Introduction to Computer Networks
Homework 4: Solution

Part I.

4.6

Consider a datagram network using 8-bit host addresses. Suppose a router uses longest prefix matching and has the following forwarding table:

Prefix Match	Interface
00	0
01	1
100	2
otherwise	3

For each of the four interfaces, give the associated range of destination host addresses and the number of addresses in the range.

Interface 0	
From	00000000
To	00111111
Count	$2^6 = 64$

Interface 1	
From	01000000
To	01111111
Count	$2^6 = 64$

Interface 2	
From	10000000
To	10011111
Count	$2^5 = 32$

Interface 3	
From	10100000
To	11111111
Count	$2^7 - 2^5 = 96$

4.14

Consider sending a 1,600-byte datagram into a link that has an MTU of 500 bytes. Suppose the original datagram is stamped with the identification number 291. How many fragments are generated? What are the values in the various fields in the IP datagram(s) generated related to fragmentation?

Since the IP header is 20 bytes, the data is $1600 - 20 = 1580$ bytes, and the maximum size of data field in each fragment is $500 - 20 = 480$ bytes.

Thus, the offset is $\frac{480}{8} = 60$, and the number of required fragments is $\left\lceil \frac{1580}{480} \right\rceil = 4$.

No.	Length	ID	Flag	Offset
1	500	291	1	0
2	500	291	1	60
3	500	291	1	120
4	160	291	0	180

4.15

Suppose datagrams are limited to 1,500 bytes (including header) between source Host A and destination Host B. Assuming a 20-byte IP header, how many datagrams would be required to send an MP3 consisting of 5 million bytes? Explain how you computed your answer.

The size of the MP3 file is 5 million bytes. Assuming the data is transmitted using TCP segments, where each TCP segment has a 20-byte header and the IP header is 20 bytes, each datagram can carry $1500 - 20 - 20 = 1460$ bytes of the MP3 file.

Therefore, the number of datagrams required is

$$\left\lceil \frac{5 \times 10^6}{1500 - 20 - 20} \right\rceil = 3425$$

All datagrams, except the last one, will be 1500 bytes. The final datagram will be $960 + 40 = 1000$ bytes. Note that there is no fragmentation in this case; the source host ensures that datagrams are not larger than 1500 bytes, and these datagrams are smaller than the MTU of the links.

Part II. Additional problems.

II.1

What is the problem of NAT when either having a server behind NAT (in the client-server model) or having a peer behind NAT (in the peer-to-peer model)? How can it be avoided/solved? Is there a special name for this solution?

In the client-server model, when a server is behind NAT, it may be inaccessible to clients outside the local network due to private IP addresses. In the peer-to-peer model, peers behind NAT may encounter difficulties establishing direct connections with each other.

To address this issue, ***port forwarding*** can be employed. Port forwarding is a networking technique that allows incoming network requests on a specific port to be directed to a specific computer or service on a private network. This is typically used to allow external users to access a service running on a computer behind a NAT router.

Other methods such as ***hole punching***, ***connection reversal***, etc., are also acceptable.

II.2

Suppose an ISP owns the block of addresses of the form 192.168.56.128/26. Suppose it wants to create four subnets from this block, with each block having the same number of IP addresses. What are the prefixes (of form a.b.c.d/x) for the four subnets?

192.168.56.128/26				
From	11000000	10101000	00111000	10000000 (128)
To	11000000	10101000	00111000	10111111 (191)

To create 4 subnets, we need 2 bits: 00, 01, 10, 11.

Subnet 1				
From	11000000	10101000	00111000	10000000 (128)
To	11000000	10101000	00111000	10001111 (143)
192.168.56.128/28				

Subnet 2				
From	11000000	10101000	00111000	10010000 (144)
To	11000000	10101000	00111000	10011111 (159)
192.168.56.144/28				

Subnet 3				
From	11000000	10101000	00111000	10100000 (160)
To	11000000	10101000	00111000	10101111 (175)
192.168.56.160/28				

Subnet 4				
From	11000000	10101000	00111000	10110000 (176)
To	11000000	10101000	00111000	10111111 (191)
192.168.56.176/28				

Thus, each block has $2^4 = 16$ IP addresses.