Ayush Agarwal
[17114017]

Assignment 3

22th August, 2019

1. Write a socket program in C to determine class, Network and Host ID of an IPv4 address.

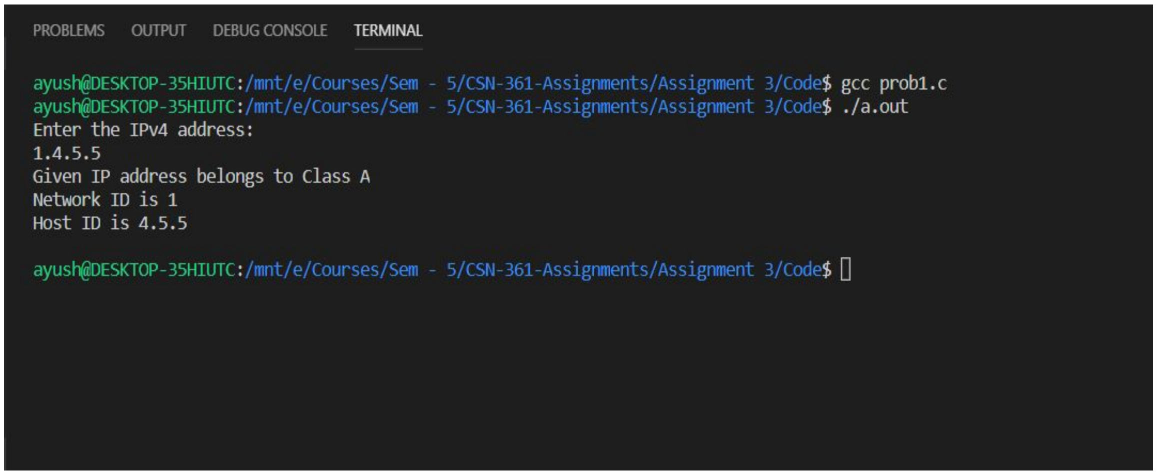
   The program first reads the first octet of the IP address. This is used to indicate the class of the address as follows:

1 - 126	:	A
128 - 191	:	B
192 - 223	:	C
224 - 239	:	D
240 - 255	:	E

   The Network ID is indicated in terms of octets as follows:

Class A	:	first 1 octet
Class B	:	first 2 octets
Class C	:	first 3 octets

   The remaining part gives the Host ID.
   Classes D and E are not divided into Network and Host IDs

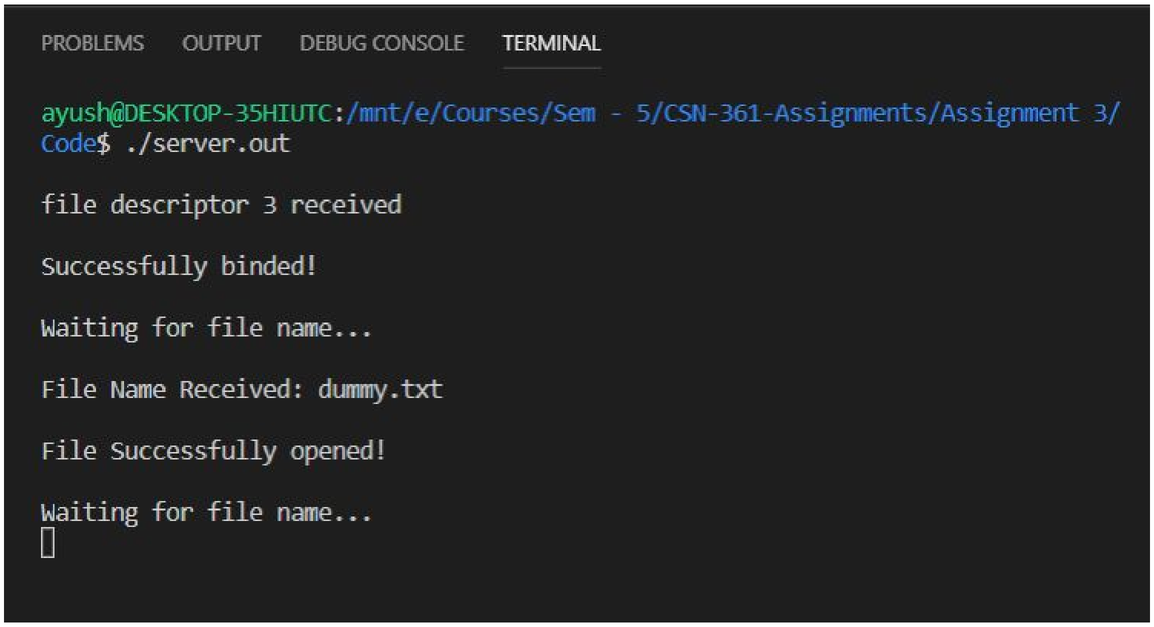


```
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ gcc prob1.c
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ ./a.out
Enter the IPv4 address:
1.4.5.5
Given IP address belongs to Class A
Network ID is 1
Host ID is 4.5.5

ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ 
```

2. Write a C program to demonstrate File Transfer using UDP.

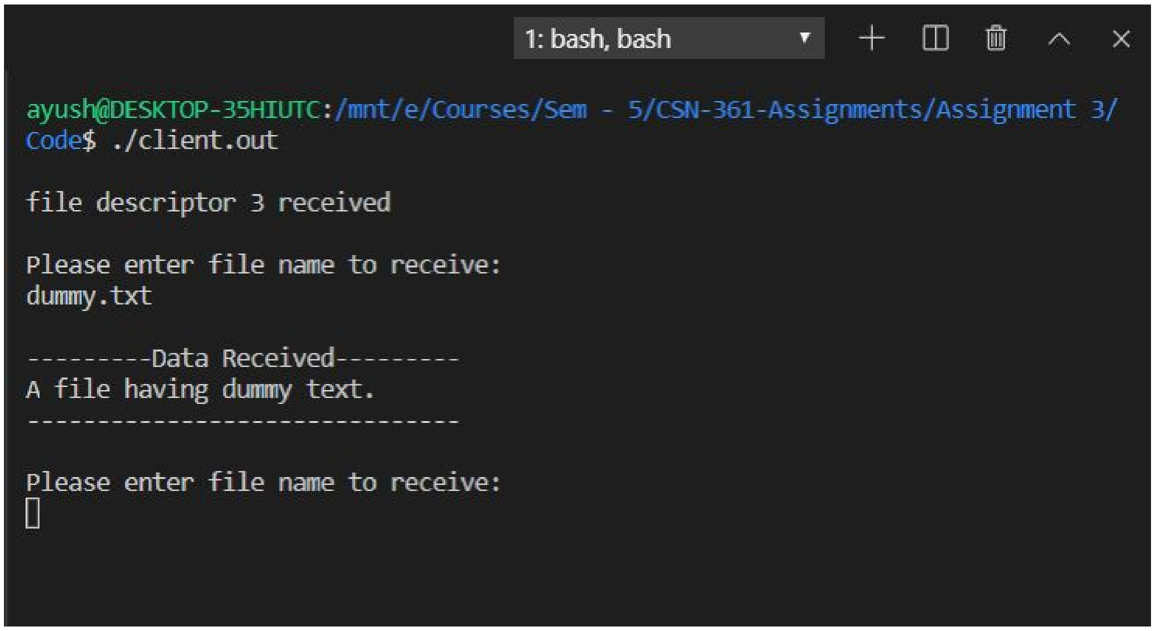
   The server first attaches to a socket and starts listening for the client. The client checks the socket's availability and sends the name of the text file to the server. The server on receiving this, reads the text file and sends the data to the client. The data is ciphered using 2 the XOR operator and using 'S' as the ciphering key -> can be seen in the Cipher() function. As used in previous assignments, struct sockaddr_in is used to use and perform operations on a socket.

   Server:



```
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/
Code$ ./server.out

file descriptor 3 received

Successfully binded!

Waiting for file name...

File Name Received: dummy.txt

File Successfully opened!

Waiting for file name...
```

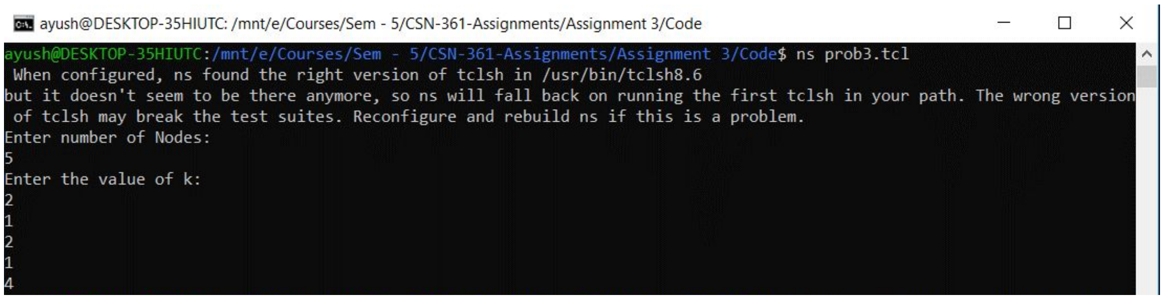
   Client:



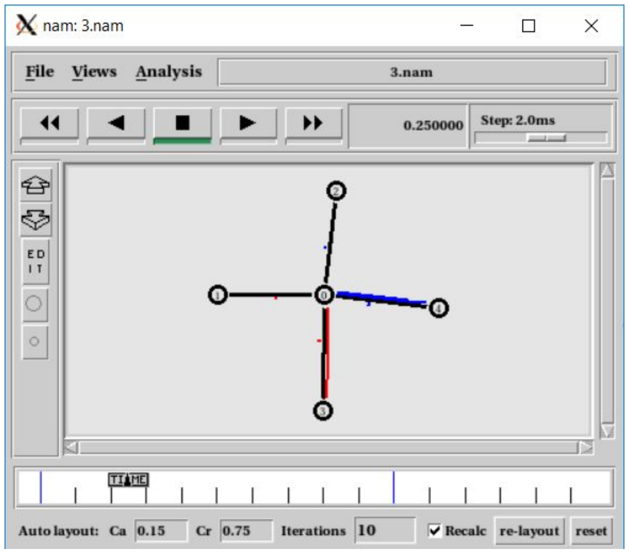
```
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/
Code$ ./client.out

file descriptor 3 received

Please enter file name to receive:
dummy.txt

---------Data Received---------
A file having dummy text.
-------------------------------

Please enter file name to receive:
```

3. Write a TCL code for network simulator NS2 to demonstrate the **star** topology among a set of computer nodes. Given N nodes, one node will be assigned as the central node and the other nodes will be connected to it to form the star. You have to set up a TCP connection between k pairs of nodes and demonstrate the packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.

   We first create a separate nam file in our TCLcode to get the visual interface. We then create a node array and fill it with nodes (no. of nodes is taken as input from the user using the ‘gets’ keyword in TCL) . For each connection, we take the two nodes on the two ends from the user and attach a new tcp connection on the sender node and a sink on the receiver node. Each data transfer is started with a gap of 0.1 seconds from the previous one. Since this is a **Star topology**, we take the 0th node as the center (router) and form a duplex-link from this node to every other node. Therefore, every data transfer has to pass through the 0th node.

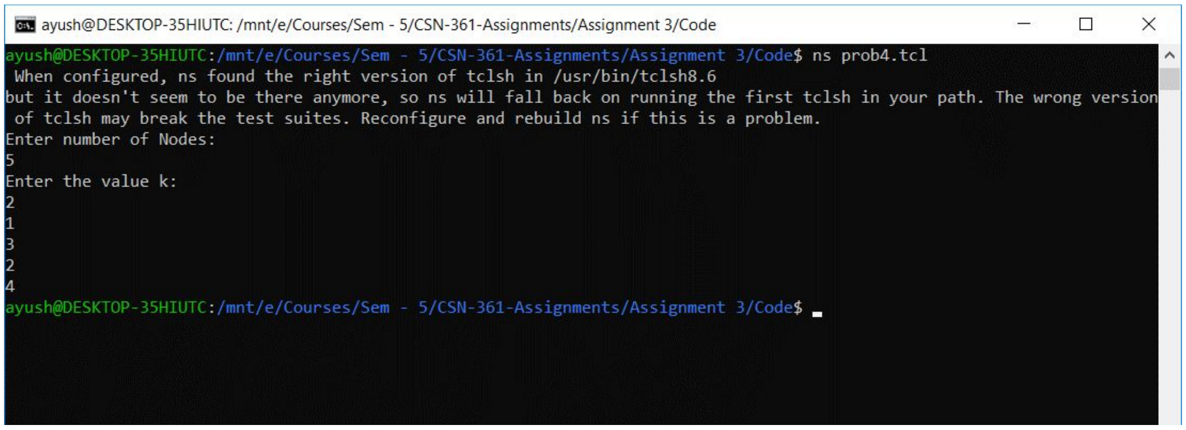


```
ayush@DESKTOP-35HIUTC: /mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ ns prob3.tcl
 When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version
 of tclsh may break the test suites. Reconfigure and rebuild ns if this is a problem.
Enter number of Nodes:
5
Enter the value of k:
2
1
2
1
4
```

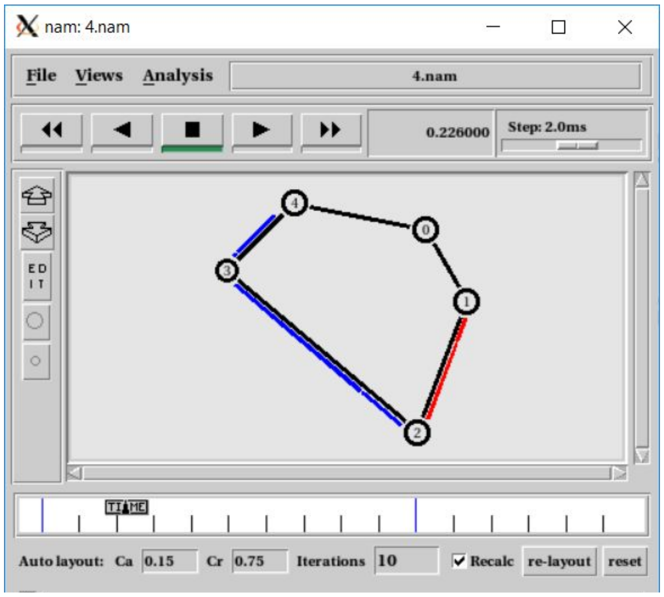

nam: 3.nam
File
Views
Analysis
3.nam
0.250000
Step: 2.0ms
EDIT
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc re-layout reset

4. Write a TCL code for network simulator NS2 to demonstrate the **ring** topology among a set of computer nodes. Given N nodes, each node will be connected to two other nodes in the form of a ring. You have to set up a TCP connection between k pairs of nodes and demonstrate packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.

The input and variable definitions are the same as above. Since this is a **Ring topology**, For every x th node, we form a duplex-link with the (x-1) th node and in the end, join the nth and the 0th node to finish the ring. TCL automatically finds the nearest path from one node to the other when required to transfer data. This can be verified in the NAM simulation.

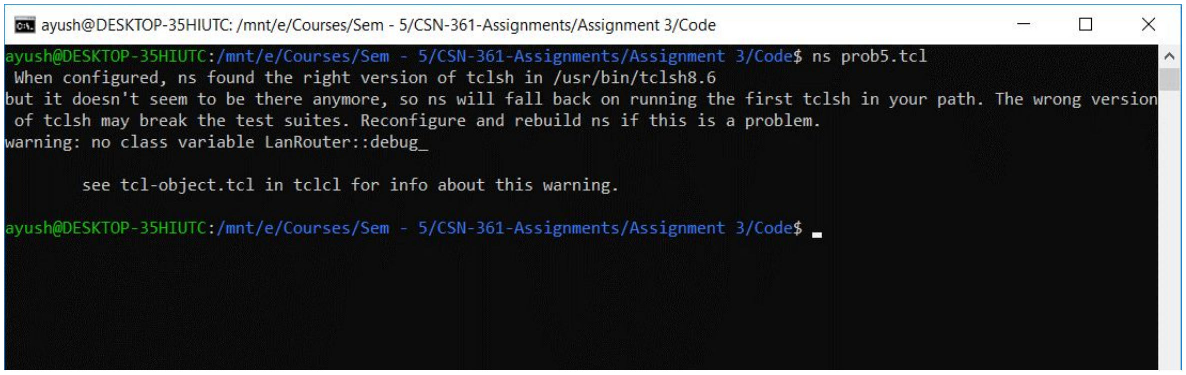


```
ayush@DESKTOP-35HIUTC: /mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ ns prob4.tcl
 When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version
 of tclsh may break the test suites. Reconfigure and rebuild ns if this is a problem.
Enter number of Nodes:
5
Enter the value k:
2
1
3
2
4
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$
```

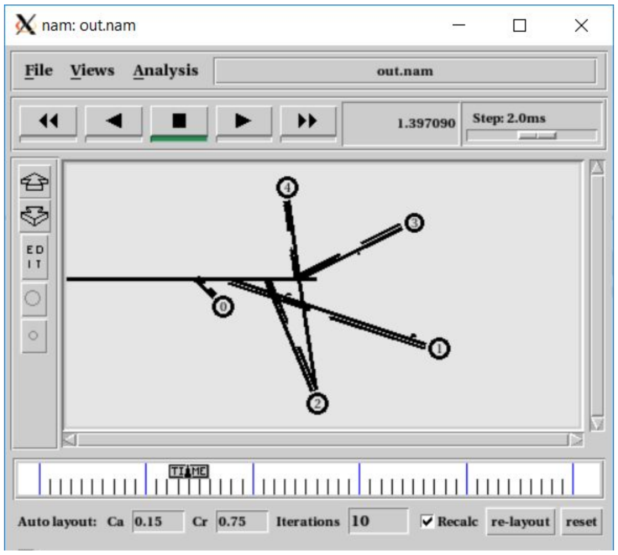


5. Write a TCL code for network simulator NS2 to demonstrate the **bus** topology among a set of computer nodes. Given N nodes, each node will be connected to a common link. You have to set up a TCP connection between k pairs of nodes and demonstrate packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.

The input and variable definitions are the same as above. Since this is a **Bus topology**, we first create a lan/bus using the “make-lan” command and connect all nodes to this lan. We then create the tcp-sink pairs as seen in previous programs and start the NAM simulator.



```
ayush@DESKTOP-35HIUTC: /mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code
ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ ns prob5.tcl
 When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version
 of tclsh may break the test suites. Reconfigure and rebuild ns if this is a problem.
warning: no class variable LanRouter::debug_

        see tcl-object.tcl in tclcl for info about this warning.

ayush@DESKTOP-35HIUTC:/mnt/e/Courses/Sem - 5/CSN-361-Assignments/Assignment 3/Code$ _
```


nam: out.nam
File
Views
Analysis
out.nam
1.397090
Step: 2.0ms
EDIT
4
3
0
1
2
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc re-layout reset