**DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT**



**DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS**

Academic year 2024-2025 (odd semester)

Semester –III

Course code **23MCA33**

**Computer Networks Lab**

Prepared by

Ms. Shama Dessai

HOD-MCA Principal

Dr. Manjula Sanjay Dr. Ravishankar M

**Vision of the Institute**

To be a centre of excellence in education, research & training and to produce citizens with exceptional leadership qualities to serve national and global needs

**Mission of the Institute**

To achieve our objectives in an environment that enhances creativity, innovation and scholarly pursuits while adhering to our vision.

**Computer Lab Rules and Regulations**

**DO’s**

* Come prepared to the Lab.
* Submit your Records to the faculty and sign in the Log Book on entering the Lab
* Observation books have to be brought for all the labs.
* Backlog exercises to be executed after completing regular exercises.
* Regularly attend all the labs
* Put the chairs back to its position before you leave.
* Treat all the devices with care and consideration.
* Behave in a responsible manner at all the times and maintain silence.
* Before leaving the lab shut down the system and rearrange the chairs
* Keep your premises clean

**DON’T**

* Use Mobile phones and pen drives
* Move around in the lab during the lab session.
* Tamper System Files or Try to access the Server.
* Write Records in the Lab
* Change the system assigned to you without the notice of the Lab Staff.
* Write on the table or mouse pads.
* Do not install or download any software or modify or delete any system files on any lab computers

**MCA Department**

## **Vision**

Nurture Continuous Learning through research and innovations in the field of Computer Science, Technology and Applications, to build competent professionals.

## **Mission**

* Create a learning environment to motivate students to build strong technology skills.
* Promote value based ethical practices in all facets of learning
* Instill Entrepreneurial collaborative thinking through structured interventions and industry participation.

## **Program Education Outcome (PEO’s):**

* **PEO1:** Analyse real life problems, design computing systems appropriate to its solutions that are technically sound, economically feasible and socially acceptable.
* **PEO2:** Exhibit professionalism, ethical attitude, communication skills, team work in their profession and adapt to current trends by engaging in lifelong learning.
* **PEO3:** Demonstrate Leadership and Entrepreneurship Skills by incorporating organizational goals.

## **Program Outcome (PO’s):**

* **PO1. Computational Knowledge:** Understand and apply mathematical foundation, computing and domain knowledge for the conceptualization of computing models from defined problems.
* **PO2. Problem Analysis:** Ability to identify, critically analyze and formulate complex computing problems using fundamentals of computer science and application domains.
* **PO3. Design / Development of Solutions:** Ability to transform complex business scenarios and contemporary issues into problems, investigate, understand and propose integrated solutions using emerging technologies
* **PO4. Conduct Investigations of Complex Computing Problems:** Ability to devise and conduct experiments, interpret data and provide well informed conclusions.
* **PO5. Modern Tool Usage:** Ability to select modern computing tools, skills and techniques necessary for innovative software solutions
* **PO6. Professional Ethics:** Ability to apply and commit professional ethics and cyber regulations in a global economic environment.
* **PO7. Life-long Learning:** Recognize the need for and develop the ability to engage in continuous learning as a Computing professional.
* **PO8. Project Management and Finance:** Ability to understand, management and computing principles with computing knowledge to manage projects in multidisciplinary environments.
* **PO9. Communication Efficacy:** Communicate effectively with the computing community as well as society by being able to comprehend effective documentations and presentations.
* **PO10. Societal & Environmental Concern:** Ability to recognize economical, environmental, social, health, legal, ethical issues involved in the use of computer technology and other consequential responsibilities relevant to professional practice.
* **PO11. Individual & Team Work:** Ability to work as a member or leader in diverse teams in multidisciplinary environment.
* **PO12. Innovation and Entrepreneurship:** Identify opportunities, entrepreneurship vision and use of innovative ideas to create value and wealth for the betterment of the individual and society.

## Program Specific Outcomes (PSO’s):

* **PSO1:** The graduates of the Program will have skills to develop, deploy and maintain applications for desktop, web, mobile, cloud and cross platforms using modern tools and technologies.
* **PSO2:** The graduates of the program analyze the societal needs to provide novel solutions through technological based research.

**Course Outcomes**

|  |  |  |
| --- | --- | --- |
| **CO’s** | **Description** | **Revised Blooms Level** |
| 1 | Apply error detection and error correction codes | L3 |
| 2 | Analyze and congestion issues in network design. | L4 |
| 3 | Implement Encoding and Decoding techniques used in presentation layer. | L5 |
| 4 | Implementing Network topologies | L6 |

**Mapping of Course Outcomes with Program Outcomes and PSO**

**Correlation levels: 1-Slight (Low) 2-Moderate (Medium) 3-Substantial (High)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO’s** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** |
| 1 | 3 | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ |
| 2 | \_ | 3 | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ |
| 3 | \_ | \_ | 3 | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ | \_ |
| 4 | \_ | \_ | \_ | \_ | 3 | \_ | \_ | \_ | \_ | \_ | 3 | \_ |
| 5 | \_ | \_ | \_ | 3 | - | \_ | \_ | \_ | \_ | \_ | \_ | \_ |

**Mapping of CO v/s PSO**:

|  |  |  |
| --- | --- | --- |
| **CO’s** | **PSO1** | **PSO2** |
| 1 | 3 | 1 |
| 2 | 3 | 2 |
| 3 | 3 | 2 |
| 4 | 2 | 3 |
| 5 | 3 | 2 |

**Justifications for CO-PO/PSO Mapping**

|  |  |  |
| --- | --- | --- |
| **Mapping** | **Low/Medium/ High** | **Justification** |
| CO1-PO1 | 3 | Strongly mapped students will be able to apply queries are create databases. |
| CO1-PSO1 | 3 | Strongly mapped students will be able to apply skills to develop, deploy and maintain applications for desktop, web, mobile, cloud and cross platforms using modern tools and technologies. |
| CO1-PSO2 | 1 | Slightly mapped students will be able to analyze the societal needs to provide novel solutions through technological based research. |
| CO2-PO2 | 3 | Strongly mapped students will be able complex computing problems. |
| CO2-PSO1 | 3 | Strongly mapped students will be able to apply skills to develop, deploy and maintain applications for desktop, web. |
| CO2-PSO2 | 2 | Strongly mapped students will be able to develop the data flow diagrams-based Diagrams and databases. |
| CO3-PO3 | 3 | Strongly mapped students will be able flow diagrams-based Diagrams and databases. |
| CO3-PO4 | 3 | Strongly mapped students will be able to work in the groups and individual in the query solving. |
| CO3-PSO1 | 3 | Strongly mapped students will be able to develop the skills in innovations. |
| CO3-PSO2 | 2 | Moderately mapped students will be able to analyze the technologies. |
| CO4-PO5 | 2 | Moderately mapped students will be able to identify the query in to relation. |
| CO4-PO11 | 3 | Strongly mapped students will be able to identify the queries of sql and transform in to sql execute queries. |
| CO4-PSO1 | 2 | Moderately mapped students will be able to identify the query in to relation. |
| CO4-PSO2 | 3 | Strongly mapped students will be able to identify the queries and implement the complex to simple. |
| CO5-PO1 | 3 | Strongly mapped students will be able will be able complex computing problems. |
| CO5-PO2 | 3 | Strongly mapped students will be able flow diagrams-based Diagrams and databases. |
| CO5-PO3 | 3 | Strongly mapped students will be able identify the query in to relation. |
| CO5-PO4 | 3 | Strongly mapped students will be able to identify the query in to relation. |
| CO5-PSO1 | 3 | Strongly mapped students will be able identify the query in to relation. |
| CO5-PSO2 | 2 | Moderately mapped students will be able identify the queries and implement the complex to simple. |

**Gaps within the Syllabus: There is no gap identified as per the curriculum**

**Total number of subjects in second semester=9**

**A=16.4**

**T=9\*3=27/2=13.5**

**A >=T[A=16.4,T=13.5]**

**It implies “Strongly mapped Pos/PSOs”**

**Internal evaluation procedure**

Marking Scheme for Internal Marks:

Total Marks: 50

Division of 50 marks is as follows:

CIE : 20

Continuous Evaluation:30

**Continuous evaluation**: Observation:10, Record:10, execution:20, viva:10 [50] Marks

Scale down to 30 Marks

**CIE-1**: **100** Marks

**CIE-2:100** Marks

Write up:20, execution: 60, Viva:20 [100] Marks

Total marks scale down to 20 Marks

**External evaluation procedure**

Marking Scheme for External Marks:

Total Marks: 100

Division of 100 marks is as follows:

Procedure : 15

Write up & Execution: 70

Viva : 15

**INDEX**

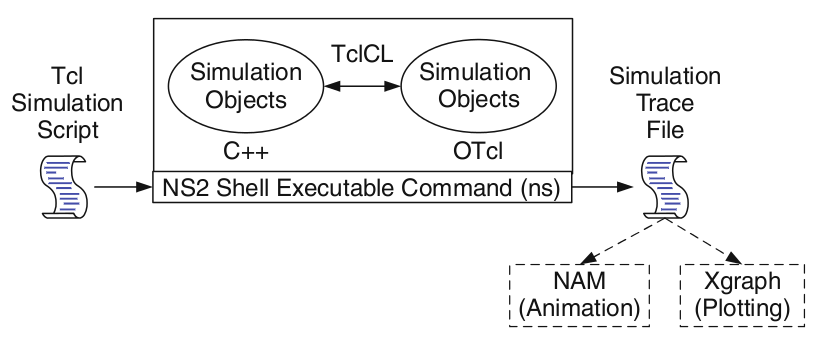
|  |  |
| --- | --- |
| **Sl. No.** | **Program** |
| 1. | **Simulate the network with five nodes n0, n1, n2, n3, n4, forming a star topology. The node n4 is at the center. Node n0 is a TCP source, which transmits packets to node n3 (a TCP sink) through the node n4. Node n1 is another traffic source, and sends UDP packets to node n2 through n4. The duration of the simulation time is 10 seconds.** |
| 2. | **Simulate a variety of IP based Networks using NS2** |
| 3. | **Develop a program to compute CRC code for the polynomials CRC-12, CRC-16 and CRC CCIP** |
| 4. | **Develop a simple data link layer that performs the flow control using the sliding window protocol, and loss recovery using the Go-Back-N mechanism** |
| 5. | **Evaluates the existing network protocol performance using NS2.** |
| 6. | **Simulate the network with five nodes n0, n1, n2, n3, n4, forming a star topology. The node n4 is at the center. Node n0 is a TCP source, which transmits packets to node n3 (a TCP sink) through the node n4. Node n1 is another traffic source and sends UDP packets to node n2 through n4. The duration of the simulation time is 10 seconds.** |
| 7. | **Simulate to study the transmission of packets over Ethernet LAN and determine the number of packets drop destination.** |
| 8. | **Simulate the different types of internet traffic such as FTP and TELNET over a wired network and analyze the packet drop and packet delivery ratio in the network.** |
| 9. | **Implement the connectivity of Network cables and Practically implement the cross-wired cable and straight-through cable using a clamping tool.** |
| 10. | **Configuring a Cisco Router as a DHCP Server.** |
| 11. | **Applications using TCP sockets like:**  **Echo client and echo server**  **Chat**  **File Transfer.** |
| 12. | **Simulation of Distance Vector/ Link State Routing algorithm** |
| **OPEN ENDED PROGRAM** | |
| 1 | **Implement the data link layer framing methods such as character-stuffing and bit stuffing.** |
| 2 | **Implement data encryption and data decryption** |
| 3 | **Implement the code to simulate ARP/RARP protocol** |

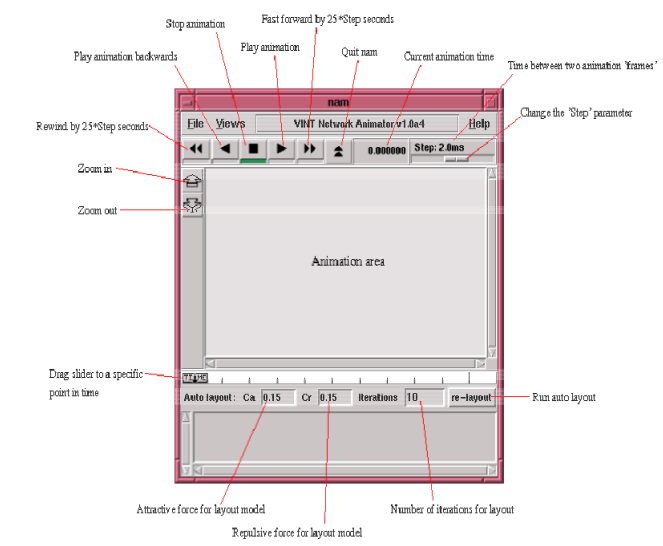
**PART-A**

**Introduction to NS-2:**

* Widely known as NS2, is simply an event driven simulation tool.
* Useful in studying the dynamic nature of communication networks.
* Simulation of wired as well as wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be done using NS2.
* In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviors.

**Basic Architecture of NS2**





**Wired TCL Script Components**

Create the event scheduler

Open new files & turn on the tracing

Create the nodes

Setup the links

Configure the traffic type (e.g., TCP, UDP, etc)

Set the time of traffic generation (e.g., CBR, FTP)

Terminate the simulation

**Initialization and Termination of TCL Script in NS-2**

An ns simulation starts with the command

**set ns [new Simulator]**

Which is thus the first line in the tcl script? This line declares a new variable as using the set command, you can call this variable as you wish, In general people declares it as ns because

it is an instance of the Simulator class, so an object the code[new Simulator] is indeed the installation of the class Simulator using the reserved word new.

In order to have output files with data on the simulation (trace files) or files used for visualization (nam files), we need to create the files using “open” command:

**#Open the Trace file**

**set tracefile1 [open out.tr w]**

**$ns trace-all $tracefile1**

**#Open the NAM trace file**

**set namfile [open out.nam w]**

**$ns namtrace-all $namfile**

The above creates a dta trace file called “out.tr” and a nam visualization trace file called “out.nam”.Within the tcl script,these files are not called explicitly by their names,but instead by pointers that are declared above and called “tracefile1” and “namfile” respectively.Remark that they begins with a # symbol.The second line open the file “out.tr” to be used for writing,declared with the letter “w”.The third line uses a simulator method called trace-all that have as parameter the name of the file where the traces will go.

The last line tells the simulator to record all simulation traces in NAM input format.It also gives the file name that the trace will be written to later by the command $ns flush-trace.In our case,this will be the file pointed at by the pointer “$namfile”,i.e the file “out.tr”.

The termination of the program is done using a “finish” procedure.

**#Define a ‘finish’ procedure**

**Proc finish { } {**

**global ns tracefile1 namfile**

**$ns flush-trace**

**Close $tracefile1**

**Close $namfile**

**Exec nam out.nam &**

**Exit 0}**

The word proc declares a procedure in this case called **finish** and without arguments. The word **global** is used to tell that we are using variables declared outside the procedure. The simulator method “**flush-trace”** will dump the traces on the respective files. The tcl command “**close”** closes the trace files defined before and **exec** executes the nam program for visualization. The command **exit** will ends the application and return the number 0 as status to the system. Zero is the default for a clean exit. Other values.

can be used to say that is a exit because something fails.

At the end of ns program we should call the procedure “finish” and specify at what time the termination should occur. For example,

**$ns at 125.0 “finish”**

will be used to call “**finish**” at time 125sec.Indeed,the **at** method of the simulator allows us to schedule events explicitly.

The simulation can then begin using the command

**$ns run**

**Structure of Trace Files**

When tracing into an output ASCII file, the trace is organized in 12 fields as follows in fig shown below, The meaning of the fields are:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Event | Time | From  Node | To  Node | PKT  Type | PKT  Size | Flags | Fid | Src  Addr | Dest  Addr | Seq  Num | Pkt  id |

**XGRAPH**

The xgraph program draws a graph on an x-display given data read from either data file or from standard input if no files are specified. It can display upto 64 independent data sets using different colors and line styles for each set. It annotates the graph with a title, axis labels, grid lines or tick marks, grid labels and a legend.

**Syntax:**

**Xgraph [options] file-name**

Options are listed here

**/-bd <color> (Border)**

This specifies the border color of the xgraph window.

**/-bg <color> (Background)**

This specifies the background color of the xgraph window.

**/-fg<color> (Foreground)**

This specifies the foreground color of the xgraph window.

**/-lf <fontname> (LabelFont)**

All axis labels and grid labels are drawn using this font.

**/-t<string> (Title Text)**

This string is centered at the top of the graph.

**/-x <unit name> (XunitText)**

This is the unit name for the x-axis. Its default is “X”.

**/-y <unit name> (YunitText)**

This is the unit name for the y-axis. Its default is “Y”.

**Awk- An Advanced**

awk is a programmable, pattern-matching, and processing tool available in UNIX. It works equally well with text and numbers.

awk is not just a command, but a programming language too. In other words, awk utility is a pattern scanning and processing language. It searches one or more files to see if they contain lines that match specified patterns and then perform associated actions, such as writing the line to the standard output or incrementing a counter each time it finds a match.

Syntax:

**awk option ‘selection\_criteria {action}’ file(s)**

Here, selection\_criteria filters input and select lines for the action component to act upon. The selection\_criteria is enclosed within single quotes and the action within the curly braces. Both the selection\_criteria and action forms an awk program.

**Example: $ awk ‘/manager/ {print}’ emp.lst**

**Variables**

Awk allows the user to use variables of there choice. You can now print a serial number, using the variable kount, and apply it those directors drawing a salary exceeding 6700:

**$ awk –F”|” ‘$3 == “director” && $6 > 6700 {**

**kount =kount+1**

**printf “ %3f %20s %-12s %d\n”, kount,$2,$3,$6 }’ empn.lst**

**THE –f OPTION: STORING awk PROGRAMS IN A FILE**

You should holds large awk programs in separate file and provide them with the awk extension for easier identification. Let’s first store the previous program in the file empawk.awk:

$ cat empawk.awk

Observe that this time we haven’t used quotes to enclose the awk program. You can now use awk with the –f *filename* option to obtain the same output:

**Awk –F”|” –f empawk.awk empn.lst**

**The Begin And End Sections**

Awk statements are usually applied to all lines selected by the address, and if there are no addresses, then they are applied to every line of input. But, if you have to print something before processing the first line, for example, a heading, then the BEGIN section can be used gainfully. Similarly, the end section useful in printing some totals after processing is over.

The BEGIN and END sections are optional and take the form

**BEGIN {action}**

**END {action}**

These two sections, when present, are delimited by the body of the awk program. You can use them to print a suitable heading at the beginning and the average salary at the end.

**BUILT-IN VARIABLES**

Awk has several built-in variables. They are all assigned automatically, though it is also possible for a user to reassign some of them. You have already used NR, which signifies the record number of the current line. We’ll now have a brief look at some of the other variable.

***The FS Variable****:* as stated elsewhere, awk uses a contiguous string of spaces as the default field delimiter. FS redefines this field separator, which in the sample database happens to be the |. When used at all, it must occur in the BEGIN section so that the body of the program knows its value before it starts processing:

**BEGIN {FS=”|”}**

This is an alternative to the –F option which does the same thing.

***The OFS Variable****:* when you used the print statement with comma-separated arguments, each argument was separated from the other by a space. This is awk’s default output field separator, and can reassigned using the variable OFS in the BEGIN section:

**BEGIN { OFS=”~” }**

When you reassign this variable with a ~ (tilde), awk will use this character for delimiting the print arguments. This is a useful variable for creating lines with delimited fields.

***The NF variable****:* NF comes in quite handy for cleaning up a database of lines that don’t contain the right number of fields. By using it on a file, say emp.lst, you can locate those lines not having 6 fields, and which have crept in due to faulty data entry:

**$awk ‘BEGIN {FS = “|”}**

**NF! =6 {Print “Record No “, NR, “has”, “fields”}’ empx.lst**

**LAB CYCLE**

**1)Simulate the network with five nodes n0, n1, n2, n3, n4, forming a star topology. The node n4 is at the center. Node n0 is a TCP source, which transmits packets to node n3 (a TCP sink) through the node n4. Node n1 is another traffic source, and sends UDP packets to node n2 through n4. The duration of the simulation time is 10 seconds.**

**#TCL Script:**

**set ns [new Simulator]**

**set namfile [open ex\_01.nam w]**

**$ns namtrace-all $namfile**

**set tracefile [open ex\_01.tr w]**

**$ns trace-all $tracefile**

**set n0 [$ns node]**

**set n1 [$ns node]**

**set n2 [$ns node]**

**set n3 [$ns node]**

**set n4 [$ns node]**

**$ns duplex-link $n0 $n4 1Mb 10ms DropTail**

**$ns duplex-link $n1 $n4 1Mb 10ms DropTail**

**$ns duplex-link $n4 $n3 1Mb 10ms DropTail**

**$ns duplex-link $n4 $n2 1Mb 10ms DropTail**

**set tcp [new Agent/TCP]**

**$ns attach-agent $n0 $tcp**

**set sink [new Agent/TCPSink]**

**$ns attach-agent $n3 $sink**

**$ns connect $tcp $sink**

**set ftp [new Application/FTP]**

**$ftp attach-agent $tcp**

**set udp [new Agent/UDP]**

**$ns attach-agent $n1 $udp**

**set null [new Agent/Null]**

**$ns attach-agent $n2 $null**

**$ns connect $udp $null**

**$udp set class\_ 1**

**$ns color 1 Blue**

**$tcp set class\_ 2**

**$ns color 2 Red**

**set cbr [new Application/Traffic/CBR]**

**$cbr set packetsize\_ 500**

**$cbr set interval\_ 0.005**

**$cbr attach-agent $udp**

**$ns at 0.0 "$cbr start"**

**$ns at 0.0 "$ftp start"**

**$ns at 9.0 "$cbr stop"**

**$ns at 9.0 "$ftp stop"**

**proc finish {} {**

**global ns namfile tracefile**

**$ns flush-trace**

**close $namfile**

**close $tracefile**

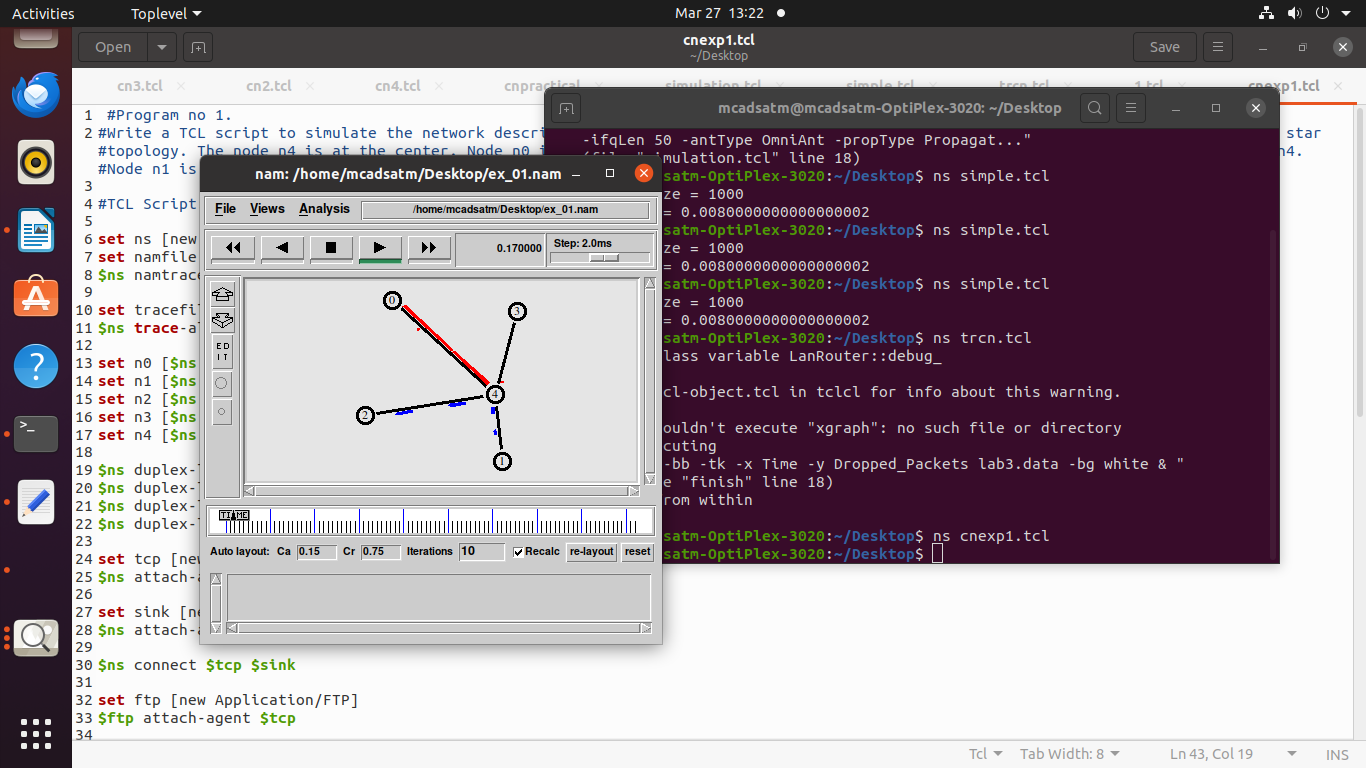
**exec nam ex\_01.nam &**

**exit 0**

**}**

**$ns at 10.0 "finish"**

**$ns run**



**2.** **Simulate a variety of IP based Networks using NS2**

**set ns [new Simulator]**

**set tracefile [open ip\_simulation.tr w]**

**$ns trace-all $tracefile**

**set namfile [open ip\_simulation.nam w]**

**$ns namtrace-all $namfile**

**set val(bw) 10Mb;**

**set val(delay) 10ms ;**

**set val(numnodes) 5 ;**

**for {set i 0} {$i < $val(numnodes)} {incr i} {**

**set node\_($i) [$ns node]**

**}**

**$ns duplex-link $node\_(0) $node\_(1) $val(bw) $val(delay) DropTail**

**$ns duplex-link $node\_(1) $node\_(2) $val(bw) $val(delay) DropTail**

**$ns duplex-link $node\_(2) $node\_(3) $val(bw) $val(delay) DropTail**

**$ns duplex-link $node\_(3) $node\_(4) $val(bw) $val(delay) DropTail**

**$ns rtproto DV;**

**set tcp [new Agent/TCP]**

**$ns attach-agent $node\_(0) $tcp**

**set sink [new Agent/TCPSink]**

**$ns attach-agent $node\_(4) $sink**

**$ns connect $tcp $sink**

**set ftp [new Application/FTP]**

**$ftp attach-agent $tcp**

**$ns at 1.0 "$ftp start"**

**$ns at 10.0 "$ftp stop"**

**$ns at 11.0 "finish"**

**proc finish {} {**

**global ns tracefile namfile**

**$ns flush-trace**

**close $tracefile**

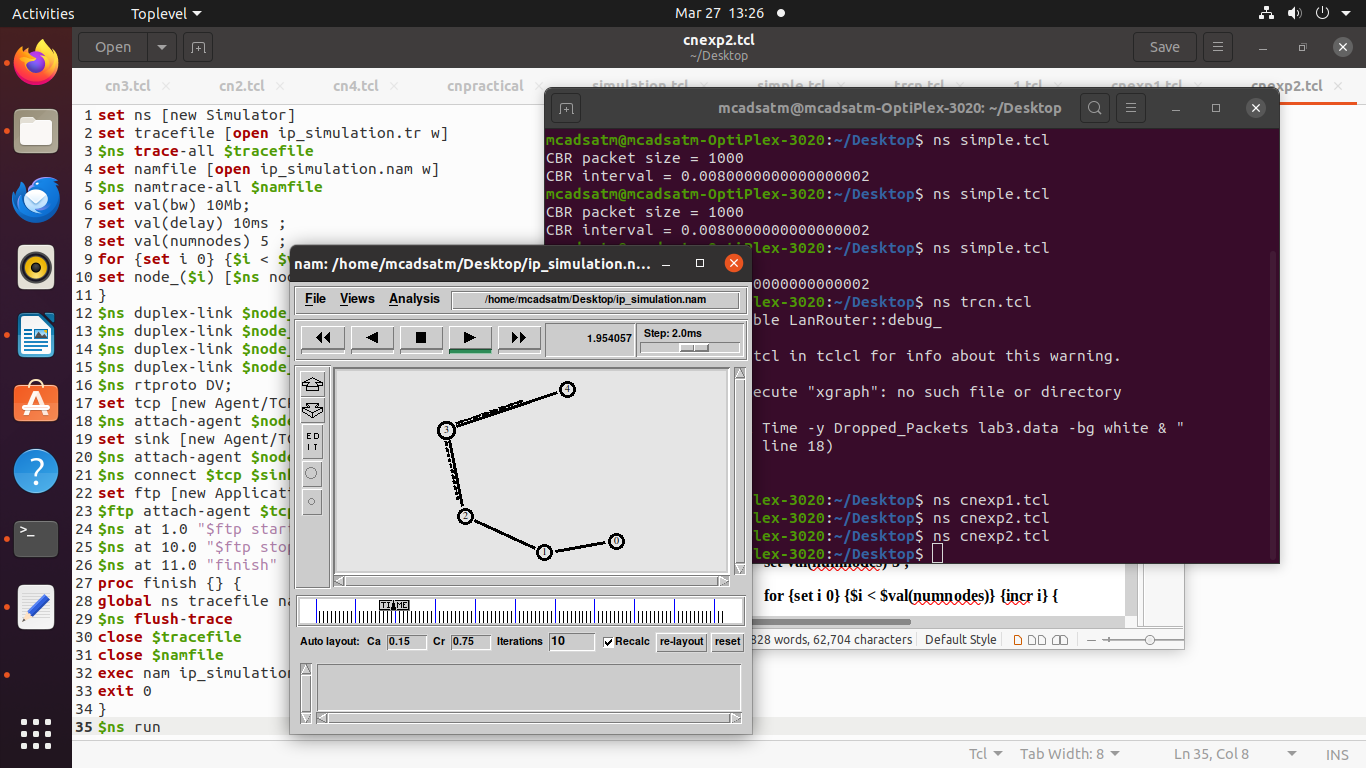
**close $namfile**

**exec nam ip\_simulation.nam &**

**exit 0**

**}**

**$ns run**

****

**3.** **Develop a program to compute CRC code for the polynomials CRC-12, CRC-16 and CRC CCIP**

**Theory**  
CRC method can detect a single burst of length n, since only one bit per column will be changed, a burst of length n+1 will pass undetected, if the first bit is inverted, the last bit is inverted and all other bits are correct. If the block is badly garbled by a long burst or by multiple shorter burst, the probability that any of the n columns will have the correct parity that is 0.5. so the probability of a bad block being expected when it should not be 2 power(-n). This scheme sometimes is known as Cyclic Redundancy Code.

**Program Algorithm/Flowchart:**

Begin

Step 1:Declare I, j , fr[8], dupfr[11], recfr[11], tlen, flag, gen[4], genl, frl, rem[4] as integer

Step 2: initialize frl=8 and genl=4

Step 3: initialize i=0

Step 4: Repeat step(5to7) until i<frl

Step 5: read fr[i]

Step 6: dupfr[i]=fr[i]

Step 7: increment i

Step 8: initialize i=0

Step 9: repeat step(10to11) until i<genl

Step 10: read gen[i]

Step 11: increment i

Step 12: tlen=frl+genl-1

Step 13: initialize i=frl

Step 14: Repeat step(15to16) until i<tlen

Step 15: dupfr[i]=0

Step 16: increment i

Step 17: call the function remainder(dupfr)

Step 18: initialize i=0

Step 19: repeat step(20 to 21) until j<genl

Step 20: recfr[i]=rem[j]

Step 21: increment I and j

Step 22: call the function remainder(dupfr)

Step 23: initialize flag=0 and i=0

Step 24: Repeat step(25to28) until i<4

Step 25: if rem[i]!=0 then

Step 26: increment flag

Step 27: end if

Step 28: increment i

Step 29: if flag=0 then

Step 25: print frame received correctly

Step 25: else

Step 25: print the received frame is wrong

End

Function: Remainder(int fr[])

Begin

Step 1: Declare k,k1,I,j as integer

Step 2: initialize k=0;

Step 3: repeat step(4 to 14) until k< frl

Step 4: if ((fr[k] == 1) then

Step 5: k1=k

Step 6: initialize i=0, j=k

Step 7: repeat step(8 to 9) until i< genl

Step 8: rem[i] =fr[j] exponential gen[i]

Step 9: increment I and j

Step 10: initialize I = 0

Step 11: repeat step(12to13) until I <genl

Step 12: fr[k1] = rem[i]

Step 13: increment k1 and i

Step 14: end if

End

**Program Code**:

**// Program for Cyclic Redundancy Check**

**#include<stdio.h>**

**int gen[4],genl,frl,rem[4];**

**void main()**

**{**

**int i,j,fr[8],dupfr[11],recfr[11],tlen,flag;**

**frl=8; genl=4;**

**printf("Enter frame:");**

**for(i=0;i<frl;i++)**

**{**

**scanf("%d",&fr[i]);**

**dupfr[i]=fr[i];**

**}**

**printf("Enter generator:");**

**for(i=0;i<genl;i++)**

**scanf("%d",&gen[i]);**

**tlen=frl+genl-1;**

**for(i=frl;i<tlen;i++)**

**{**

**dupfr[i]=0;**

**}**

**remainder(dupfr);**

**for(i=0;i<frl;i++)**

**{**

**recfr[i]=fr[i];**

**}**

**for(i=frl,j=1;j<genl;i++,j++)**

**{**

**recfr[i]=rem[j];**

**}**

**remainder(recfr);**

**flag=0;**

**for(i=0;i<4;i++)**

**{**

**if(rem[i]!=0)**

**flag++;**

**}**

**if(flag==0)**

**{**

**printf("frame received correctly");**

**}**

**else**

**{**

**printf("the received frame is wrong");**

**}**

**}**

**remainder(int fr[])**

**{**

**int k,k1,i,j;**

**for(k=0;k<frl;k++)**

**{**

**if(fr[k]==1)**

**{**

**k1=k;**

**for(i=0,j=k;i<genl;i++,j++)**

**{**

**rem[i]=fr[j]^gen[i];**

**}**

**for(i=0;i<genl;i++)**

**{**

**fr[k1]=rem[i];**

**k1++;**

**}**

**}**

**}**

**}**

**Program Output:**

**Enter frame: MLRITM**

**Enter generator: frame received correctly**

**------------------**

**(program exited with code: 24)**

**Press return to continue**

**4.** **Develop a simple data link layer that performs the flow control using the sliding window protocol, and loss recovery using the Go-Back-N mechanism**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**#include <time.h>**

**#define MAX\_FRAME 10 // Max number of frames to send**

**#define WINDOW\_SIZE 4 // Sliding window size**

**#define TIMEOUT 3 // Timeout in seconds for retransmission**

**// Simulate a frame with a sequence number and data (for simplicity, just a number)**

**typedef struct {**

**int seq\_num; // Sequence number of the frame**

**int data; // Data (just a number for simplicity)**

**} Frame;**

**// Function to simulate packet loss with a certain probability**

**int packet\_loss\_simulation() {**

**return rand() % 2; // 50% chance of packet loss**

**}**

**// Sender function that sends frames with sliding window protocol**

**void sender(Frame frames[], int total\_frames, int window\_size) {**

**int next\_frame\_to\_send = 0;**

**int frame\_expected = 0;**

**int ack\_received = 0;**

**while (frame\_expected < total\_frames) {**

**// Send all frames in the current window**

**while (next\_frame\_to\_send < total\_frames && next\_frame\_to\_send < frame\_expected + window\_size) {**

**printf("Sender: Sending frame %d\n", frames[next\_frame\_to\_send].seq\_num);**

**// Simulate packet loss**

**if (packet\_loss\_simulation()) {**

**printf("Sender: Frame %d lost!\n", frames[next\_frame\_to\_send].seq\_num);**

**} else {**

**printf("Sender: Frame %d sent successfully.\n", frames[next\_frame\_to\_send].seq\_num);**

**}**

**next\_frame\_to\_send++;**

**}**

**// Simulate waiting for acknowledgment (ACK)**

**sleep(TIMEOUT);**

**// If we haven't received an ACK, resend the unacknowledged frames**

**if (next\_frame\_to\_send > frame\_expected) {**

**printf("Sender: Timeout, resending frames starting from %d\n", frame\_expected);**

**next\_frame\_to\_send = frame\_expected; // Resend the frames in the window**

**}**

**}**

**}**

**// Receiver function that acknowledges received frames**

**void receiver(Frame frames[], int total\_frames, int window\_size) {**

**int expected\_frame = 0;**

**while (expected\_frame < total\_frames) {**

**// Simulate receiving frames (only accepting the ones with correct sequence number)**

**printf("Receiver: Waiting for frame %d\n", expected\_frame);**

**// Simulate receiving a frame (just the correct one for simplicity)**

**printf("Receiver: Frame %d received.\n", expected\_frame);**

**expected\_frame++;**

**// Acknowledge the frame**

**printf("Receiver: Acknowledging frame %d\n", expected\_frame - 1);**

**}**

**}**

**int main() {**

**srand(time(NULL)); // Initialize random seed for simulation**

**// Create a list of frames (max of MAX\_FRAME frames)**

**Frame frames[MAX\_FRAME];**

**for (int i = 0; i < MAX\_FRAME; i++) {**

**frames[i].seq\_num = i;**

**frames[i].data = i; // Just using the index as data**

**}**

**// Simulate the sender and receiver**

**printf("=== Sender starts ===\n");**

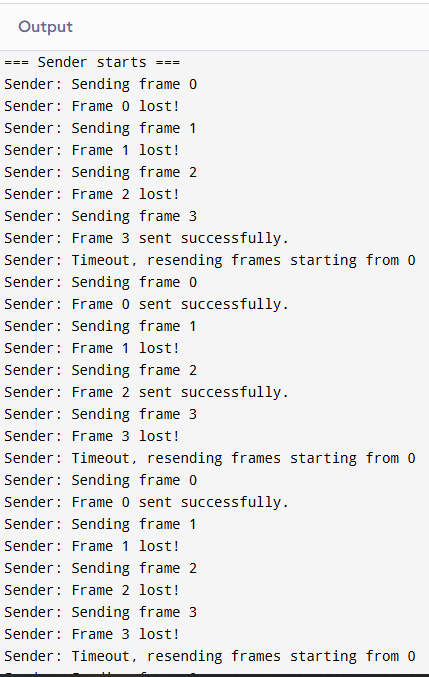
**sender(frames, MAX\_FRAME, WINDOW\_SIZE);**

**printf("=== Receiver starts ===\n");**

**receiver(frames, MAX\_FRAME, WINDOW\_SIZE);**

**return 0;**

**}**



**5.Evaluates the existing network protocol performance using NS2**

**#Create a simulator object**

**set ns [new Simulator]**

**#Define different colors for data flows (for NAM)**

**$ns color 1 Blue**

**$ns color 2 Red**

**#Open the NAM trace file**

**set nf [open out.nam w]**

**$ns namtrace-all $nf**

**#Define a 'finish' procedure**

**proc finish {} {**

**global ns nf**

**$ns flush-trace**

**#Close the NAM trace file**

**close $nf**

**#Execute NAM on the trace file**

**exec nam out.nam &**

**exit 0**

**}**

**#Create four nodes**

**set n0 [$ns node]**

**set n1 [$ns node]**

**set n2 [$ns node]**

**set n3 [$ns node]**

**#Create links between the nodes**

**$ns duplex-link $n0 $n2 2Mb 10ms DropTail**

**$ns duplex-link $n1 $n2 2Mb 10ms DropTail**

**$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail**

**#Set Queue Size of link (n2-n3) to 10**

**$ns queue-limit $n2 $n3 10**

**#Give node position (for NAM)**

**$ns duplex-link-op $n0 $n2 orient right-down**

**$ns duplex-link-op $n1 $n2 orient right-up**

**$ns duplex-link-op $n2 $n3 orient right**

**#Monitor the queue for link (n2-n3). (for NAM)**

**$ns duplex-link-op $n2 $n3 queuePos 0.5**

**#Setup a TCP connection**

**set tcp [new Agent/TCP]**

**$tcp set class\_ 2**

**$ns attach-agent $n0 $tcp**

**set sink [new Agent/TCPSink]**

**$ns attach-agent $n3 $sink**

**$ns connect $tcp $sink**

**$tcp set fid\_ 1**

**#Setup a FTP over TCP connection**

**set ftp [new Application/FTP]**

**$ftp attach-agent $tcp**

**$ftp set type\_ FTP**

**#Setup a UDP connection**

**set udp [new Agent/UDP]**

**$ns attach-agent $n1 $udp**

**set null [new Agent/Null]**

**$ns attach-agent $n3 $null**

**$ns connect $udp $null**

**$udp set fid\_ 2**

**#Setup a CBR over UDP connection**

**set cbr [new Application/Traffic/CBR]**

**$cbr attach-agent $udp**

**$cbr set type\_ CBR**

**$cbr set packet\_size\_ 1000**

**$cbr set rate\_ 1mb**

**$cbr set random\_ false**

**#Schedule events for the CBR and FTP agents**

**$ns at 0.1 "$cbr start"**

**$ns at 1.0 "$ftp start"**

**$ns at 4.0 "$ftp stop"**

**$ns at 4.5 "$cbr stop"**

**#Detach tcp and sink agents (not really necessary)**

**$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"**

**#Call the finish procedure after 5 seconds of simulation time**

**$ns at 5.0 "finish"**

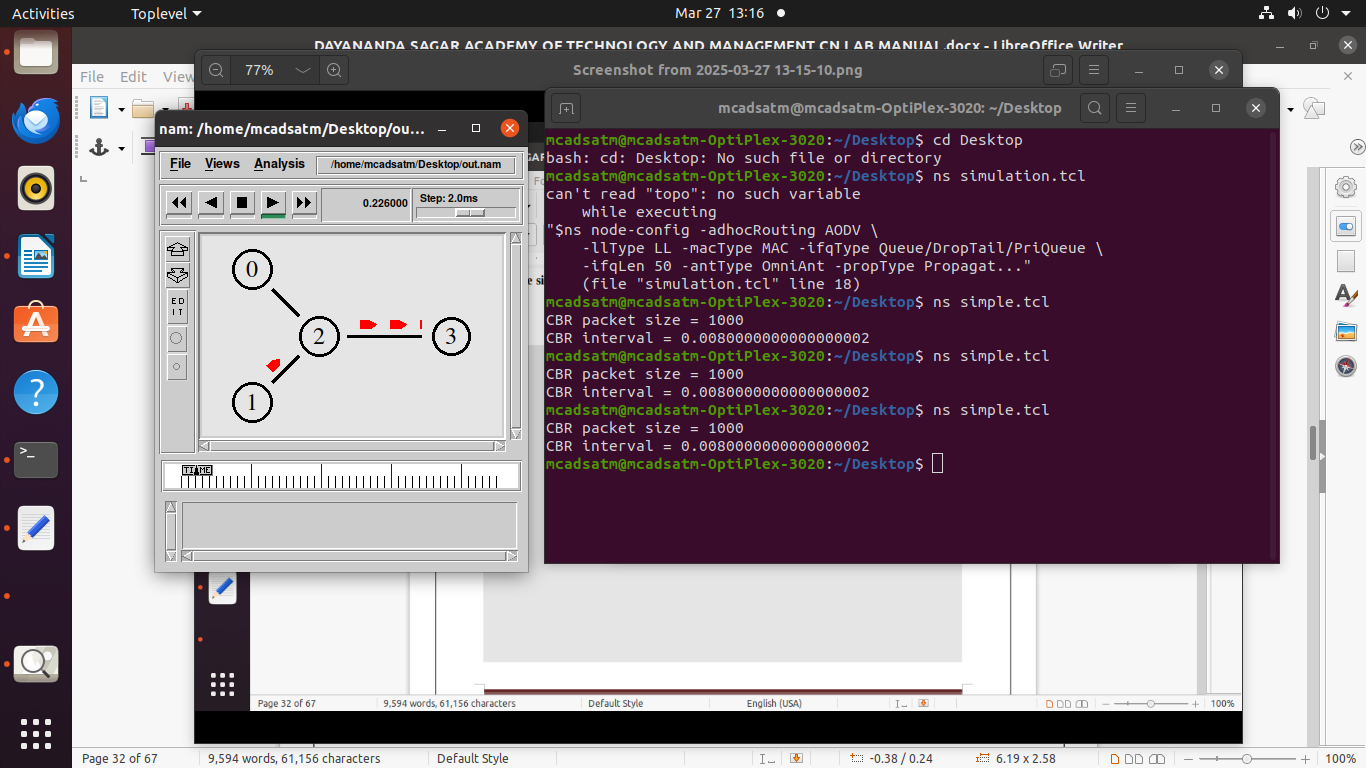
**#Print CBR packet size and interval**

**puts "CBR packet size = [$cbr set packet\_size\_]"**

**puts "CBR interval = [$cbr set interval\_]"**

**#Run the simulation**

**$ns run**



**6.** **Simulate to study the transmission of packets over Ethernet LAN and determine the number of packets drop destination**

**# Create a new instance of a Simulator named "ns"**

**set ns [new Simulator]**

**# Open a new file to store Network Animator (NAM)**

**set namfile [open lab3.nam w]**

**$ns namtrace-all $namfile**

**# Open a new file to store trace data which is used for \*Analysis\***

**set tracefile [open lab3.tr w]**

**$ns trace-all $tracefile**

**# Before we proceed, set the TCP packet size to 1500 bytes**

**Agent/TCP set packetSize\_ 1460**

**# Creating five instances of \*node\* inside the "ns" instance.**

**set n0 [$ns node]**

**set n1 [$ns node]**

**set n2 [$ns node]**

**set n3 [$ns node]**

**set n4 [$ns node]**

**$ns duplex-link $n0 $n1 1Mb 10ms DropTail**

**$ns duplex-link $n1 $n2 1Mb 10ms DropTail**

**$ns make-lan "$n2 $n3 $n4" 100Mb 1ms LL Queue/DropTail Mac/802\_3 Channel Phy/WiredPhy**

**# The link between node 1 and node 2 to behave as a faulty link**

**set errmodel [new ErrorModel]**

**$errmodel set rate\_ 0.2 ;# error rate in the link is fixed to 0.2**

**$errmodel ranvar [new RandomVariable/Uniform]**

**$errmodel drop-target [new Agent/Null]**

**$ns lossmodel $errmodel $n1 $n2**

**# Setting up TCP Source**

**set tcp [new Agent/TCP]**

**$ns attach-agent $n0 $tcp**

**# Setting up TCP Destination (also known as TCP Sink)**

**set sink [new Agent/TCPSink]**

**$ns attach-agent $n4 $sink**

**# Setup connection between TCP Source and Destination**

**$ns connect $tcp $sink**

**# Enable FTP application on TCP Source**

**set ftp [new Application/FTP]**

**$ftp attach-agent $tcp**

**set filesize [expr 4\*1024\*1024]**

**$ns at 0.0 "$ftp send $filesize"**

**# User defined procedure to terminate the simulation**

**proc finish {} {**

**global ns namfile tracefile**

**$ns flush-trace**

**close $namfile ;# Closes ex\_04.nam**

**close $tracefile ;# Closes ex\_04.tr**

**set awkCode {**

**BEGIN{}**

**{**

**if ($1 == "d" && $5 == "tcp" && $6 > 1460) {**

**count\_packets++;**

**print $2, count\_packets >> "lab3.data"**

**}**

**}**

**END{}**

**}**

**exec awk $awkCode lab3.tr**

**exec nam lab3.nam &**

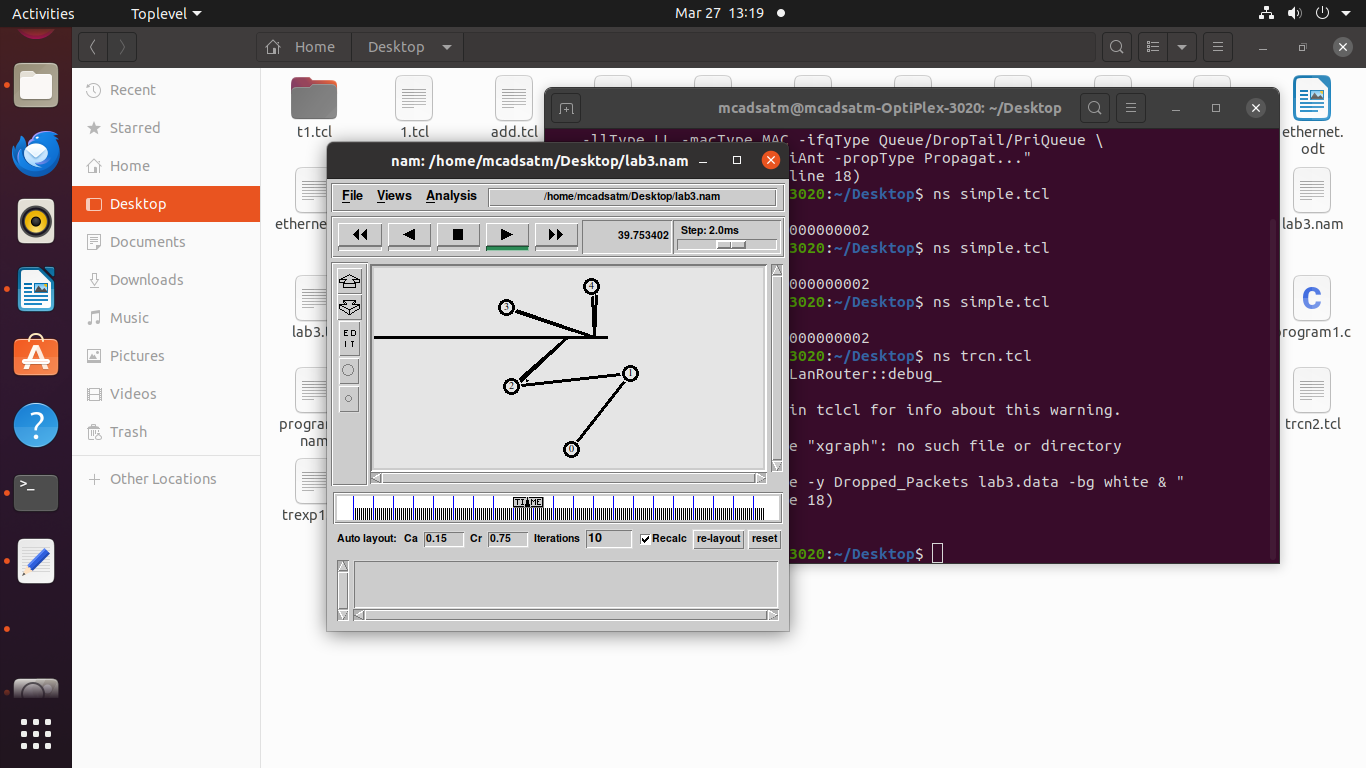
**exec xgraph -bb -tk -x Time -y Dropped\_Packets lab3.data -bg white &**

**exit 0**

**}**

**$ns at 100.0 "finish"**

**$ns run**

****

**7.** **Simulate the different types of internet traffic such as FTP and TELNET over a wired network and analyze the packet drop and packet delivery ratio in the network.**

**#TELNET**

**#===================================**

**# Simulation parameters setup**

**#===================================**

**set val(stop) 10.5 ;# time of simulation end**

**#===================================**

**# Initialization**

**#===================================**

**#Create a ns simulator**

**set ns [new Simulator]**

**#Open the NS trace file**

**set tracefile [open telnet.tr w]**

**$ns trace-all $tracefile**

**#Open the NAM trace file**

**set namfile [open telnet.nam w]**

**$ns namtrace-all $namfile**

**#===================================**

**# Nodes Definition**

**#===================================**

**#Create 6 nodes**

**set n0 [$ns node]**

**set n1 [$ns node]**

**set n2 [$ns node]**

**set n3 [$ns node]**

**set n4 [$ns node]**

**set n5 [$ns node]**

**#===================================**

**# Links Definition**

**#===================================**

**#Create links between nodes**

**$ns duplex-link $n0 $n2 100.0Mb 10ms RED**

**$ns queue-limit $n0 $n2 50**

**$ns duplex-link $n3 $n2 100.0Mb 10ms RED**

**$ns queue-limit $n3 $n2 50**

**$ns duplex-link $n1 $n2 100.0Mb 10ms RED**

**$ns queue-limit $n1 $n2 50**

**$ns duplex-link $n3 $n4 100.0Mb 10ms RED**

**$ns queue-limit $n3 $n4 50**

**$ns duplex-link $n3 $n5 100.0Mb 10ms RED**

**$ns queue-limit $n3 $n5 50**

**#Give node position (for NAM)**

**$ns duplex-link-op $n0 $n2 orient right-down**

**$ns duplex-link-op $n3 $n2 orient left**

**$ns duplex-link-op $n1 $n2 orient right-up**

**$ns duplex-link-op $n3 $n4 orient right-up**

**$ns duplex-link-op $n3 $n5 orient right-down**

**#===================================**

**# Agents Definition**

**#===================================**

**#Setup a TCP connection**

**set tcp0 [new Agent/TCP]**

**$ns attach-agent $n0 $tcp0**

**set sink3 [new Agent/TCPSink]**

**$ns attach-agent $n5 $sink3**

**$ns connect $tcp0 $sink3**

**$tcp0 set packetSize\_ 1500**

**#Setup a TCP connection**

**set tcp1 [new Agent/TCP]**

**$ns attach-agent $n4 $tcp1**

**set sink2 [new Agent/TCPSink]**

**$ns attach-agent $n1 $sink2**

**$ns connect $tcp1 $sink2**

**$tcp1 set packetSize\_ 1500**

**#===================================**

**# Applications Definition**

**#===================================**

**#Setup a TELNET Application over TCP connection**

**set telnet0 [new Application/Telnet]**

**$telnet0 attach-agent $tcp0**

**$ns at 1.0 "$telnet0 start"**

**$ns at 10.0 "$telnet0 stop"**

**$telnet0 set interval\_ 0.002**

**$telnet0 set type\_ Telnet**

**#Setup an FTP Application over a TCP connection**

**set ftp1 [new Application/FTP]**

**$ftp1 attach-agent $tcp1**

**$ns at 1.0 "$ftp1 start"**

**$ns at 10.0 "$ftp1 stop"**

**#===================================**

**# Termination**

**#===================================**

**#Define a 'finish' procedure**

**proc finish {} {**

**global ns tracefile namfile**

**$ns flush-trace**

**close $tracefile**

**close $namfile**

**exec nam telnet.nam &**

**exit 0**

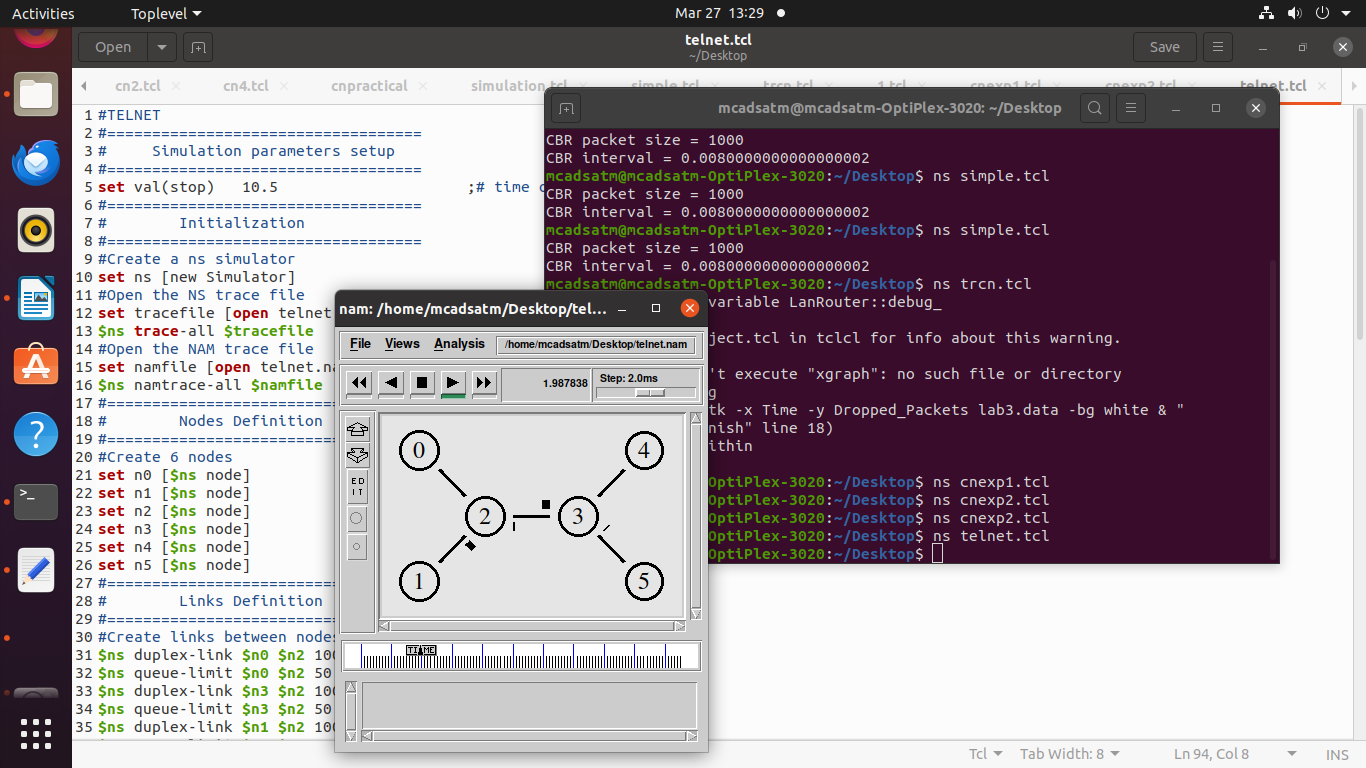
**}**

**$ns at $val(stop) "$ns nam-end-wireless $val(stop)"**

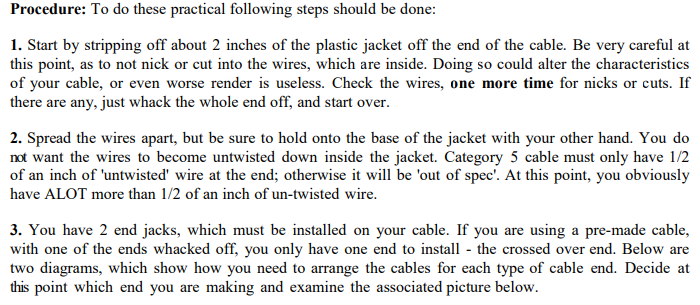
**$ns at $val(stop) "finish"**

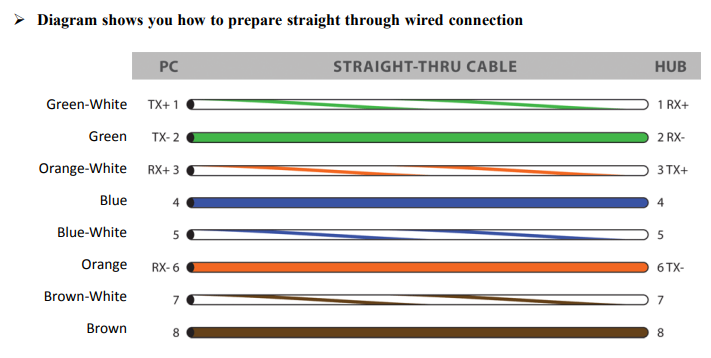
**$ns at $val(stop) "puts \"done\" ; $ns halt"**

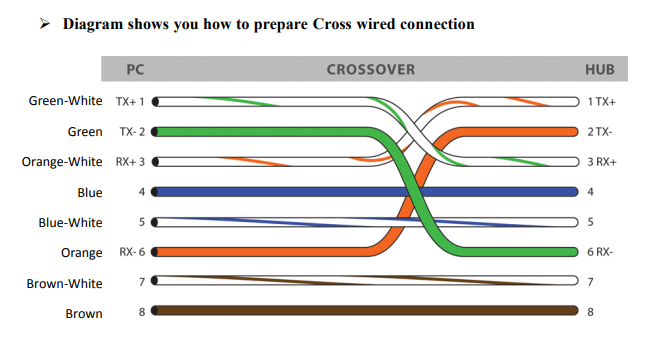
**$ns run**

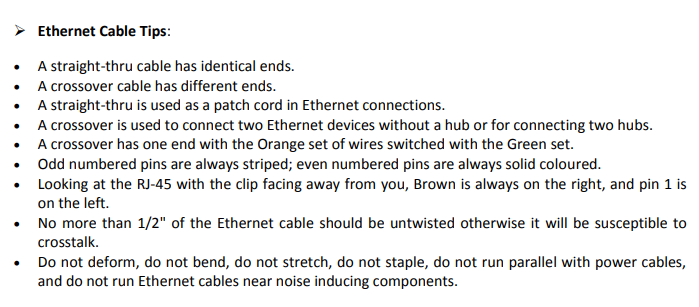
****

**8.** **Implement the connectivity of Network cables and practically implement the cross-wired cable and straight-through cable using a clamping tool.**









**9.** **Configuring a Cisco Router as a DHCP Server**

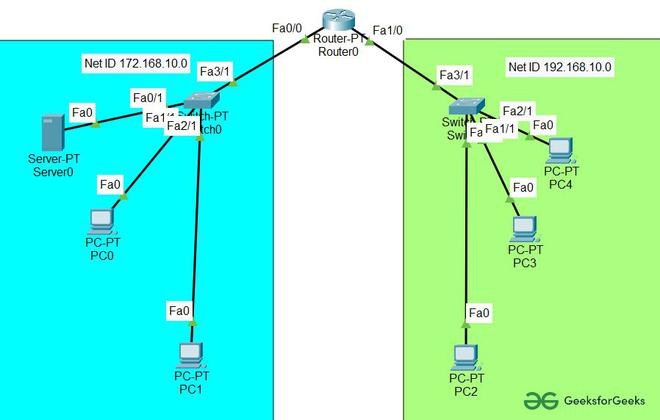
**DHCP is a network management protocol used in networks to dynamically assign IP addresses and other network configuration information like default gateway, mask, DNS server address, etc. It is an application layer protocol.**

**Steps to Configure and Verify DHCP Server in Cisco Packet Tracer:**

**Step 1: First, open the cisco packet tracer desktop and select the devices given below:**

| **S.NO** | **Device** | **Model-Name** | **Unit** |
| --- | --- | --- | --- |
| **1.** | **PC** | **PC** | **5** |
| **2.** | **Switch** | **PT-Switch** | **2** |
| **3.** | **Router** | **PT-Router** | **1** |
| **4.** | **Server** | **Server-PT** | **1** |

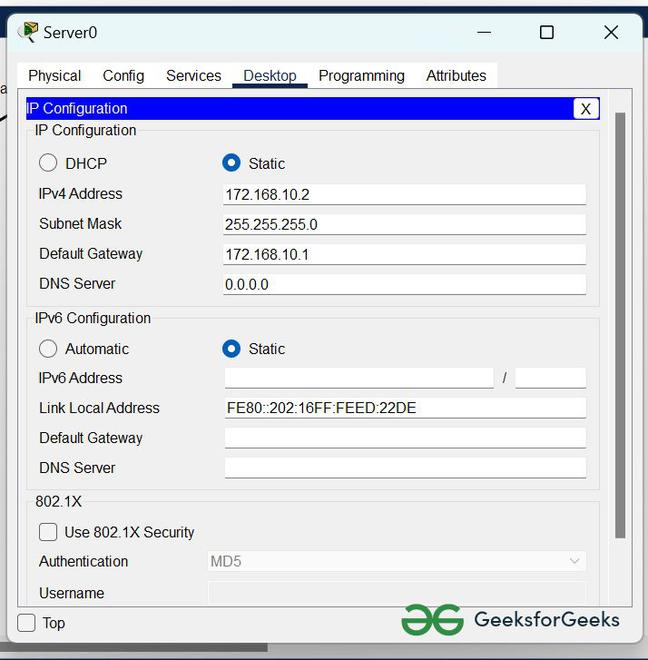
* **Now create a network topology as shown below the image.**
* **Use an Automatic connecting cable to connect the devices with others.**



**Step 2: Configure the Server with IPv4 address and Subnet Mask according to the Data given above.**

* **To assign an IP address in Server, click on Server-PT.**
* **Then, go to desktop and IP configuration and there you will find IPv4 configuration.**
* **Add IPv4 address, subnet mask, and Default Gateway.**

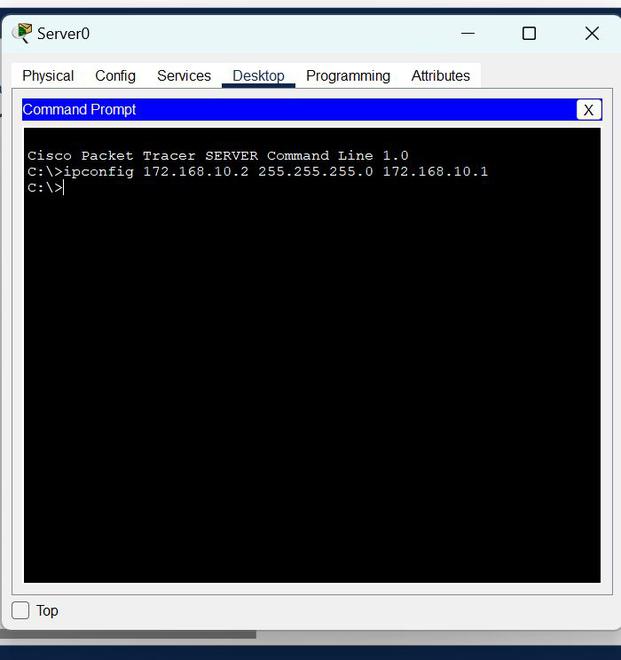
| **Parameters** | **Address value** |
| --- | --- |
| **IPv4 Address** | **172.168.10.2** |
| **Subnet Mask** | **255.255.255.0** |
| **Default-Gateway** | **172.168.10.1** |



**2. Assigning IP address using the ipconfig command.**

* **We can also assign an IP address with the help of a command.**
* **Go to the command prompt of the server**
* **Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)**

**example: ipconfig 172.168.10.2 255.255.255.0 172.168.10.1**

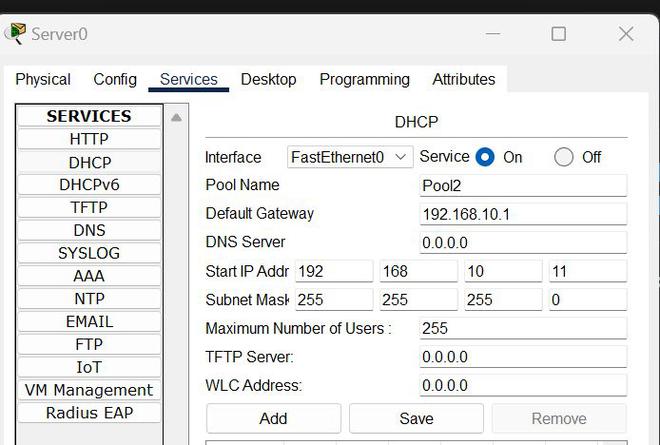


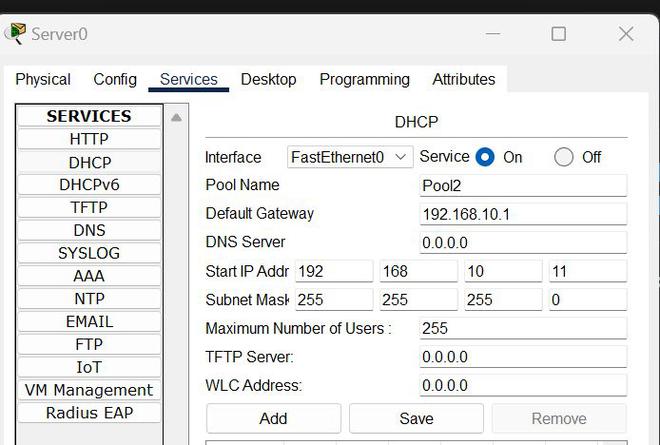
**Step 3: Configuring the DHCP server.**

**To configure the DHCP server first,**

* **Click on Server then, Go to services.**
* **Click on DHCP and turn on the services and, configure the DHCP server with the help of the data given below.**
  + **Delete the default values of Start IP Address and subnet Mask then save the info.**
  + **Create two new pools.**

**POOL1 and POOL2 and fill the data as shown in the images below.**





**Step 4: Configuring Router with IPv4 Address and Subnet Mask.**

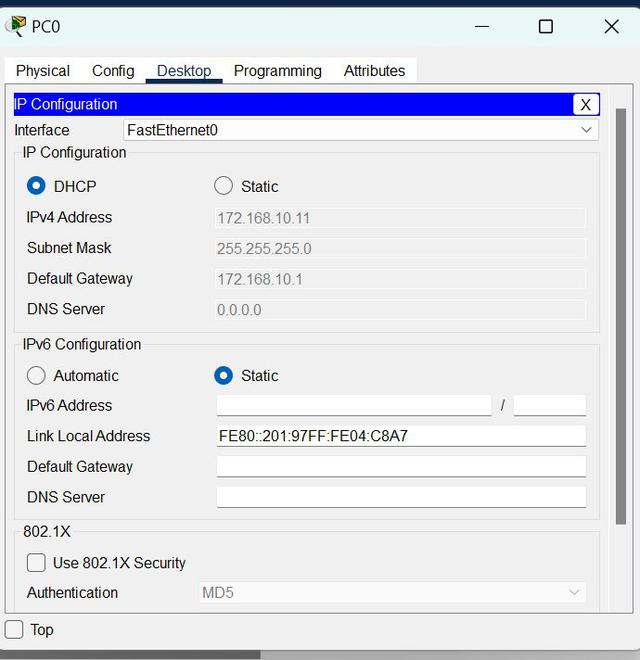
**IP Addressing Table for Router:**

| **S.NO** | **Device** | **Interface** | **IPv4 Address** | **Subnet Mask** |
| --- | --- | --- | --- | --- |
| **1.** | **router0** | **FastEthernet0/0** | **172.168.10.1** | **255.255.255.0** |
| **FastEthernet0/1** | **192.168.10.1** | **255.255.255.0** |

* **To assign an IP address in router0, click on router0.**
* **Then, go to config and then Interfaces, and make sure to turn on the ports.**
* **Then, configure the IP address in FastEthernet according to IP addressing Table.**
* **Fill IPv4 address and subnet mask.**

**Step 5: Configuring the PCs and changing the IP configuration.**

* **To assign an IP address in PC0, click on PC0.**
* **Then, go to desktop and IP configuration and there you will find IPv4 configuration.**
* **Change its state from static to DHCP.**
* **It will automatically fetch the data and configure itself.**



* **Repeat the same procedure with other PCs to configure them thoroughly.**

**10.** **Applications using TCP sockets like:**

**Echo client and echo server**

**Chat**

**File Transfer**

**#server.c**

**#include<sys/types.h> //including data types**

**#include<sys/socket.h> //listening**

**#include<netdb.h> //provide network db operations**

**#include<fcntl.h> //open the connection**

**#include<unistd.h> //close the connection**

**#include<stdio.h>**

**#include<string.h>**

**int main()**

**{**

**char str[100]; //receiving data**

**char sendline1[100];**

**bzero(sendline1,100);**

**int listen\_fd,comm\_fd;**

**struct sockaddr\_in servaddr; //IPv4\_IPv6, IP add, PORT add**

**//creating a socket, address family inet(using v4),use INET6 for v6**

**listen\_fd=socket(AF\_INET,SOCK\_STREAM,0);**

**//SOCK\_STREAM = TCP, SOCK\_DGRAM = UDP**

**//0 is default protocol not flag operation**

**bzero(str,100); //want to make 100 bytes 0**

**bzero(&servaddr,sizeof(servaddr)); //initialize all values with 0**

**//defining server address..passing those 3 parameters**

**servaddr.sin\_family=AF\_INET; //IPv4**

**servaddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);**

**//server will connect with any IP addr**

**//htonl is used to convert host byte into network byte**

**servaddr.sin\_port=htons(21000);**

**//0 to 65,535....first 1024 ports are fixed**

**//binding server addr with socket**

**bind(listen\_fd,(struct sockaddr \*)&servaddr,sizeof(servaddr));**

**//listening**

**listen(listen\_fd,5); //5 clients can be connected to the server**

**//to establish connection with client...like a pipeline**

**comm\_fd=accept(listen\_fd,(struct sockaddr \*)NULL,NULL);**

**//client initializing data**

**while(1){**

**recv(comm\_fd,str,100,0); //hardcoded 100 because we do not know length of incoming stream**

**printf("Client: %s",str);**

**printf("Your message: ");**

**fgets(sendline1,100,stdin);**

**send(comm\_fd,sendline1,strlen(sendline1),0);**

**bzero(sendline1,100);**

**//0 is flag operation...tells abt successfull and unsuccessfull op**

**}**

**close(comm\_fd);**

**}**

**#client.c**

**#include<sys/types.h>**

**#include<sys/socket.h>**

**#include<netdb.h>**

**#include<arpa/inet.h>**

**#include<stdio.h>**

**#include<string.h>**

**int main()**

**{**

**int sockfd;**

**char sendline[100];**

**char recvline[100];**

**struct sockaddr\_in servaddr;**

**sockfd=socket(AF\_INET,SOCK\_STREAM,0);**

**bzero(sendline,100);**

**bzero(recvline,100);**

**bzero(&servaddr,sizeof(servaddr));**

**servaddr.sin\_family=AF\_INET;**

**servaddr.sin\_port=htons(21000);**

**servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");**

**connect(sockfd,(struct sockaddr \*)&servaddr,sizeof(servaddr));**

**while(1){**

**printf("Your Message: ");**

**fgets(sendline,100,stdin);**

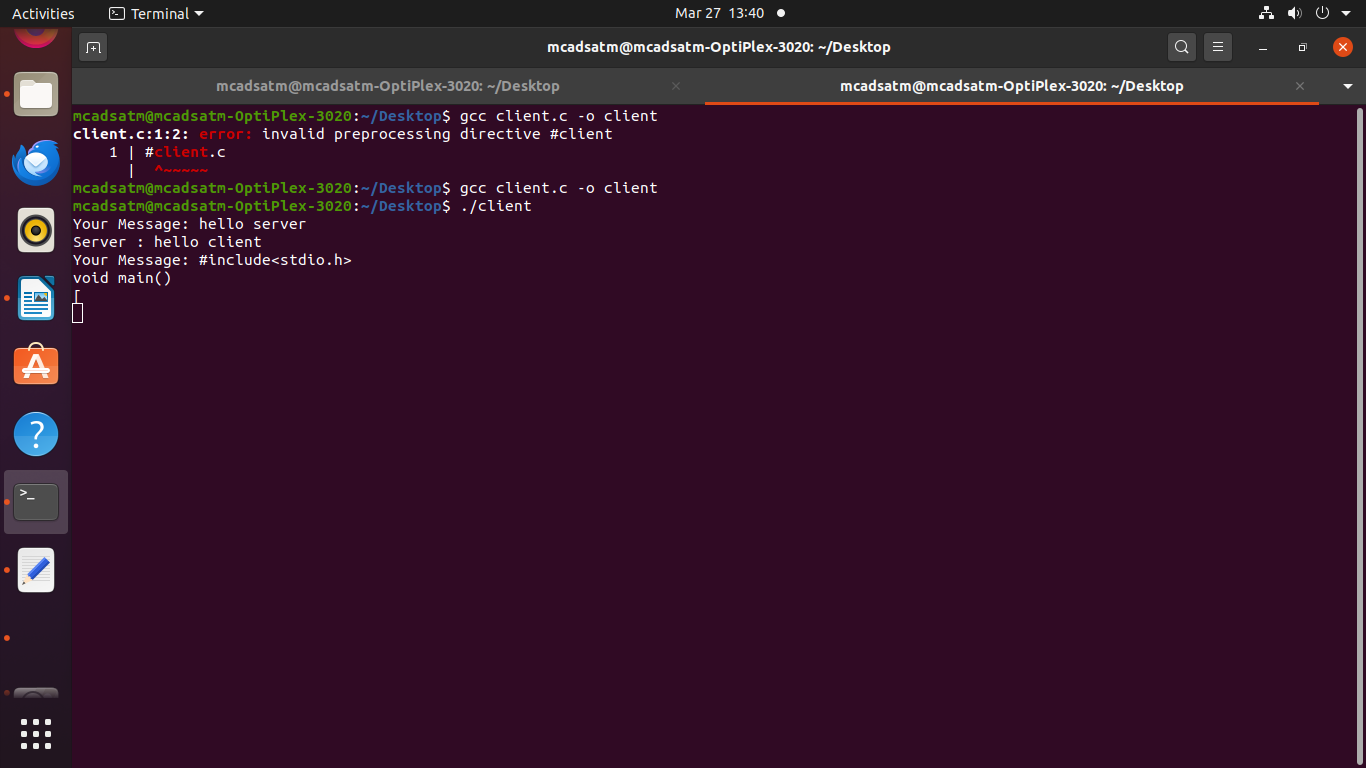
**send(sockfd,sendline,strlen(sendline),0);**

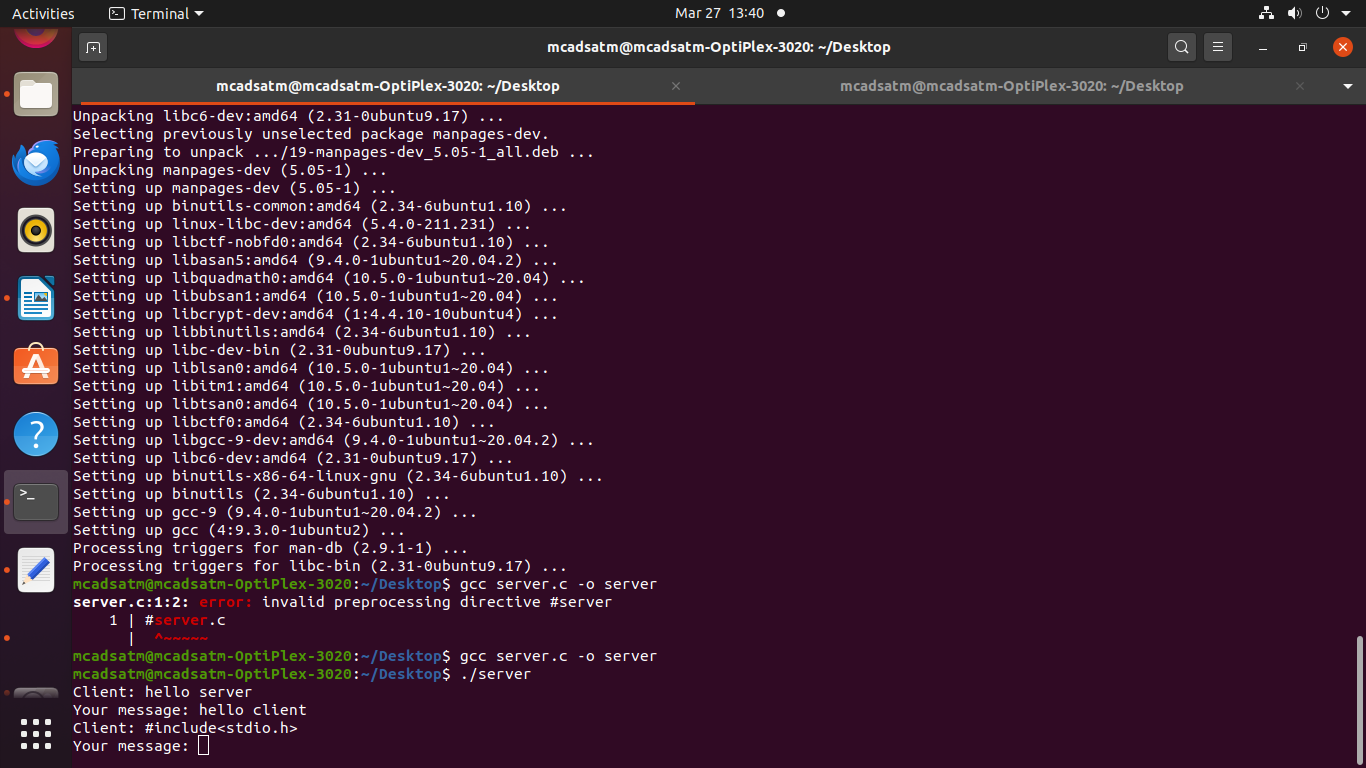
**recv(sockfd,recvline,100,0);**

**printf("Server : %s",recvline);**

**}**

**}**





**11.** **Simulation of Distance Vector/ Link State Routing algorithm**

**#Distance**

**# Create the simulator object**

**set ns [new Simulator]**

**# Open a trace file to log the simulation**

**set tracefile [open dsdv\_trace.tr w]**

**$ns trace-all $tracefile**

**# Open a NAM file to visualize the simulation**

**set namfile [open dsdv\_simulation.nam w]**

**$ns namtrace-all $namfile**

**# Define the finish procedure**

**proc finish {} {**

**global ns tracefile namfile**

**$ns flush-trace**

**close $tracefile**

**close $namfile**

**exec nam dsdv\_simulation.nam &**

**exit 0**

**}**

**# Create nodes**

**set n0 [$ns node]**

**set n1 [$ns node]**

**set n2 [$ns node]**

**set n3 [$ns node]**

**set n4 [$ns node]**

**# Create links between the nodes (bandwidth, delay, queuing mechanism)**

**$ns duplex-link $n0 $n1 2Mb 10ms DropTail**

**$ns duplex-link $n1 $n2 2Mb 10ms DropTail**

**$ns duplex-link $n2 $n3 2Mb 10ms DropTail**

**$ns duplex-link $n3 $n4 2Mb 10ms DropTail**

**# Enable the DSDV routing protocol (Distance Vector Routing)**

**set opt(adhocRouting) DSDV**

**# Create a UDP agent and attach it to node 0 (source)**

**set udp0 [new Agent/UDP]**

**$ns attach-agent $n0 $udp0**

**# Create a Null agent and attach it to node 4 (destination)**

**set null0 [new Agent/Null]**

**$ns attach-agent $n4 $null0**

**# Connect the UDP agent to the null agent (unicast communication)**

**$ns connect $udp0 $null0**

**# Create CBR (Constant Bit Rate) traffic over the UDP connection**

**set cbr [new Application/Traffic/CBR]**

**$cbr set packetSize\_ 512**

**$cbr set interval\_ 0.2**

**$cbr attach-agent $udp0**

**# Schedule the CBR traffic to start and stop**

**$ns at 1.0 "$cbr start"**

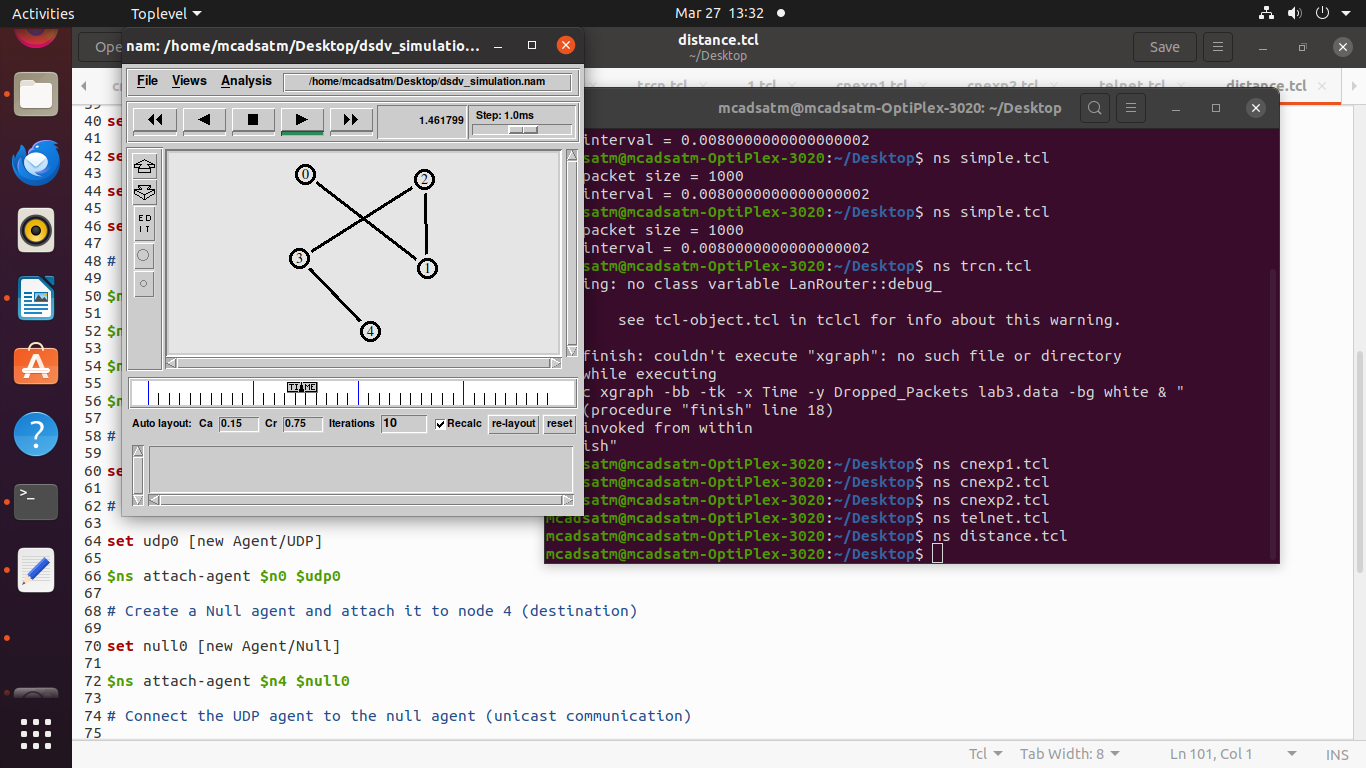
**$ns at 4.0 "$cbr stop"**

**# Schedule the end of the simulation at 5 seconds**

**$ns at 5.0 "finish"**

**# Run the simulation**

**$ns run**

****

**OPEN ENDED PROGRAMS**

**1. Implement the data link layer framing methods such as character, character-stuffing and bit stuffing.**

**Theory**  
Security and Error detection are the most prominent features that are to be provided by any application which transfers data from one end to the other end. One of such a mechanism in tracking errors which may add up to the original data during transfer is known as Stuffing. It is of two types namely Bit Stuffing and the other Character Stuffing. Coming to the Bit Stuffing, 01111110 is appended within the original data while transfer of it. The following program describes how it is stuffed at the sender end and de-stuffed at the reciever end.

**Bit stuffing:**

**Program**:

#include<stdio.h>

#include<string.h>

int main()

{

int a[20],b[30],i,j,k,count,n;

printf("Enter frame size (Example: 8):");

scanf("%d",&n);

printf("Enter the frame in the form of 0 and 1 :");

for(i=0; i<n; i++)

scanf("%d",&a[i]);

i=0;

count=1;

j=0;

while(i<n)

{

if(a[i]==1)

{

b[j]=a[i];

for(k=i+1; a[k]==1 && k<n && count<5; k++)

{

j++;

b[j]=a[k];

count++;

if(count==5)

{

j++;

b[j]=0;

}

i=k;

}

}

else

{

b[j]=a[i];

}

i++;

j++;

}

printf("After Bit Stuffing :");

for(i=0; i<j; i++)

printf("%d",b[i]);

return 0;

}  
  
**OUTPUT:**

Enter frame size (Example: 8):12

Enter the frame in the form of 0 and 1 :0 1 0 1 1 1 1 1 1 0 0 1

After Bit Stuffing :0101111101001

**Character Stuffing**

#include<stdio.h>

#include<string.h>

int main()

{

char a[30], fs[50] = " ", t[3], sd, ed, x[3], s[3], d[3], y[3];

int i, j, p = 0, q = 0;

printf("Enter characters to be stuffed:");

scanf("%s", a);

printf("\nEnter a character that represents starting delimiter:");

scanf(" %c", &sd);

printf("\nEnter a character that represents ending delimiter:");

scanf(" %c", &ed);

x[0] = s[0] = s[1] = sd;

x[1] = s[2] = '\0';

y[0] = d[0] = d[1] = ed;

d[2] = y[1] = '\0';

strcat(fs, x);

for(i = 0; i < strlen(a); i++)

{

t[0] = a[i];

t[1] = '\0';

if(t[0] == sd)

strcat(fs, s);

else if(t[0] == ed)

strcat(fs, d);

else

strcat(fs, t);

}

strcat(fs, y);

printf("\n After stuffing:%s", fs);

}

*Output:*

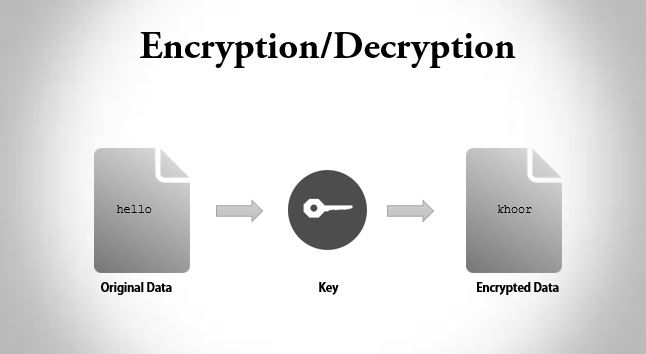
Enter characters to be stuffed: goodday

Enter a character that represents starting delimiter: d

Enter a character that represents ending delimiter: g

After stuffing: dggooddddayg.

**2.** Implement data encryption and data decryption



#include <stdio.h>

int main()

{

int i, x;

char str[100];

printf("\nPlease enter a string:\t");

gets(str);

printf("\nPlease choose following options:\n");

printf("1 = Encrypt the string.\n");

printf("2 = Decrypt the string.\n");

scanf("%d", &x);

//using switch case statements

switch(x)

{

case 1:

for(i = 0; (i < 100 && str[i] != '\0'); i++)

str[i] = str[i] + 3; //the key for encryption is 3 that is added to ASCII value

printf("\nEncrypted string: %s\n", str);

break;

case 2:

for(i = 0; (i < 100 && str[i] != '\0'); i++)

str[i] = str[i] - 3; //the key for encryption is 3 that is subtracted to ASCII value

printf("\nDecrypted string: %s\n", str);

break;

default:

printf("\nError\n");

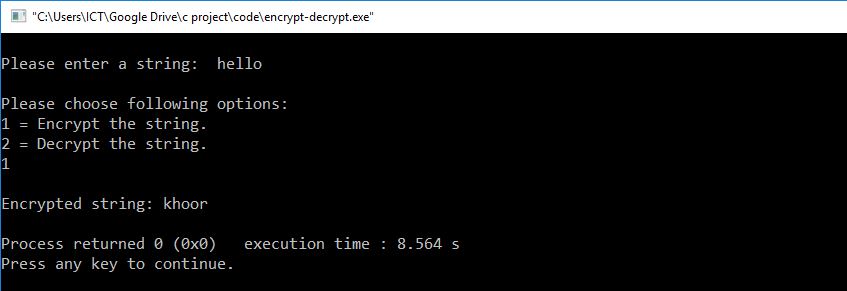
}

return 0;

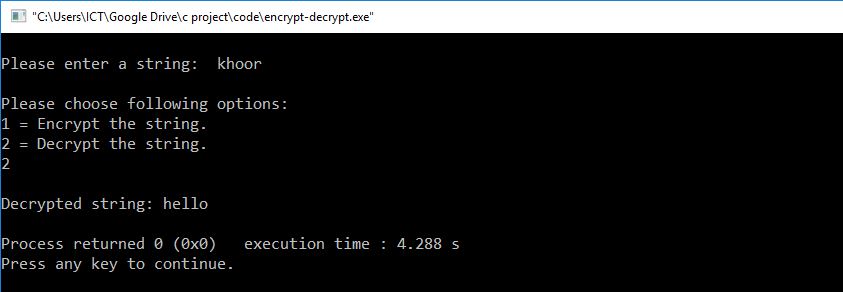
}

OUT PUT

Encryption



Decryption



**3.Implement the code to simulate ARP/RARP protocol**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_DEVICES 5

#define MAX\_IP\_LEN 16 // Maximum length for an IPv4 address (xxx.xxx.xxx.xxx)

#define MAX\_MAC\_LEN 18 // Maximum length for a MAC address (xx:xx:xx:xx:xx:xx)

// Arrays to store IP and MAC addresses of devices

char ip\_addresses[MAX\_DEVICES][MAX\_IP\_LEN] = {

"192.168.1.1", "192.168.1.2", "192.168.1.3", "192.168.1.4", "192.168.1.5"

};

char mac\_addresses[MAX\_DEVICES][MAX\_MAC\_LEN] = {

"00:14:22:01:23:45", "00:14:22:01:23:46", "00:14:22:01:23:47", "00:14:22:01:23:48", "00:14:22:01:23:49"

};

// Function to find the MAC address for a given IP address

void find\_mac\_address\_by\_ip(const char\* target\_ip) {

int found = 0;

for (int i = 0; i < MAX\_DEVICES; i++) {

if (strcmp(ip\_addresses[i], target\_ip) == 0) {

printf("ARP Reply: IP %s has MAC address %s\n", target\_ip, mac\_addresses[i]);

found = 1;

break;

}

}

if (!found) {

printf("ARP Reply: IP %s not found on the network.\n", target\_ip);

}

}

// Function to find the IP address for a given MAC address

void find\_ip\_address\_by\_mac(const char\* target\_mac) {

int found = 0;

for (int i = 0; i < MAX\_DEVICES; i++) {

if (strcmp(mac\_addresses[i], target\_mac) == 0) {

printf("RARP Reply: The IP address for MAC %s is %s\n", target\_mac, ip\_addresses[i]);

found = 1;

break;

}

}

if (!found) {

printf("RARP Reply: MAC address %s not found on the network.\n", target\_mac);

}

}

// ARP Request Simulation

void arp\_request(const char\* source\_ip, const char\* target\_ip) {

printf("ARP Request: Who has IP %s? (From IP %s)\n", target\_ip, source\_ip);

find\_mac\_address\_by\_ip(target\_ip);

}

// RARP Request Simulation

void rarp\_request(const char\* source\_mac) {

printf("RARP Request: What is the IP address for MAC %s?\n", source\_mac);

find\_ip\_address\_by\_mac(source\_mac);

}

int main() {

// Simulate ARP Request and Response

printf("\n== ARP Simulation ==\n");

arp\_request("192.168.1.1", "192.168.1.3"); // Request for MAC address of 192.168.1.3

arp\_request("192.168.1.2", "192.168.1.6"); // Request for non-existing IP

// Simulate RARP Request and Response

printf("\n== RARP Simulation ==\n");

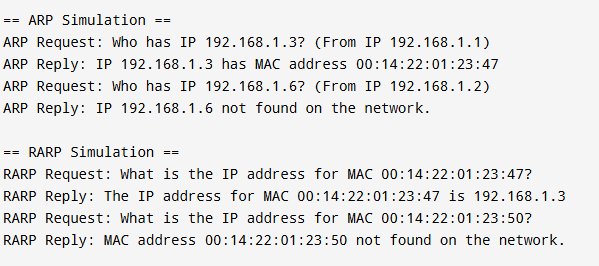
rarp\_request("00:14:22:01:23:47"); // Request for IP address of device with this MAC

rarp\_request("00:14:22:01:23:50"); // Request for non-existing MAC

return 0;

}

**OUTPUT**



**VIVA QUESTIONS**

1) What is a Link?

A link refers to the connectivity between two devices. It includes the type of cables and protocols used in order for one device to be able to communicate with the other.

2) What are the layers of the OSI reference model?

There are 7 OSI layers: Physical Layer, Data Link Layer, Network Layer, Transport Layer, Session Layer, Presentation Layer and Application Layer.

3) What is a LAN?

LAN is short for Local Area Network. It refers to the connection between computers and other network devices that are located within a small physical location.

4) What is a node?

A node refers to a point or joint where a connection takes place. It can be computer or device that is part of a network. Two or more nodes are needed in order to form a network connection.

5) What are routers?

Routers can connect two or more network segments. These are intelligent network devices that store information in its routing table such as paths, hops and bottlenecks. With this info, they are able to determine the best path for data transfer. Routers operate at the OSI Network Layer.

6) What is point to point link?

It refers to a direct connection between two computers on a network. A point to point connection does not need any other network devices other than connecting a cable to the NIC cards of both computers.

7) What is subnet mask?

A subnet mask is combined with an IP address in order to identify two parts: the extended network address and the host address. Like an IP address, a subnet mask is made up of 32 bits.

8) What is the maximum length allowed for a UTP cable?

A single segment of UTP cable has an allowable length of 90 to 100 meters. This limitation can be overcome by using repeaters and switches.

9) Describe Network Topology

Network Topology refers to the layout of a computer network. It shows how devices and cables are physically laid out, as well as how they connect to one another.

10) What is VPN?

VPN means Virtual Private Network, a technology that allows a secure tunnel to be created across a network such as the Internet. For example, VPNs allow you to establish a secure dial-up connection to a remote server.

11) What is the job of the Network Layer under the OSI reference model?

The Network layer is responsible for data routing, packet switching and control of network congestion. Routers operate under this layer.

12) What is NIC?

NIC is short for Network Interface Card. This is a peripheral card that is attached to a PC in order to connect to a network. Every NIC has its own MAC address that identifies the PC on the network.

13) What is WAN?

WAN stands for Wide Area Network. It is an interconnection of computers and devices that are geographically dispersed. It connects networks that are located in different regions and countries.

14) What is the importance of the OSI Physical Layer?

The physical layer does the conversion from data bits to electrical signal, and vice versa. This is where network devices and cable types are considered and setup.

15) How many layers are there under TCP/IP?

There are four layers: the Network Layer, Internet Layer, Transport Layer and Application Layer.

16) What is the function of the OSI Session Layer?

This layer provides the protocols and means for two devices on the network to communicate with each other by holding a session.  This includes setting up the session, managing information exchange during the session, and tear-down process upon termination of the session.

17) What does 10Base-T mean?

The 10 refers to the data transfer rate, in this case is 10Mbps. The word Base refers to base band, as oppose to broad band. T means twisted pair, which is the cable used for that network.

18) What is OSI and what role does it play in computer networks?

OSI (Open Systems Interconnect) serves as a reference model for data communication. It is made up of 7 layers, with each layer defining a particular aspect on how network devices connect and communicate with one another. One layer may deal with the physical media used, while another layer dictates how data is actually transmitted across the network.

19) What is the purpose of cables being shielded and having twisted pairs?

The main purpose of this is to prevent crosstalk. Crosstalks are electromagnetic interferences or noise that can affect data being transmitted across cables.

20) What are MAC addresses?

MAC, or Media Access Control, uniquely identifies a device on the network. It is also known as physical address or Ethernet address. A MAC address is made up of 6-byte parts.

21) What is the equivalent layer or layers of the TCP/IP Application layer in terms of OSI reference model?

The TCP/IP Application layer actually has three counterparts on the OSI model: the Session layer, Presentation Layer and Application Layer.

22) How can you identify the IP class of a given IP address?

By looking at the first octet of any given IP address, you can identify whether it’s Class A, B or C. If the first octet begins with a 0 bit, that address is Class A. If it begins with bits 10 then that address is a Class B address. If it begins with 110, then it’s a Class C network.

23) What is the main purpose of OSPF?

OSPF, or Open Shortest Path First, is a link-state routing protocol that uses routing tables to determine the best possible path for data exchange.

24) Describe star topology

Star topology consists of a central hub that connects to nodes. This is one of the easiest to setup and maintain.

25) What are gateways?

Gateways provide connectivity between two or more network segments. It is usually a computer that runs the gateway software and provides translation services. This translation is a key in allowing different systems to communicate on the network.

26) What is the disadvantage of a star topology?

One major disadvantage of star topology is that once the central hub or switch get damaged, the entire network becomes unusable.

27) Describe at one disadvantage of a peer to peer network.

When you are accessing the resources that are shared by one of the workstations on the network, that workstation takes a performance hit.

28) What is Hybrid Network?

A hybrid network is a network setup that makes use of both client-server and peer-to-peer architecture.

29) What is the main job of the ARP?

The main task of ARP or Address Resolution Protocol is to map a known IP address to a MAC layer address.

30) What is TCP/IP?

TCP/IP is short for Transmission Control Protocol / Internet Protocol. This is a set of protocol layers that is designed to make data exchange possible on different types of computer networks, also known as heterogeneous network.

31) How can you manage a network using a router?

Routers have built in console that lets you configure different settings, like security and data logging. You can assign restrictions to computers, such as what resources it is allowed access, or what particular time of the day they can browse the internet. You can even put restrictions on what websites are not viewable across the entire network.

32) What is the use of a default gateway?

Default gateways provide means for the local networks to connect to the external network. The default gateway for connecting to the external network is usually the address of the external router port.

33) What happens when you use cables longer than the prescribed length?

Cables that are too long would result in signal loss. This means that data transmission and reception would be affected, because the signal degrades over length.

34) What is ICMP?

ICMP is Internet Control Message Protocol. It provides messaging and communication for protocols within the TCP/IP stack. This is also the protocol that manages error messages that are used by network tools such as PING.

35) What is Ping?

Ping is a utility program that allows you to check connectivity between network devices on the network. You can ping a device by using its IP address or device name, such as a computer name.

36) What is peer to peer?

Peer to peer are networks that does not reply on a server. All PCs on this network act as individual workstations.

37) What is DNS?

DNS is Domain Name System. The main function of this network service is to provide host names to TCP/IP address resolution.

38) What advantages does fiber optics have over other media?

One major advantage of fiber optics is that is it less susceptible to electrical interference. It also supports higher bandwidth, meaning more data can be transmitted and received. Signal degrading is also very minimal over long distances.

39) What is the difference between a hub and a switch?

A hub acts as a multiport repeater. However, as more and more devices connect to it, it would not be able to efficiently manage the volume of traffic that passes through it. A switch provides a better alternative that can improve the performance especially when high traffic volume is expected across all ports.

40) What is ipconfig?

Ipconfig is a utility program that is commonly used to identify the addresses information of a computer on a network. It can show the physical address as well as the IP address.

41) What is client/server?

Client/server is a type of network wherein one or more computers act as servers. Servers provide a centralized repository of resources such as printers and files. Clients refers to workstation that access the server.

42) Describe networking.

Networking refers to the inter connection between computers and peripherals for data communication. Networking can be done using wired cabling or through wireless link.

43) When you move the NIC cards from one PC to another PC, does the MAC address gets transferred as well?

Yes, that’s because MAC addresses are hard-wired into the NIC circuitry, not the PC. This also means that a PC can have a different MAC address when the NIC card was replace by another one.

44) Describe Ethernet.

Ethernet is one of the popular networking technologies used these days. It was developed during the early 1970s and is based on specifications as stated in the IEEE. Ethernet is used in local area networks.

45) What are some drawbacks of implementing a ring topology?

In case one workstation on the network suffers a malfunction, it can bring down the entire network. Another drawback is that when there are adjustments and reconfigurations needed to be performed on a particular part of the network, the entire network has to be temporarily brought down as well.

46) What is the difference between CSMA/CD and CSMA/CA?

CSMA/CD, or Collision Detect, retransmits data frames whenever a collision occurred. CSMA/CA, or Collision Avoidance, will first broadcast intent to send prior to data transmission.

47) What is SMTP?

SMTP is short for Simple Mail Transfer Protocol. This protocol deals with all Internal mail, and provides the necessary mail delivery services on the TCP/IP protocol stack.

48) What is multicast routing?

Multicast routing is a targeted form of broadcasting that sends message to a selected group of user, instead of sending it to all users on a subnet.

49) What do mean by tunnel mode?

This is a mode of data exchange wherein two communicating computers do not use IPSec themselves. Instead, the gateway that is connecting their LANs to the transit network creates a virtual tunnel that uses the IPSec protocol to secure all communication that passes through it.

50) What is one advantage of mesh topology?

In the event that one link fails, there will always be another available. Mesh topology is actually one of the most fault-tolerant network topology.

51) What is IPv6?

IPv6 , or Internet Protocol version 6, was developed to replace IPv4. At present, IPv4 is being used to control internet traffic, butis expected to get saturated in the near future. IPv6 was designed to overcome this limitation.

52) What is mesh topology?

Mesh topology is a setup wherein each device is connected directly to every other device on the network. Consequently, it requires that each device have at least two network connections.

53) what is the maximum segment length of a 100Base-FX network?

The maximum allowable length for a network segment using 100Base-FX is 412 meters. The maximum length for the entire network is 5 kilometers.

54) Explain Stop-and-Wait Automatic Repeat Request for noisy channel.

In this protocol the sending device keeps a copy of the last frame transmitted until it receives an acknowledgment for that frame. A data frames uses a sequence number; an ACK frame uses an acknowledgment number. The sender has a control variable, which we call Sn (sender, next frame to send), that holds the sequence number for the next frame to be sent (0 or 1).

55) what is the difference between Pure ALOHA and Slotted Aloha protocol?

In Pure Aloha, station can send data in a continuous time manner whereas in Slotted Aloha, time is divided in slots. Pure ALOHA doesn't check whether the channel is busy before transmission. In slotted ALOHA, a frame can be sent only at fixed times, whereas in pure ALOHA, you can send any time. Pure ALOHA has a vulnerable time of 2 x Tfr. Whereas in Slotted ALOHA vulnerable time = Tfr {Tfr: Average transmission time for a frame}.

56) What is IP?

It's a unique 32 bits software address of a node in a network.

57) What is Multiple Access?

If the physical links are shared by more than two nodes, it is said to be Multiple Access.

58) What is Protocol?

A protocol is a set of rules that govern all aspects of information communication.

59) Define Bandwidth and Latency?

Network performance is measured in Bandwidth (throughput) and Latency (Delay). Bandwidth of a network is given by the number of bits that can be transmitted over the network in a certain period of time. Latency corresponds to how long it t5akes a message to travel from one end off a network to the other. It is strictly measured in terms of time.

60) Define the terms Unicasting, Multicasting and Broadcasting?

If the message is sent from a source to a single destination node, it is called Unicasting.  
If the message is sent to some subset of other nodes, it is called Multicasting.  
If the message is sent to all the m nodes in the network it is called Broadcasting.

61) What is Multiplexing?

Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

62) CWND

Congestion Window (cwnd)

63) TCP Reno

is the extension of TCP Tahoe, and NewReno is the extension of TCP Reno. In Reno, when packet loss occurs, the sender reduces the cwnd by 50% along with the ssthresh value.

64) Telnet

Developed in 1969, is a protocol that provides a command line interface for communication with a remote device or server, sometimes employed for remote management but also for initial device setup like network hardware.