks cannot communicate without an intermediating device. For example:

Host A: 158.80.164.100

255.255.0.0

Host B: 158.85.164.101

255.255.0.0

The subnet mask has remained the same, but the network addresses are now different (158.80 and 158.85 respectively). Thus, the two hosts are not on the same network, and cannot communicate without a router between them. Routing is the process of forwarding packets from one network to another.

Consider the following, trickier example:

Host A: 158.80.1.1 255.248.0.0

Host B: 158.79.1.1 255.248.0.0

The specified subnet mask is now 255.248.0.0, which doesn't fall cleanly on an octet boundary. To determine if these hosts are on separate networks, first convert everything to binary:

Host A Address: 10011110.01010000.00000001.00000001

Host B Address: 10011110.01001111.00000001.00000001

Subnet Mask: 11111111.11111000.00000000.00000000

Remember, the 1 (or on) bits in the subnet mask identify the network portion of the address. In this example, the first 13 bits (the 8 bits of the first octet, and the first 5 bits of the second octet) identify the network. Looking at only the first 13 bits of each address:

Host A Address: 10011110.01010

Host B Address: 10011110.01001

Clearly, the network addresses are not identical. Thus, these two hosts are on separate networks, and require a router to communicate.

IP Address Classes

The IPv4 address space has been structured into several classes. The value of the first octet of an address determines the class of the network:

Class	*First Octet Range*	*Default Subnet Mask*
Class A	1 - 127	255.0.0.0
Class B	128 - 191	255.255.0.0
Class C	192 - 223	255.255.255.0
Class D	224 - 239	-

Class A networks range from 1 to 127.The default subnet mask is 255.0.0.0. Thus, by default, the first octet defines the network, and the last three octets define the host. This results in a maximum of 127 Class A networks, with 16,777,214 hosts per network!

Example of a Class A address:

Address: 64.32.254.100

Subnet Mask: 255.0.0.0

Class B networks range from 128 to 191. The default subnet mask is 255.255.0.0. Thus, by default, the first two octets define the network, and the last two octets define the host. This results in a maximum of 16,384 Class B networks, with 65,534 hosts per network.

Example of a Class B address:

Address: 152.41.12.195

Subnet Mask: 255.255.0.0

Class C networks range from 192 to 223. The default subnet mask is 255.255.255.0. Thus, by default, the first three octets define the network, and the last octet defines the host. This results in a maximum of 2,097,152 Class C networks, with 254 hosts per network.

Example of a Class C address:

Address: 207.79.233.6

Subnet Mask: 255.255.255.0

Class D networks are reserved for multicast traffic. Class D addresses do not use a subnet mask.

Subnetting

Subnetting is the process of creating new networks (or subnets) by stealing bits from the host portion of a subnet mask. There is one caveat: stealing bits from hosts creates more networks but fewer hosts per network.

Consider the following Class C network:

192.168.254.0

The default subnet mask for this network is 255.255.255.0. This single network can be segmented, or subnetted, into multiple networks. For example, assume a minimum of 10 new networks are required. Resolving this is possible using the following magical formula:

2^n

The exponent ‘n’ identifies the number of bits to steal from the host portion of the subnet mask. The default Class C mask (255.255.255.0) looks as follows in binary:

11111111.1111111.1111111.00000000

There are a total of 24 bits set to 1, which are used to identify the network. There are a total of 8 bits set to 0, which are used to identify the host, and these host bits can be stolen.

Stealing bits essentially involves changing host bits (set to 0 or off) in the subnet mask to network bits (set to 1 or on). Remember, network bits in a subnet mask must always be contiguous - skipping bits is not allowed.

Consider the result if three bits are stolen. Using the above formula:

$2^n = 2\ 3 = 8 = 8$ new networks created

However, a total of 8 new networks does not meet the original requirement of at least 10 networks. Consider the result if four bits are stolen:

$2^n = 2\ 4 = 16 = 16$ new networks created

A total of 16 new networks does meet the original requirement. Stealing four host bits results in the following new subnet mask:

11111111.11111111.11111111.11110000 = 255.255.255.240

In the previous example, a Class C network was subnetted to create 16 new networks, using a subnet mask of 255.255.255.240 (or /28 in CIDR). Four bits were stolen in the subnet mask, leaving only four bits for hosts.

To determine the number of hosts this results in, for each of the new 16 networks, a slightly modified formula is required:

$2^n – 2$

Consider the result if four bits are available for hosts:

$2^n – 2 = 2\ 4 – 2 = 16 – 2 = 14$ usable hosts per network

Thus, subnetting a Class C network with a /28 mask creates 16 new networks, with 14 usable hosts per network.

Why is the formula for calculating usable hosts $2^n – 2$? Because it is never possible to assign a host an address with all 0 or all 1 bits in the host portion of the address. These are reserved for the subnet and broadcast addresses, respectively. Thus, every time a network is subnetted, useable host addresses are lost.

The 2^n -2 Rule and Subnetted Networks

To avoid confusion, it was historically unacceptable to use the first and last new networks created when subnetting, as it is possible for a classful network to have the same subnet and broadcast address as its subnetted networks. This required the $2^n – 2$ formula to also be used when calculating the number of new networks created while subnetting.

However, this is no longer a restriction for modern equipment and routing protocols. Specifically, on Cisco IOS devices, the following command is now enabled by default:

Router(config)# ip subnet-zero

The ip subnet-zero commands allows for the use of networks with all 0 or all 1 bits in the stolen network portion of the address. Thus, the formula for calculating the number of new networks created is simply 2^n .

Remember though, the formula for calculating usable hosts is always $2^n - 2$.

Supernetting

Supernetting is the opposite of Subnetting**.** In subnetting, a single big network is divided into multiple smaller subnetworks. In Supernetting, multiple networks are combined into a bigger network termed as a Supernetwork or Supernet.
Supernetting is mainly used in Route Summarization, where routes to multiple networks with similar network prefixes are combined into a single routing entry, with the routing entry pointing to a Super network, encompassing all the networks. This in turn significantly reduces the size of routing tables and also the size of routing updates exchanged by routing protocols.

More specifically,

- When multiple networks are combined to form a bigger network, it is termed as super-netting
- Super netting is used in route aggregation to reduce the size of routing tables and routing table updates

There are some points which should be kept in mind while supernetting:

1. All the Networks should be contiguous.
2. The block size of every networks should be equal and must be in form of 2^n.
3. First Network id should be exactly divisible by whole size of supernet.

Advantages of Supernetting –
1. Control and reduce network traffic
2. Helpful to solve the problem of lacking IP addresses
3. Minimizes the routing table

Disadvantages of Supernetting –
- It cannot cover different area of network when combined
- All the networks should be in same class and all IP should be contiguous

Codes-

i)Program to validate an IP address and find its class?

```
#include <stdio.h>
#include <string.h>

void extractIpAddress(unsigned char *sourceString,short *ipAddress)
{
unsigned short len=0;
unsigned char  oct[4]={0},cnt=0,cnt1=0,i,buf[5];

len=strlen(sourceString);
for(i=0;i<len;i++)
{
if(sourceString[i]!='.'){
buf[cnt++] =sourceString[i];
}
if(sourceString[i]=='.' || i==len-1){
buf[cnt]='\0';
cnt=0;
oct[cnt1++]=atoi(buf);
}
}
ipAddress[0]=oct[0];
ipAddress[1]=oct[1];
ipAddress[2]=oct[2];
ipAddress[3]=oct[3];
}
int main()
{
```

```
unsigned char ip[20]={0};
short ipAddress[4];
printf("Enter IP Address (xxx.xxx.xxx.xxx format): ");
scanf("%s",ip);
extractIpAddress(ip,&ipAddress[0]);
printf("\nIp Address: %03d. %03d. %03d. %03d\n",ipAddress[0],ipAddress[1],ipAddress[2],ipAddress[3]);
if(ipAddress[0]>=0 && ipAddress[0]<=127)
printf("Class A Ip Address.\n");
if(ipAddress[0]>127 && ipAddress[0]<191)
printf("Class B Ip Address.\n");
if(ipAddress[0]>191 && ipAddress[0]<224)
printf("Class C Ip Address.\n");
if(ipAddress[0]>224 && ipAddress[0]<=239)
printf("Class D Ip Address.\n");
if(ipAddress[0]>239)
printf("Class E Ip Address.\n");
return 0;
}
```

ii)Program to determine class, Network and Host ID of an IPv4 address

```
#include<stdio.h>
#include<string.h>

// Function to find out the Class
char findClass(char str[])
{
```

```
// storing first octet in arr[] variable
char arr[4];
int i = 0;
while (str[i] != '.')
{
        arr[i] = str[i];
        i++;
}
i--;


int ip = 0, j = 1;
while (i >= 0)
{
        ip = ip + (str[i] - '0') * j;
        j = j * 10;
        i--;
}

// Class A
if (ip >=1 && ip <= 126)
        return 'A';

// Class B
else if (ip >= 128 && ip <= 191)
        return 'B';

// Class C
else if (ip >= 192 && ip <= 223)
        return 'C';
```

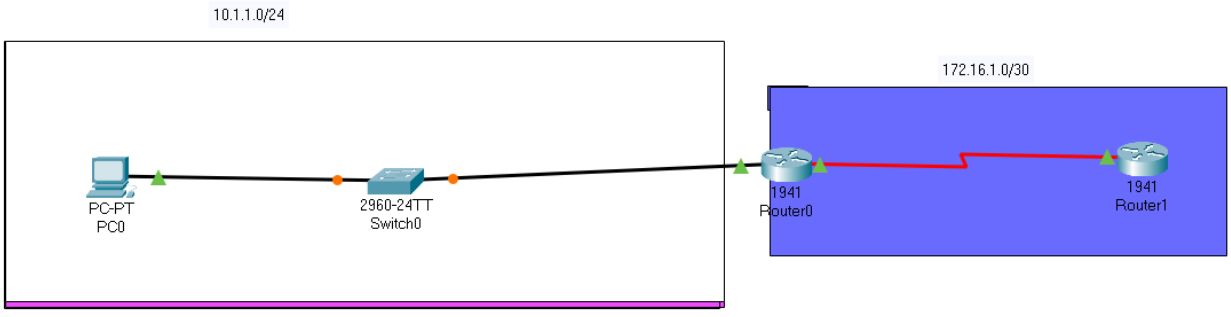
```
	// Class D
	else if (ip >= 224 && ip <= 239)
		return 'D';

	// Class E
	else
		return 'E';
}

// Function to separate Network ID as well as
// Host ID and print them
void separate(char str[], char ipClass)
{
	// Initializing network and host array to NULL
	char network[12], host[12];
	for (int k = 0; k < 12; k++)
		network[k] = host[k] = '\0';

	// for class A, only first octet is Network ID
	// and rest are Host ID
	if (ipClass == 'A')
	{
		int i = 0, j = 0;
		while (str[j] != '.')
			network[i++] = str[j++];
		i = 0;
		j++;
		while (str[j] != '\0')
			host[i++] = str[j++];
```

```
        printf("Network ID is %s\n", network);
        printf("Host ID is %s\n", host);
}

// for class B, first two octet are Network ID
// and rest are Host ID
else if (ipClass == 'B')
{
        int i = 0, j = 0, dotCount = 0;

        // storing in network[] up to 2nd dot
        // dotCount keeps track of number of
        // dots or octets passed
        while (dotCount < 2)
        {
                network[i++] = str[j++];
                if (str[j] == '.')
                        dotCount++;
        }
        i = 0;
        j++;

        while (str[j] != '\0')
                host[i++] = str[j++];

        printf("Network ID is %s\n", network);
        printf("Host ID is %s\n", host);
}
```

```
// for class C, first three octet are Network ID
    // and rest are Host ID
    else if (ipClass == 'C')
    {
            int i = 0, j = 0, dotCount = 0;

            // storing in network[] up to 3rd dot
            // dotCount keeps track of number of
            // dots or octets passed
            while (dotCount < 3)
            {
                    network[i++] = str[j++];
                    if (str[j] == '.')
                            dotCount++;
            }

            i = 0;
            j++;

            while (str[j] != '\0')
                    host[i++] = str[j++];

            printf("Network ID is %s\n", network);
            printf("Host ID is %s\n", host);
    }

    // Class D and E are not divided in Network
    // and Host ID
    else
            printf("In this Class, IP address is not"
```

```
                " divided into Network and Host ID\n");
}

// Driver function is to test above function
int main()
{
        char str[] = "192.226.12.11";
        char ipClass = findClass(str);
        printf("Given IP address belongs to Class %c\n",

        separate(str, ipClass);
        return 0;
}
```

Simulation of ipv4



CONFIGURATION STEPS

1. enable
2. configure terminal
3. interface type number
4. no shutdown
5. ip address ip-address mask
6. end

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface and enters interface configuration mode.
Step 4	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Enables the interface.

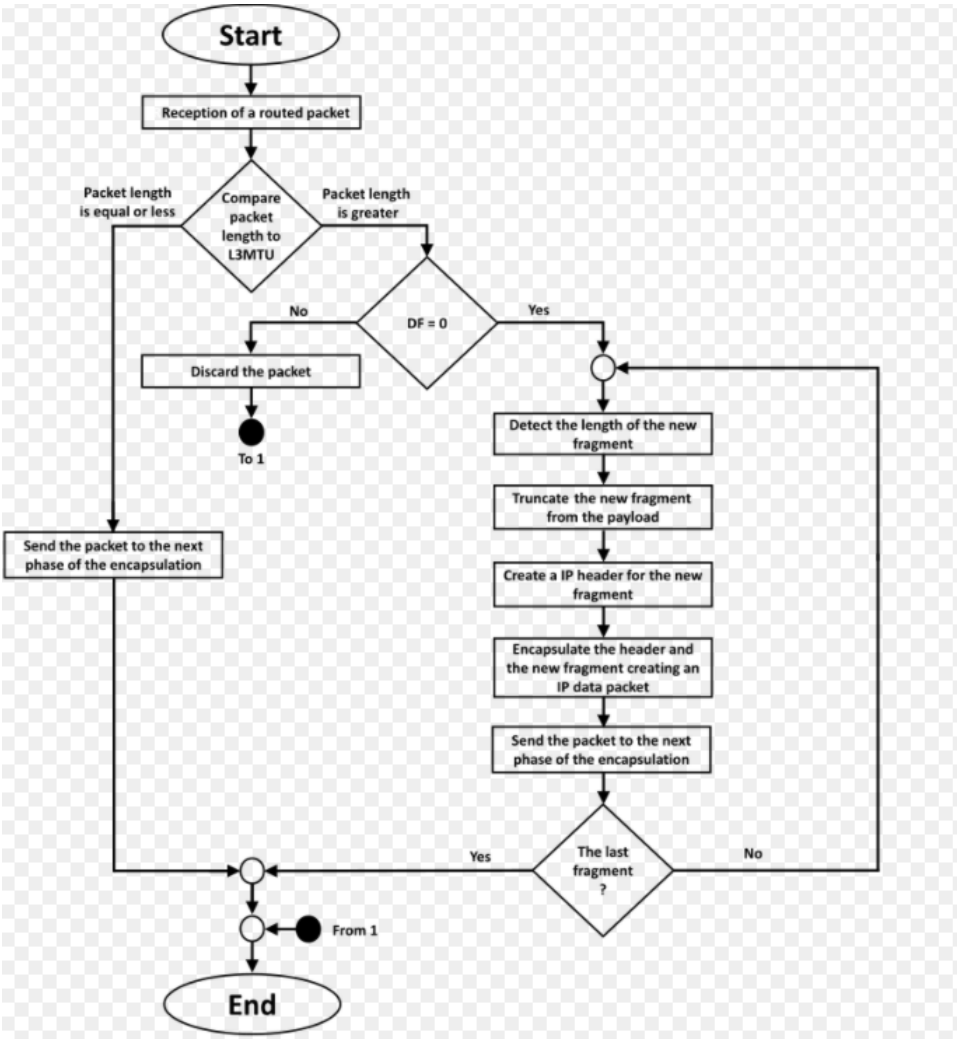
	Command or Action	Purpose
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.16.16.1 255.255.240.0`	Configures the IP address on the interface.
Step 6	**end** **Example:** `Router(config-if)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

FRAGMENTATION

When a host sends an IP packet onto the network it cannot be larger than the maximum size supported by that local network. This size is determined by the network's data link and IP Maximum Transmission Units (MTUs) which are usually the same. A typical contemporary office, campus or data centre network provided over Ethernet will have 1500 byte MTUs.However, packets that are initially transmitted over a network supporting one MTU may need be routed across networks (such as a WAN or VPN tunnel) with a smaller MTU. In these cases, if the packet size exceeds the lower MTU the data in the packet must be fragmented (if possible). This means it is broken into pieces carried within new packets (fragments) that are equal to or smaller than the lower MTU. This is called Fragmentation and the data in these fragments is then typically reassembled when they reach their destination.

Preventing Fragmentation

A node can prevent packets being fragmented by setting the Don't Fragment (DF) flag in those packets to a value of 1. Packets that must be fragmented but have the DF bit set are discarded.



CONCLUSION

Ipv4 is early stage internet protocol which is used in the core networking communication device. In this project we discussed about various terms and class of ipv4 like class A,B,C,D,E and we know about various cocept like subnetting and supernetting.and we perfrom simulation of ipv4 using cisco packet tracer,some codes to determine net id and host as well as classes of ip address.Finally we saw the problem statement og ipv4 like fragmantaion and implemented algorithm to resolve this problem and draw the flow chart of it.

References

1. https://commons.wikimedia.org/wiki/File:IPv4_Fragmentation_Algorithm-en.svg
2. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_ipv4/configuration/xe-3s/ipv4-xe-3s-book.pdf
3. https://en.wikipedia.org/wiki/IPv4
4. https://www.riverpublishers.com/journal/journal_articles/RP_Journal_2245-800X_714.pdf
5. https://www.geeksforgeeks.org/what-is-ipv4/
6. https://www.slideshare.net/sonaltelang/ipv4-ppt

Also refered youtube video for simulation and configuration.