



CSN-361 Lecture-3

21.08.2019

Goddu Vishal

17114034

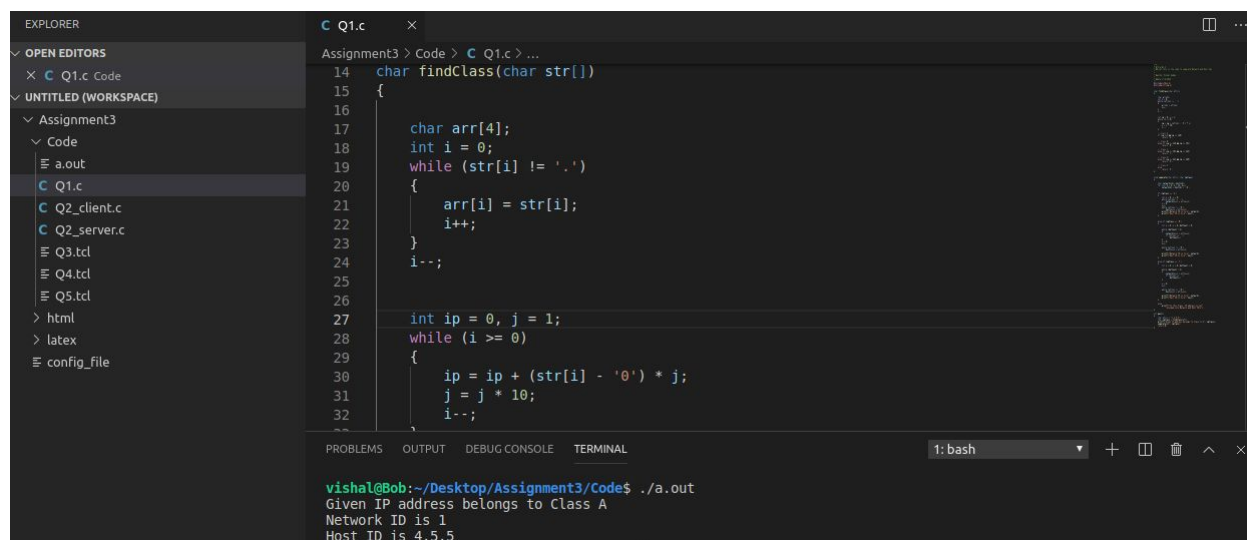
CS-1

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

This is done by basic string parsing and value checking.

We look at the first octet (the integer before the first '.' in the IP address) and depending on the range that it lies in, we find out its class (A to E). Then depending on this class, we split the IP address into network and host addresses.

For example: class A -> first octet is network address and the remaining is host address.



```
14 char findClass(char str[])
15 {
16
17     char arr[4];
18     int i = 0;
19     while (str[i] != '.')
20     {
21         arr[i] = str[i];
22         i++;
23     }
24     i--;
25
26
27     int ip = 0, j = 1;
28     while (i >= 0)
29     {
30         ip = ip + (str[i] - '0') * j;
31         j = j * 10;
32         i--;
33     }
```

```
vishal@Bob:~/Desktop/Assignment3/Code$ ./a.out
Given IP address belongs to Class A
Network ID is 4
Host ID is 5
```

Write a C program to demonstrate File Transfer using UDP.

Similar to previous assignments, struct sockaddr_in is used to use and perform operations on a socket.

The data is ciphered using the XOR operator and using 'S' as the ciphering key -> can be seen in the Cipher() function.

The server first binds to a socket and starts listening for the client. The client checks whether the socket is available, and if available, it sends the name of the file to be read through the buffer. The server receives the file name checks whether file exists or not. If

the file exists it reads the file and sends the data to the client.

```

File Edit Selection View Go Debug Terminal Help
EXPLORER
  OPEN EDITORS
    Q1.c Code
    Q2_client.c Code
    Q2_server.c Code
  UNTITLED (WORKSPACE)
    Assignment3
      Code
        Q1.c
        Q2_client.c
        Q2_server.c
        q2c
        q2s
        Q3.tcl
        Q4.tcl
        Q5.tcl
        test.txt
        html
        latex
        config_file

Q1.c
Q2_client.c
Q2_server.c
q2c
q2s
Q3.tcl
Q4.tcl
Q5.tcl
test.txt
html
latex
config_file

Q2_server.c
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108

else
printf("\nFile has been opened\n");

while (1) {
    #define NET_BUF_SIZE 32
    Expands to:
    if (sendFile(fp, net_buf, NE 32
        sendto(sockfd, net_buf, NET_BUF_SIZE, sendrecvflag, (struct sockaddr*)&addr_c
        break;
    }

    sendto(sockfd, net_buf, NET_BUF_SIZE, sendrecvflag, (struct sockaddr*)&addr_con,
    clearBuf(net_buf);
}

if (fp != NULL)

```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
vishal@Bob:~/Desktop/Assignment3/Code$ ./q2s
file descriptor 3 received
Waiting for file name...
File Name is test.txt
File has been opened

vishal@Bob:~/Desktop/Assignment3/Code$ ./q2c
file descriptor 3 received
Enter the file name to receive:
test.txt
Data Received
Test file

```

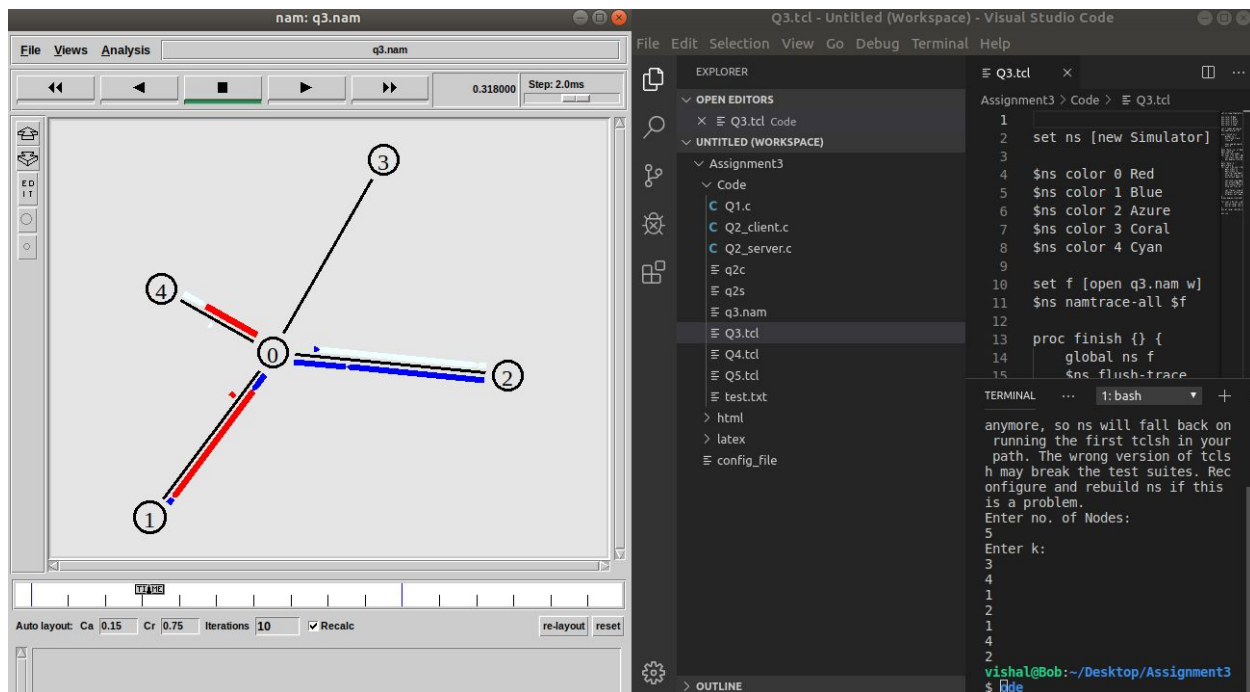
Write a TCL code for network simulator NS2 to demonstrate the STAR TOPOLOGY among a set of computer nodes. Given N nodes, one node will be assigned as the central node and the other nodes will be connected to it to form the star. You have to set up a TCP connection between k pairs of nodes and demonstrate 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.

Since we have to make a star topology, we place the 0th node as the center node and form a duplex-link to every other node. This ensures that all the data passes through node 0.

We create a nam file in our TCL code to get the simulation.

We take the number of nodes as input and fill the node array.

We take two nodes at the ends from the user and we attach a tcp connection on the sender node and a sink on the receiver node, identically for every connection.

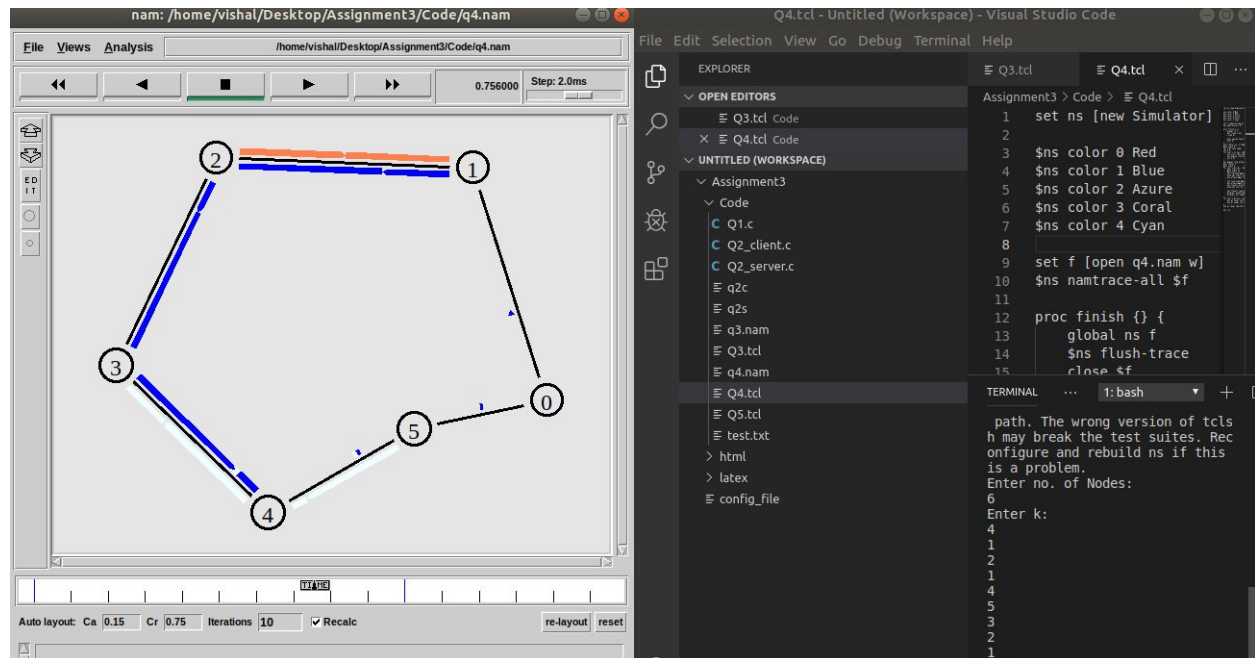


Write a TCL code for network simulator NS2 to demonstrate the RING TOPOLOGY among a set of computer nodes. Given N nodes, each node will be connected to two other nodes in the form of a ring. You have to set up a TCP connection between k pairs of nodes and demonstrate packet transfer between them using 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.

Since we need to make a ring topology. Every node must be connected with its next as well as previous node with a duplex-link, and the nth node must be joined with the 0th.

TCL ensures the nearest path of transfer from one node to another as seen in the simulation

The input and variable definitions are the same as the previous question.



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 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.

As we need to build a bus topology, we create a lan using the "make-lan" command. All the subsequent nodes are then connected to this lan. The tcp pairing follows the same process as the previous two questions. The input and variable definitions are the same as the previous question.

The image displays a NetSim environment on the left and a Visual Studio Code editor on the right.

NetSim Environment:

- File: `/home/vishal/Desktop/Assignment3/Code/q5.nam`
- Analysis: `/home/vishal/Desktop/Assignment3/Code/q5.nam`
- Simulation progress: 0.882000, Step: 2.0ms
- Network topology: A diagram showing five nodes (0, 1, 2, 3, 4) connected by links. Node 0 is at the bottom left, node 1 is at the bottom right, node 2 is in the center, node 3 is at the top right, and node 4 is at the top left. Links connect 0-2, 1-2, 2-3, 2-4, and 3-4.
- Bottom status bar: Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc re-layout reset

Visual Studio Code Editor:

- File: `Q5.tcl - Untitled (Workspace) - Visual Studio Code`
- Explorer: Shows the project structure with files like `Q1.c`, `Q2_client.c`, `Q2_server.c`, `Q3.nam`, `Q3.tcl`, `Q4.nam`, `Q4.tcl`, `Q5.nam`, `Q5.tcl`, `test.txt`, `html`, `latex`, and `config_file`.
- Code Editor: Shows the content of `Q5.tcl`:

```
Assignment3 > Code > Q5.tcl
32 $cbr0 set packetSize 5000
33 $cbr0 set interval_ 0.0
34 $cbr0 attach-agent $tcp
35
36 $ns at 0.5 "$cbr0 start"
37 $ns at 4.5 "$cbr0 stop"
38
39 $ns at 5.0 "finish"
40
41
42 $ns run
```
- Terminal: Shows the output of the simulation:

```
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. Rec onfigure and rebuild ns if this is a problem.
warning: no class variable LanRouter::debug

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