DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION & MISSION OF THE INSTITUTE

Vision of the Institute:

To be a globally recognized institution that fosters innovation, excellence, and leadership in education, research, and technology development, empowering students to create sustainable solutions for the advancement of society.

Mission of the Institute:

To foster a transformative learning environment that empowers students to excel in engineering, innovation, and leadership.

To produce skilled, ethical, and socially responsible engineers who contribute to sustainable technological advancements and address global challenges.

To shape future leaders through cutting-edge research, industry collaboration, and community engageme

PROGRAMME EDUCATIONAL OBJECTIVES

The Programme Educational Objectives (PEOs) that are formulated for the Information Technology programme are listed below:

PEO1 : To induce strong foundation in mathematical and core concepts, which enable them to participate in research, in the field of computer science.

PEO2 : To be able to become the part of application development and problem solving by learning the computer programming methods, of the industry and related domains.

PEO3 : To gain the multidisciplinary knowledge by understanding the scope of association of computer science engineering discipline with other engineering disciplines.

PEO4 : To improve the communication skills, soft skills, organizing skills which build the professional qualities, there by understanding the social responsibilities and ethical attitude.

PROGRAMME OUT COMES

The Program Outcomes (POs) of the department are defined in a way that the Graduate Attributes are included, which can be seen in the Program Outcomes (POs) defined. The Program Outcomes (POs) of the department are as stated below:

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2:Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3:Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4:Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5:Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6:The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

PO7:Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8:Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9:Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10:Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11:Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12:Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAMME SPECIFIC OUT COMES

PSO1: Applications of Computing: Ability to use knowledge in various domains to provide solution to new ideas and innovations.

PSO2: Programming Skills: Identify required data structures, design suitable algorithms, develop and maintain software for real world problems.

PSO3: Make use of computational and experimental tools for creating innovative career paths, to be an entrepreneur and desire for higher studies

Course Outcomes:

After successful completion of the course, students should be able to:

CO1	Implement Data link layer framing methods
CO2	Analyze Error Detection And Error Correction Codes.
CO3	Compare various routing and congestion algorithms
CO4	Create Various Encoding and Decoding techniques
CO5	Utilize different network Simulation tools

MAPPING OF EACH CO WITH PO(s), PSO(s)

Course Outcomes (COs)	Program Outcomes (POs)												Program Specific Outcomes (PSOs)		
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
CO 1	Y		Y	Y								Y	Y		
CO 2	Y		Y	Y	Y							Y	Y		
CO 3	Y		Y	Y	Y							Y	Y		
CO 4	Y		Y	Y	Y				Y			Y			Y
CO 5	Y	Y	Y	Y	Y					Y		Y			Y

Simple-1 Moderate-2 High-3

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VISION & MISSION

.

Vision of the Department:

To empower the students to be technologically adept, innovative, self-motivated and responsible global citizen possessing human values and contribute significantly towards high quality technical education with ever changing world.

Mission of the Department:

- To offer high-quality education in the computing fields by providing an environment where the knowledge is gained and applied to participate in research, for both students and faculty.
- To develop the problem-solving skills in the students to be ready to deal with cutting edge technologies of the industry.
- To make the students and faculty excel in their professional fields by inculcating the communication skills, leadership skills, team building skills with the organization of various co-curricular and extra curricular programmes.
- To provide the students with theoretical and applied knowledge, and adopt an education approach that promotes lifelong learning and ethical growth

Program Educational Objectives (PEOs)

The Programme Educational Objectives (PEOs) that are formulated for the Information Technology programme are listed below:

PEO1	To induce strong foundation in mathematical and core concepts, which enable them to participate in research, in the field of computer science.
PEO2	To be able to become the part of application development and problem solving by learning the computer programming methods, of the industry and related domains.
PEO3	To gain the multidisciplinary knowledge by understanding the scope of association of computer science engineering discipline with other engineering disciplines.
PEO4	To improve the communication skills, soft skills, organizing skills which build the professional qualities, there by understanding the social responsibilities and ethical attitude.

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CN LABORATORY

Virtual lab details

Name of the Virtual Lab:

Virtual Lab Host Institute:

URL/Link to Lab

Academic Year

Semester

List of Experiments Available in Virtual Lab

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CN LABORATORY

LAB PLANNER

S.No	Experiment	CO	Virtual Lab Availabilty	Date planned	Date conducted
1	Implement the data link layer framing methods such as character, character-stuffing and bit stuffing.	CO1			
2	Write a program to compute CRC code for the polynomials CRC-12 and CRC-16 and CRC CCIP	CO2			
3	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.	CO3			
4	Implement Dijsktra‟s algorithm to compute the shortest path through a network	CO3			
5	Take an example subnet of hosts and obtain a broadcast tree for the subnet.	CO3			
6	Implement distance vector routing algorithm for obtaining routing tables at each node.				
7	Implement data encryption and data decryption	CO1			
8	Write a program for congestion control using Leaky bucket algorithm.	CO4			
9	Write a program for frame sorting technique used in buffers.	CO4			
10	Implement the following using Wireshark i. Packet Capture Using Wireshark ii. Starting Wireshark iii. Viewing Captured Traffic iv. Analysis and Statistics &Filters.	CO5			
11	How to run Nmapscan Operating System Detection using Nmap	CO5			
12	Do the following using NS2Simulator Simulate to Find the Number of Packets Dropped	CO5			
13	Simulate to Find the Number of Packets Dropped by TCP/UDP	CO5			
14	Simulate to Find the Number of Packets Dropped due to Congestion	CO5			
15	Simulate to Compare Data Rate & Throughput	CO5			
16	Simulate to Plot Congestion for Different Source/Destination	CO5			
17	Simulate to Determine the Performance with respect to Transmission of Packets				

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CN LABORATORY

RUBRICS USED TO ASSESS LEARNINGS IN LABORATORIES

1. RUBRICS FOR DAY TO DAY EVALUATION

Parameter	Max Marks	Level-1 (Very Poor)	Level-2 (Poor)	Level-3 (Average)	Level-4 (Good)	Level-5 (Excellent)
Observation Book	05	No observations or irrelevant data. (0–1)	Incomplete or incorrect data. (2)	Basic values with some errors. (3)	Mostly correct with good format. (4)	Fully correct, clear, and well-formatted. (5)
Record Writing	05	Not submitted. (0-1)	Submitted but mostly incomplete. (2)	Submitted with some missing/wrong parts. (3)	Submitted with minor issues. (4)	Fully complete, correct algorithm & flowchart. (5)
Result	05	No result or major errors. (0–1)	Result partially obtained. (2)	Acceptable result with limited error. (3)	Near-correct result and reasonable error. (4)	Accurate result. (5)
Viva-Voce	05	Did not answer any questions. (1)	Answered very few questions. (2)	Answered some questions with help. (3)	Answered most questions correctly. (4)	Answered all questions accurately. (5)

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CN LABORATORY

2.RUBRICS FOR INTERNAL EVALUATION

Criterion	Max Marks	Level-1 *(Very Poor)*	Level-2 *(Poor)*	Level-3 *(Average)*	Level-4 *(Good)*	Level-5 *(Excellent)*
Design/Tool/Apparatus Selection	2 Marks	Incorrect tool/design and no reasoning. **(0)**	Tool/design selection attempted with unclear logic. **(0.5)**	Satisfactory selection with partial justification. **(1)**	Correct selection and proper analysis with few errors. **(1.5)**	Smart selection with accurate, relevant analysis. **(2)**
Execution (Code/Debug/Run) /Analysis/Method Used	4 Marks	Did not attempt or completely failed to execute. **(0)**	Attempted but unable to proceed or with major errors. **(1)**	Partial execution with some logic/syntax errors. **(2)**	Mostly correct execution with minimal help. **(3)**	Fully correct and independently executed program. **(4)**
Results& Documentation	2 Marks	Incomplete or poorly presented. **(0)**	Basic structure but lacks clarity or formatting. **(0.5)**	Complete but generic or with formatting issues. **(1)**	Well-structured and mostly clear. **(1.5)**	Well-organized, professional, and engaging documentation. **(2)**
Viva-Voce (Understanding of Concepts)	2 Marks	No understanding; could not answer questions. **(0)**	Answered a few with difficulty. **(0.5)**	Answered half the questions with basic clarity. **(1)**	Good understanding with confident answers. **(1.5)**	Answered all questions with clarity and depth. **(2)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CN LABORATORY

3. RUBRICS FOR SEMESTER END EXAMINATIONS

Criterion	Max Marks	Level-1 (Very Poor (*0–2 marks)*	Level-2 (Poor) *(3–4 marks)*	Level-3 (Average) *(5–6 marks)*	Level-4 (Good) *(7–9 marks)*	Level-5 (Excellent) *(10–12 marks)*
Preparedness for the Experiment	12 marks	No clarity on objective or procedure. Unable to explain basics.	Limited idea of the objective/procedure. Needed prompting.	Has basic understanding; minor gaps in concept or preparation.	Well-prepared, with clear understanding of steps and background.	Fully prepared with strong conceptual clarity and confident explanation.
Performance in the Laboratory	12 marks	Unable to perform experiment. Relied entirely on examiner's help.	Performed with multiple errors and constant support.	Performed with some errors; required occasional help.	Performed mostly independently with minimal support.	Performed independently, efficiently, and with precision.
Calculations & Graphs	12 marks	No or incorrect calculations. Graphs missing or irrelevant.	Multiple calculation errors. Graphs/plots inaccurate or poorly labeled.	Calculations partially correct. Graphs present but with some flaws.	Correct calculations and graphs with minor errors.	Accurate calculations and well-labeled graphs with proper interpretation
Results & Error Analysis	12 marks	No result or invalid result. No error analysis attempted.	Incorrect result with vague or no error discussion.	Acceptable result. Error analysis attempted but limited.	Correct result with sound error discussion.	Accurate result with detailed and relevant error analysis.
Viva-Voce (Subject Knowledge)	12 marks	Unable to answer any questions. No conceptual understanding.	Answered few questions with poor logic.	Answered half of the questions with average understanding.	Answered most questions with clarity and confidence.	Answered all questions with depth, clarity, and reasoning.

B.Tech Syllabus (MLRS-R24) MLRITM

2050578: COMPUTER NETWORKS LAB

III Year B.Tech. CSE I – Sem. **L T P C 0 0 3 1.5**

Course Objectives

- To understand the working principle of various communication protocols. · To understand the network simulator environment and visualize a network topology and observe its performance.
- To analyze the traffic flow and the contents of protocol frames.

Course Outcomes:
The students will be able to:

- Implement data link layer farming methods
- Analyze error detection and error correction codes.
- Implement and analyze routing and congestion issues in network design. · Implement Encoding and Decoding techniques used in presentation layer · Work with different network tools

LIST OF EXPERIMENTS

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

2. Write a program to compute CRC code for the polynomials CRC-12 andCRC-16

3. 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.

4. Implement Dijsktra's algorithm to compute the shortest path through a network 5.

Take an example subnet of hosts and obtain a broadcast tree for the subnet. 6. Implement

distance vector routing algorithm for obtaining routing tables at each node.

7. Implement data encryption and data decryption
8. Write a program for congestion control using Leaky bucket algorithm
9. Write a program for frame sorting technique used in buffers.

10. Wire shark
• Packet Capture Using Wire shark
• Starting Wire shark
• Viewing Captured Traffic
• Analysis and Statistics & Filters.

11. How to run Nmap scan
12. Operating System Detection using Nmap

13. Do the following using NS2 Simulator
    - NS2Simulator-Introduction
    - Simulate to Find the Number of Packets Dropped
    - Simulate to Find the Number of Packets Dropped by TCP/UDP
    - Simulate to Find the Number of Packets Dropped due to Congestion • Simulate to Compare Data Rate & Throughput.
    - Simulate to Plot Congestion for Different Source/Destination

• Simulate to Determine the Performance with respect to Transmission of Packets.

TEXTBOOKS:

1. Computer Networks, Andrew S Tanenbaum, David. j. Wetherall, 5th Edition. Pearson Education/PHI.

REFERENCES:

1. An Engineering Approach to Computer Networks, S. Keshav, 2ndEdition, Pearson Education

2. Data Communications and Networking– Behrouz A. Forouzan.3rd Edition, TMH.

GENERAL INSTRUCTIONS

1. Students are instructed to come to Computer Networks laboratory on time. Late comers are not entertained in the lab.
2. Students should be punctual to the lab. If not, the conducted experiments will not be repeated.
3. Students are expected to come prepared at home with the experiments which are going to be performed.
4. Students are instructed to display their identity cards before entering into the lab.
5. Students are instructed not to bring mobile phones to the lab.
6. Any damage/loss of system parts like keyboard, mouse during the lab session, it is student's responsibility and penalty or fine will be collected from the student.
7. Students should update the records and lab observation books session wise. Before leaving the lab the student should get his lab observation book signed by the faculty.
8. Students should submit the lab records by the next lab to the concerned faculty members in the staffroom for their correction and return.
9. Students should not move around the lab during the lab session.
10. If any emergency arises, the student should take the permission from faculty member concerned in written format.
11. The faculty members may suspend any student from the lab session on disciplinary grounds.
12. Never copy the output from other students. Write down your own outputs.

EXPERIMENT -1

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

1 a. NAME OF THE EXPERIMENT: Character Stuffing.

OBJECTIVE: Implement the data link layer framing methods.
RESOURCE: Turbo C

PROGRAM LOGIC:
The framing method gets around the problem of resynchronization after an error by having each frame start with the ASCII character sequence DLE STX and the sequence DLE ETX. If the destination ever losses the track of the frame boundaries all it has to do is look for DLE STX or DLE ETX characters to figure out. The data link layer on the receiving end removes the DLE before the data are given to the network layer. This technique is called character stuffing.

ALOGRITM:

Begin
Step 1: Initialize I and j as 0
Step 2: Declare n and pos as integer and a[20],b[50],ch as character Step
3: read the string a
Step 4: find the length of the string n, i.e n-strlen(a)
Step 5: read the position, pos
Step 6: if pos > n then
Step 7: print invalid position and read again the position, pos
Step 8: endif
Step 9: read the character, ch
Step 10: Initialize the array b, b[0…5] as 'd', 'l', 'e', 's', 't','x' respectively
Step 11: j=6;
Step 12: Repeat step[(13to22) until i<n
Step 13: if i==pos-1 then
Step 14: initialize b array,b[j],b[j+1]…b[j+6] as'd', 'l', 'e' ,'ch, 'd', 'l','e' respectively Step
15: increment j by 7, i.e j=j+7
Step 16: endif
Step 17: if a[i]=='d' and a[i+1]=='l' and a[i+2]=='e' then Step
18: initialize array b, b[13…15]='d', 'l', 'e' respectively Step 19:
increment j by 3, i.e j=j+3
Step 20: endif
Step 21: b[j]=a[i]
Step 22: increment I and j;
Step 23: initialize b array,b[j],b[j+1]…b[j+6] as'd', 'l','e','e','t', 'x','\0' respectively Step
24: print frame after stuffing
Step 25: print b End

PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

SOURCE CODE:

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void charc();
void main()
{
  int choice;
  while(1)
  {
      printf("\n\n\n 1.character stuffing");
      printf("\n\n 2.exit"); printf("\\n\n\nenter
      choice:"); scanf("%d",&choice);
      if(choice>2)
      {
        printf("\n\n invalid option...please re-enter");
       }
        switch(choice)
        {
            case 1:charc();
             break;
            case 2:exit(0);
        }}
   }
 void charc(void)
   {
      char c[50],d[50],t[50]; int
      i,m,j;
      printf("enter the number of characters:\n");
      scanf("%d",&m);
      printf("enter the characters:\n");
      for(i=0;i<m+1;i++)
      {
         scanf("%c",&c[i]);

 }
      printf("\n original data \n");
      for(i=0;i<m+1;i++)
printf("%c",c[i]);

d[0]='d';
      d[1]='l';
      d[2]='e';
      d[3]='s';
      d[4]='t';
      d[5]='x';
      for(i=0,j=6;i<m+1;i++,j++)
```

```
    {
      if((c[i]=='d'&&c[i+1]=='l'&&c[i+2]=='e'))
      {
          d[j]='d';
          j++;
          d[j]='l';
          j++;
          d[j]='e';
          j++;
          m=m+3;
      }
      d[j]=c[i];
    }
    m=m+6;
    m++;
    d[m]='d';
    m++;
    d[m]='l';
    m++;
    d[m]='e'; m++;
    d[m]='e'; m++;
    d[m]='t';
    m++;
    d[m]='x';
    m++;
    printf("\n\n transmitted data:\n");
for(i=0;i<m;i++)
    {
        printf("%c",d[i]);
   }
    for(i=6,j=0;i<m-6;i++,j++)
    {

  if(d[i]=='d'&&d[i+1]=='l'&&d[i+2]=='e'&&d[i+3]=='d'&&d[i+4]=='l'&&d[i+5]=='e')
      i=i+3;
      t[j]=d[i];
    }
    printf("\n\n received data:");
    for(i=0;i<j;i++)
{
        printf("%c",t[i]);
    }
  }
```

OUTPUT:

```
 1.character stuffing

 2.exit\n

enter choice:1
enter the number of characters:
5
enter the characters:
HELLO

 original data

HELLO

 transmitted data:
dlestx
HELLOdleetx

 received data:
HELLO


 1.character stuffing

 2.exit\n

enter choice:
```

1. **B NAME OF THE EXPERIMENT**: **Bit Stuffing**.

OBJECTIVE: Implement the data link layer framing method.

RESOURCE: Turbo C

PROGRAM LOGIC:

The new technique allows data frames to contain an arbitrary number if bits and allows character codes with an arbitrary no of bits per character. Each frame begins and ends with special bit pattern, 01111110, called a flag byte. Whenever the sender's data link layer encounters five consecutive ones in the data, it automatically stuffs a 0 bit into the outgoing bit stream. This bit stuffing is analogous to character stuffing, in which a DLE is stuffed into the outgoing character stream before DLE in the data.

ALOGRITM:

Begin
Step 1: Read frame length n
Step 2: Repeat step (3 to 4) until i<n(: Read values into the input frame (0's and1's) i.e.
Step 3: initialize I i=0;
Step 4: read a[i] and increment i Step
5: Initialize i=0, j=0,count =0
Step 6: repeat step (7 to 22) until i<n Step
7: If a[i] == 1 then
Step 8: b[j] = a[i]
Step 9: Repeat step (10 to 18) until (a[k] =1 and k<n and count <5) Step
10: Initialize k=i+1;
Step 11: Increment j and b[j]= a[k]; Step
12: Increment count ;
Step 13: if count =5 then
Step 14: increment j, Step
15: b[j] =0
Step 16: end if Step
17: i=k;
Step 18: increment k Step
19: else
Step 20: b[j] = a[i] Step
21: end if
Step 22: increment I and j
Step 23: print the frame after bit stuffing Step
24: repeat step (25 to 26) until i< j Step 25:
print b[i]
Step 26: increment i End

SOURCE CODE:

```
#include<stdio.h>
#include<string.h> void
main()
{
  int a[20],b[20],i,j,k,n,count; printf("enter
  frame length:"); scanf("%d",&n);
  printf("enter input frame(0's & 1's only);");
  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 stuffing the frame is:");
for(i=0;i<j;i++)
printf("%d",b[i]);

}
```

OUTPUT :

```
enter frame length:7
enter input frame(0's & 1's only):1
1
1
1
1
1
0
after stuffing the frame is:11111010
```

1. c. NAME OF THE EXPERIMENT: Character count

SOURCE CODE:

```
#include<stdio.h>
#include<string.h>
char data[20][20]; int
n;
void main()
{
int i,ch,j;
char tmp[20][20];
printf("enter the number of frames:");
scanf("%d",&n);
for(i=0;i<=n;i++)
{
if(i!=0)
{
printf("frame%d:",i);
scanf("%s",&data[i]);
}
}
for(i=0;i<=n;i++)
{
tmp[i][0]=49+strlen(data[i]);
tmp[i][1]='\0';
strcat(tmp[i],data[i]);
}
printf("\n\t\t AT THE SENDER:\n");
printf("data as frames:\n"); for(i=1;i<=n;i++)
{
printf("frame%d:",i);
puts(tmp[i]);
}
printf("data transmitted:");
for(i=1;i<=n;i++)
printf("%s",tmp[i]);
printf("\n\t\t AT THE RECEIVER\n");
printf("the data received:");
```

```
for(i=1;i<=n;i++)
{
ch=(int)(tmp[i][0]-49);
for(j=1;j<=ch;j++)
data[i][j-1]=tmp[i][j];
data[i][j-1]='\0';
}
printf("\n the data after removing count char:");
for(i=1;i<=n;i++)
printf("%s",data[i]);
printf("\n the data in frame form:\n");
for(i=1;i<=n;i++)
{
printf("frame%d:",i);
puts(data[i]);
}
}
```

OUTPUT:

```
enter the number of frames:5
frame1:computer
frame2:networks
frame3:lab
frame4:mlritm
frame5:college

                AT THE SENDER:
data as frames:
frame1:9computer
frame2:9networks
frame3:4lab
frame4:7mlritm
frame5:8college
data transmitted:9computer9networks4lab7mlritm8college
                AT THE RECEIVER
the data received:
 the data after removing count char:computernetworkslabmlritmcollege
 the data in frame form:
frame1:computer
frame2:networks
frame3:lab
frame4:mlritm
frame5:college


...Program finished with exit code 0
Press ENTER to exit console.
```

Additional Programmes

Program – 1:- Implement three nodes point–to–point networks with duplex links between them. Set the queue size, vary the bandwidth, and find the number of packets dropped.

Program 2:- Implement transmission of ping messages/traceroute over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Define a network.	CO1	Understand
2	How do links work?	CO1	Remember
3	Describe a Node.	CO1	Remember
4	What is a router or gateway?	CO1	Remember
5	Describe the point-to-point link.	CO1	Remember
6	What is multiple access, exactly?	CO1	Remember
7	What benefits do distributed processing systems offer?	CO1	Remember
8	What are the names of the variables that impact the performance of the network?	CO1	Remember
9	What standards must a network meet to be effective and efficient?	CO1	Remember
10	Talk about the elements that affect the network's dependability.	CO1	Remember
11	What are the factors that affect the security of the network?	CO1	Remember
12	What is the protocol?	CO1	Remember
13	What is the essential component of protocol?	CO1	Remember
14	Discuss the key design issues of a computer network.	CO1	Understand
15	What are latency and bandwidth?	CO1	Understand
16	Talk about routing.	CO1	Understand
17	What is a peer-to-peer process?	CO1	Understand
18	What is the congested switch?	CO1	Understand
19	How does WDM work?	CO1	Understand
20	What is TDM?	CO1	Remember

EXPERIMENT - 2

1. **Write a program to compute CRC code for the polynomials CRC-12 and CRC-16 and CRC CCIP**

NAME OF THE EXPERIMENT: Cyclic Redundancy Check.

OBJECTIVE: Implement on a data set of characters the three CRC polynomials – CRC 12,CRC 16
RESOURCE: Turbo C

PROGRAM LOGIC:
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 known as Cyclic Redundancy Code

PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

ALGORITHM:

Step 1:A string of n as is appended to the data unit. The length of predetermined divisor is n+ 1. Step 2:The newly formed data unit 1. A string of n as is appended to the data unit. The length of predetermined divisor is n+ 1. i.e. original data + string of n as are divided by the divisor using binary division and remainder is obtained. This remainder is called CRC.
Step 3:Now, string of n Os appended to data unit is replaced by the CRC remainder (which is also of n bit).
Step 4: The data unit + CRC is then transmitted to receiver.
Step 5: The receiver on receiving it divides data unit + CRC by the same divisor & checks the remainder.
Step 6: If the remainder of division is zero, receiver assumes that there is no error in data and it accepts it.
Step 7: If remainder is non-zero then there is an error in data and receiver rejects it.

SOURCE CODE:

```
//PROGRAM FOR CYCLIC REDUNDENCY CHECK
#include<string.h>
#include<stdio.h>
char text[20],key[20],rem[20];
void crc()
{
int i,j,keylen,textlen; char
temp[100];
strcpy(temp,text);
keylen=strlen(key);
for(i=0;i<keylen-1;i++)
strcat(temp,"0");
textlen=strlen(temp);
strncpy(rem,temp,keylen);
while(i!=textlen)
{
if(rem[0]=='0')
```

```
{
strcpy(rem,&rem[1]);
rem[keylen-1]=temp[++i];
rem[keylen]='\0'; continue;
}
for(j=0;j<keylen;j++) rem[j]=((rem[j]-
'0')^(key[j]-'0'))+'0';
}}
main()
{
int i,choice;
while(1)
{
printf("\n1.Find CRC\n2.Check CRC\n3.Quit\nEnter the choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:
printf("Enter the i/p\n");
scanf("%s",text);
printf("Enter the key\n");

scanf("%s",key);
crc();
printf("Msg %s \n",strcat(text,rem));
break;
case 2:
printf("Enter the input");
scanf("%s",text);
printf("\nEnter key");
scanf("%s",key);
crc();
for(i=0;i<strlen(key)-1;i++)
if(rem[i]=='1')
break;
if(i==strlen(key)-1)
printf("NO ERROR\n");
else
printf("ERROR"); break;
case 3:
exit(0);
}}}
```

OUTPUT:

Additional Programmes

Program 1:- Implement transmission of ping messages/traceroute over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Which layer in the OSI stack connects the user-supported and network-supported layers?	CO1	Understand
2	Talk about the issues with the physical layer.	CO1	Understand
3	Discuss the responsibilities of the data link layer.	CO1	Understand
4	Explain the functions of the network layer.	CO1	Understand
5	Explain the functions of the transport layer.	CO1	Understand
6	Explain the function of the session layer.	CO1	Understand
7	Explain the responsibilities of the presentation layer role.	CO1	Understand
8	Describe the responsibility of the application layer.	CO1	Understand
9	What are the two categories of hardware components?	CO1	Understand
10	What different kinds of links can be employed to construct a computer network?	CO1	Understand
11	Discuss the various categories of transmission media.	CO1	Understand
12	What kinds of errors are there?	CO1	Understand
13	What is computer network error detection, and what are its methods?	CO1	Understand
14	Describe redundancy.	CO1	Understand
15	What is a vertical redundancy check?	CO1	Understand
16	How do you perform a longitudinal redundancy check?	CO1	Understand
17	What does cyclic redundancy check mean?	CO1	Understand
18	What is the checksum?	CO1	Understand
19	Describe the steps involved in creating the checksum.	CO1	Understand
20	Describe the data link protocols.	CO1	Understand
21	What is the difference between error correction and error detection?	CO1	Understand
22	Define forward error correction.	CO1	Understand
23	What is retransmission?	CO1	Understand
24	Define data words.	CO1	Understand

25	What are code words?	CO1	Understand
26	Define linear block code.	CO1	Understand
27	What is cyclic code?	CO1	Understand
28	What is an encoder?	CO1	Understand
29	What is a decoder?	CO1	Understand
30	What is framing?	CO1	Understand

EXPERIMENT -3

3 .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.

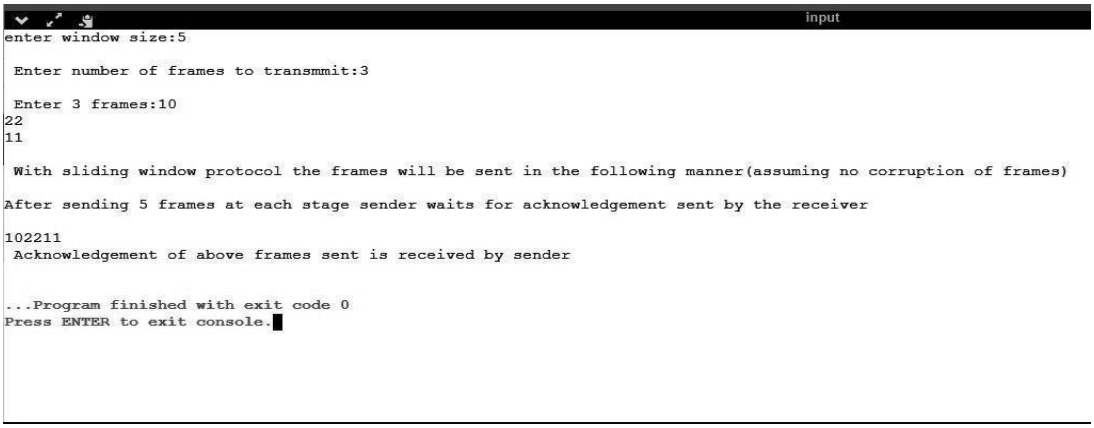
3.a. Sliding window protocol

Objective: implanting Sliding window protocol

SOURCE CODE :

```
#include<stdio.h> int
main()
{
int w,i,f,frames[50];
printf("enter window size:");
scanf("%d",&w);
printf("\n Enter number of frames to transmmit:");
scanf("%d",&f);
printf("\n Enter %d frames:",f);
for(i=1;i<=f;i++) scanf("%d",&frames[i]);
printf("\n With sliding window protocol the frames will be sent in the following manner(assuming no corruption of frames)\n\n");
printf("After sending %d frames at each stage sender waits for acknowledgement sent by the receiver\n\n",w);
for(i=1;i<=f;i++)
{
if(i%w==0)
{
printf("%d\n",frames[i]);
printf("Acknowledgement of above frames sent is received by sender\n\n");
}
else printf("%d",frames[i]);
}
if(f%w!=0)
printf("\n Acknowledgement of above frames sent is received by sender\n"); return 0;
}
```

OUTPUT:



3.b. - Sliding window protocol using go-bach N:

SOURCE CODE :

```
#include<stdio.h> int
main()
{
        int windowsize,sent=0,ack,i;
        printf("enter window size\n");
        scanf("%d",&windowsize); while(1)
        {
                for( i = 0; i<windowsize; i++)
                        {
                                printf("Frame %d has been transmitted.\n",sent);
                                sent++;
                                if(sent == windowsize)
                                        break;
                        }
                        printf("\nPlease enter the last Acknowledgement received.\n");
                        scanf("%d",&ack);

                        if(ack == windowsize)
                                break;
                        else
                                                                sent = ack;
        }
return 0;
}
```

OUTPUT:

```
input
enter window size
6
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.
Frame 5 has been transmitted.

Please enter the last Acknowledgement received.
5
Frame 5 has been transmitted.

Please enter the last Acknowledgement received.
0
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.
Frame 5 has been transmitted.

Please enter the last Acknowledgement received.
4
Frame 4 has been transmitted.
Frame 5 has been transmitted.

Please enter the last Acknowledgement received.
```

Additional Programmes

Program 4:- Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to the transmission of packets.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What is character stuffing?	CO1	Understand
2	What is bit stuffing?	CO1	Understand
3	How does flow control work?	CO1	Remember
4	Define error control.	CO1	Remember
5	Define automatic repeat request.	CO1	Remember
6	Define stop-and-wait protocol.	CO1	Remember
7	What is an automatic stop-and-wait repeat request?	CO1	Understand
8	What is the use of sequence number in reliable transmission?	CO1	Understand
9	Discuss pipelining.	CO1	Remember
10	What is a sliding window?	CO1	Remember
11	What are the types of transmission technology available?	CO1	Remember
12	Define subnet.	CO1	Remember
13	What are the differences between transmission and communication?	CO1	Remember
14	Describe the possible ways of data exchange.	CO1	Remember
15	Define SAP.	CO1	Remember
16	Define triple X in the computer network.	CO1	Remember
17	Define frame relay and in which layer it comes under.	CO1	Remember
18	Define terminal emulation and in which layer it comes under.	CO1	Remember
19	Define beaconing.	CO1	Remember
20	Define redirector.	CO1	Remember
21	Define the terms NETBEUI and NETBIOS.	CO1	Remember
22	What is RAID?	CO1	Understand
23	Define the term passive topology.	CO1	Remember
24	What is cladding?	CO1	Understand
25	Define point-to-point protocol.	CO1	Remember

26	How is the gateway different from routers?	CO1	Remember
27	What is a Mac address?	CO1	Understand
28	What is the difference between bit rate and baud rate?	CO1	Understand
29	Discuss piggybacking.	CO1	Remember
30	Discuss the types of transmission media.	CO1	Remember

EXPERIMENT - 4

4. **Implement Dijsktra's algorithm to compute the shortest path through a network**

Objective: Implementing Dijsktra's algorithm
RESOURCE: Turbo C

Program Logic: Dijkstra's algorithm is very similar to Prim's algorithm for minimum spanning tree. Like Prim's MST, we generate a SPT (shortest path tree) with given source as root. We maintain two sets, one set contains vertices included in shortest path tree, and other set includes vertices not yet included in shortest path tree.

PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

ALGORITM:

1. In distance vector routing algorithm, the **cost** is considered as the **hop** count (number of networks passed to reach the destination node). So a cost between two neighbouring routers is set to 1.
2. Each router updates its routing table when it receives the information (distance vector) form the neighbouring routers.
3. After updating its routing table, a router must forward its result to its neighbouring router So that they can update their routing table.
4. Each router keeps the three information in its routing table i.e. **destination network**, **cost** & the **next hop**.
5. The router sends the information of each route as a record **R**.

SOURCE CODE:

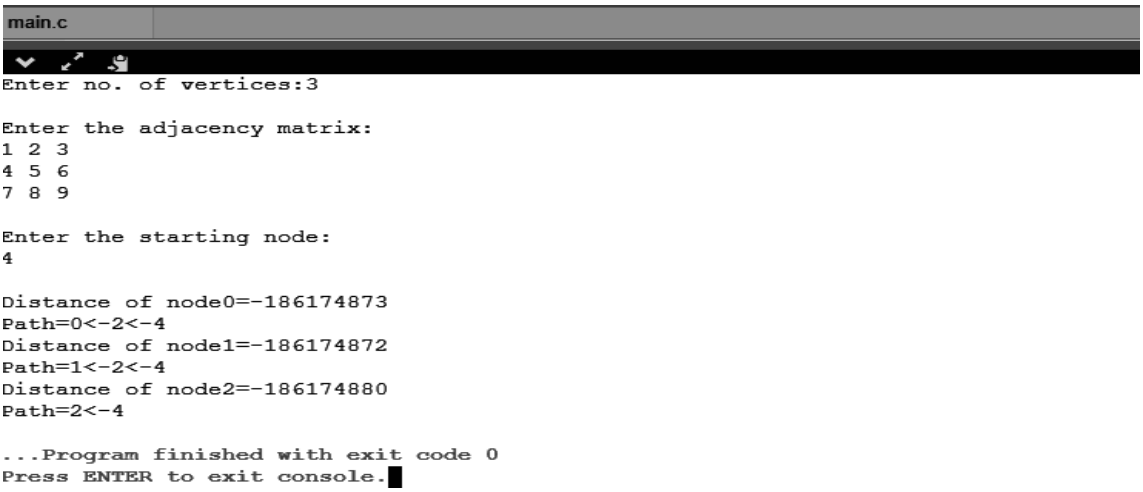
```
// PROGRAM FOR FINDING SHORTEST PATH FOR A GIVEN GRAPH//

#include<stdio.h>
#include<conio.h> #define
INFINITY 9999
#define MAX 10
 void dijkstra(int G[MAX][MAX],int n,intstartnode); int
main()
{
   int G[MAX][MAX],i,j,n,u;
   printf("Enter no. of vertices:");
   scanf("%d",&n);
   printf("\nEnter the adjacency matrix:\n");
    for(i=0;i<n;i++)
      for(j=0;j<n;j++)
         scanf("%d",&G[i][j]);
   printf("\nEnter the starting node:");
   scanf("%d",&u);
   dijkstra(G,n,u);
    return 0;
}
 void dijkstra(int G[MAX][MAX],int n,intstartnode)
{
 int cost[MAX][MAX],distance[MAX],pred[MAX]; int
   visited[MAX],count,mindistance,nextnode,i,j;
   for(i=0;i<n;i++)
      for(j=0;j<n;j++)
        if(G[i][j]==0)
            cost[i][j]=INFINITY; else
```

```
            cost[i][j]=G[i][j];
    for(i=0;i<n;i++)
   {
      distance[i]=cost[startnode][i];
      pred[i]=startnode; visited[i]=0;
    }
    distance[startnode]=0;
    visited[startnode]=1;
    count=1; while(count<n-
    1)

{
      mindistance=INFINITY;
      for(i=0;i<n;i++)
         if(distance[i]<mindistance&&!visited[i])
         {
            mindistance=distance[i];
            nextnode=i;
         }
         for(i=0;i<n;i++)
            if(!visited[i])
               if(mindistance+cost[nextnode][i]<distance[i])
               {
                                                    distance
                                                    [i]=min
                                                 } distance
                                          count++; +cost[n
                        }                        extnode
                                                 ][i];
                                                 pred[i]=
                                                 nextnod
                                                 e;
 for(i=0;i<n;i++)
      if(i!=startnode)
      {
        printf("\nDistance of node%d=%d",i,distance[i]); printf("\nPath=%d",i);
         j=i; do
         {
            j=pred[j]; printf("<-
            %d",j);
         }while(j!=startnode);
   }
}
```

OUTPUT:



Additional Programme

Program 1:- Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Define project 802.	CO2	Remember
2	Define protocol data unit.	CO2	Remember
3	Discuss the different types of networking or networking devices.	CO2	Remember
4	Define ICMP.	CO2	Remember
5	Describe the data units at different TCP/ IP protocol suite layers.	CO2	Understand
6	Describe the difference between ARP and RARP?	CO2	Understand
7	What is the minimum and maximum length of the header in the TCP segmentand the IP datagram?	CO2	Understand
8	What is the range of addresses in the class of Internet address?	CO2	Understand
9	What are the differences between FTP and TFTF application layer protocols?	CO2	Understand
10	Describe the major types of networks and explain them.	CO2	Understand
11	Define the virtual path.	CO2	Understand
12	Define packet filtering.	CO2	Understand
13	Describe the important topologies for networks.	CO2	Understand
14	What is a mesh network?	CO2	Remember
15	Describe the differences between broadband and basebandtransmission.	CO2	Understand
16	What it frequency spectrum	CO2	Remember
17	What is MAU?	CO2	Remember
18	Describe the difference between non-routable and routable protocols.	CO2	Understand
19	What is the importance of the OSI reference model?	CO2	Understand
20	Describe the logical link control.	CO2	Understand

EXPERIMENT - 5

5. Take an example subnet of hosts and obtain a broadcast tree for the subnet.

NAME OF THE EXPERIMENT: Broadcast Tree.

OBJECTIVE: Implement broadcast tree for a given subnet of hosts
RESOURCE: Turbo C

PROGRAM LOGIC:
This technique is widely used because it is simple and easy to understand. The idea of this algorithm is to build a graph of the subnet with each node of the graph representing a router and each arc of the graph representing a communication line. To choose a route between a given pair of routers the algorithm just finds the broadcast between them on the graph.

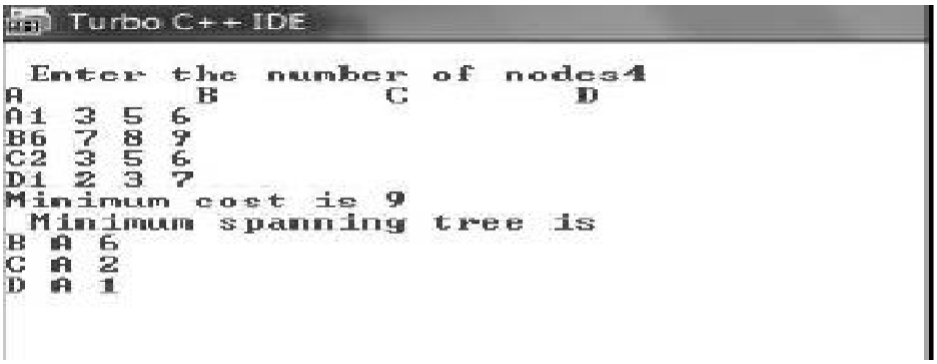
PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

SOURCE CODE:

```
#include<stdio.h> #include<conio.h> int p,q,u,v,n;
int min=99,mincost=0; int
t[50][2],i,j;
int parent[50],edge[50][50];
main(){
clrscr();
printf("\n Enter the number of nodes"); scanf("%d",&n);
for(i=0;i<n;i++){
printf("%c\t",65+i);
parent[i]=-1;
}
printf("\n");
for(i=0;i<n;i++)
{
printf("%c",65+i);
for(j=0;j<n;j++)
scanf("%d",&edge[i][j]);
}
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
if(edge[i][j]!=99)
if(min>edge[i][j])
{
min=edge[i][j]; u=i;
v=j;
}
p=find(u);
q=find(v);
if(p!=q)
{ t[i][0]=u;
```

```
t[i][1]=v; mincost=mincost+edge[u][v];
sunion(p,q);
}
Else
{
t[i][0]=-1;t[i][1]=-1;
}
min=99;
}
printf("Minimum cost is %d\n Minimum spanning tree is\n" ,mincost);
for(i=0;i<n;i++)
if(t[i][0]!=-1 && t[i][1]!=-1)
{
printf("%c %c %d", 65+t[i][0],65+t[i][1],edge[t[i][0]][t[i][1]]);printf("\n");
}
getch();
}
sunion(int l,int m)
{
parent[l]=m;
}
find(int l)
{
if(parent[l]>0)
i=parent[i]; return i;
}
```

Output:



Additional Programme

Program 1:- Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.

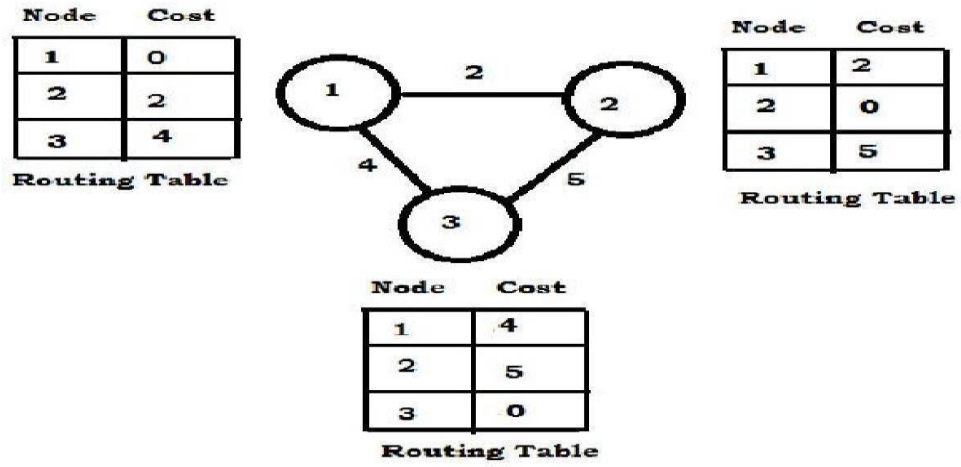
VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Define Network?	CO3	Understand
2	What is a Link?	CO3	Remember
3	What is a node?	CO3	Remember
4	What is a gateway or Router?	CO3	Remember
5	What is point-point link?	CO3	Remember
6	What is Multiple Access?	CO3	Remember
7	What are the advantages of Distributed Processing?	CO3	Understand
8	What are the criteria necessary for an effective and efficient network?	CO3	Understand
9	Name the factors that affect the performance of the network?	CO3	Understand
10	Name the factors that affect the reliability of the network?	CO3	Understand
11	Name the factors that affect the security of the network?	CO3	Understand
12	What is Protocol?	CO3	Remember
13	What are the key elements of protocols?	CO3	Remember
14	What are the key design issues of a computer Network?	CO3	Remember
15	Define Bandwidth and Latency?	CO3	Understand
16	Define Routing?	CO3	Understand
17	What is a peer-peer process?	CO3	Remember
18	When a switch is said to be congested?	CO3	Remember
19	What is semantic gap?	CO3	Remember
20	What is Round Trip Time?	CO3	Remember

EXPERIMENT - 6

6. Implement distance vector routing algorithm for obtaining routing tables at eachnode.

NAME OF HE EXPERIMENT: Distance Vector routing.
OBJECTIVE: Obtain Routing table at each node using distance vector routing algorithm for a given subnet.



RESOURCE: Turbo C

PROGRAM LOGIC:
Distance Vector Routing Algorithms calculate a best route to reach a destination based solely on distance. E.g. RIP. RIP calculates the reach ability based on hop count. It's different from link state algorithms which consider some other factors like bandwidth and other metrics to reach a destination. Distance vector routing algorithms are not preferable for complex networks and take longer to converge.

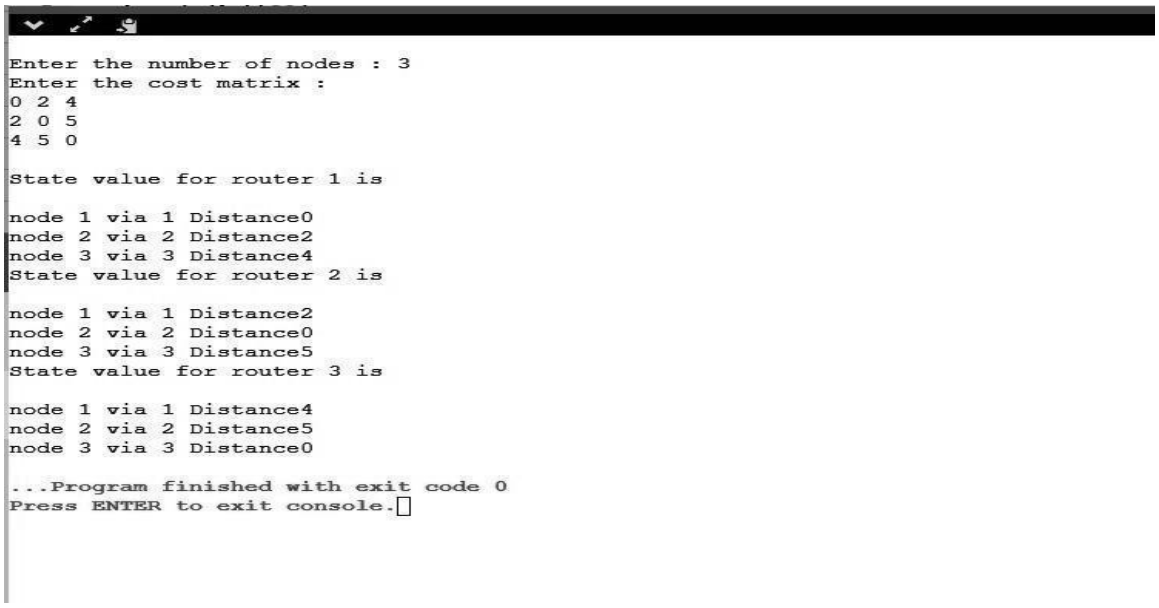
PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
struct node
{
unsigned dist[20];
unsigned from[20];
}rt[10];
int main(){
int dmat[20][20]; int n,i,j,k,count=0; clrscr();
printf("\nEnter the number of nodes : ");
scanf("%d",&n);printf("Enter the cost matrix :\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++){
scanf("%d",&dmat[i][j]);
dmat[i][i]=0;
rt[i].dist[j]=dmat[i][j];
rt[i].from[j]=j;
```

```
}
do
{
count=0;
for(i=0;i<n;i++)
for(j=0;j<n;j++)
for(k=0;k<n;k++)
if(rt[i].dist[j]>dmat[i][k]+rt[k].dist[j])
{
rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
rt[i].from[j]=k;count++;
}
}while(count!=0);
for(i=0;i<n;i++)
{
printf("\nState value for router %d is \n",i+1);
for(j=0;j<n;j++)
{
printf("\nnode %d via %d Distance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
}
}
printf("\n");
}
```

Output:



```
Enter the number of nodes : 3
Enter the cost matrix :
0 2 4
2 0 5
4 5 0

State value for router 1 is

node 1 via 1 Distance0
node 2 via 2 Distance2
node 3 via 3 Distance4
State value for router 2 is

node 1 via 1 Distance2
node 2 via 2 Distance0
node 3 via 3 Distance5
State value for router 3 is

node 1 via 1 Distance4
node 2 via 2 Distance5
node 3 via 3 Distance0

...Program finished with exit code 0
Press ENTER to exit console.
```

Additional Programme :

Program 1:- Write a program for error detecting code using CRC-CCITT (16- bits)

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Define Decoder?	CO1	Understand
2	Define Encoder?	CO1	Understand
3	What are Cyclic Codes?	CO1	Remember
4	What is a Linear Block Code?	CO1	Remember
5	What are Code Words?	CO1	Remember
6	What are Data Words?	CO1	Remember
7	Define Retransmission?	CO1	Understand
8	What is Forward Error Correction?	CO1	Remember
9	Compare Error Detection and Error Correction:	CO1	Understand
10	What are the Data link protocols?	CO1	Remember
11	List the steps involved in creating the checksum.	CO1	Understand
12	What is Checksum?	CO1	Remember
13	What is CRC?	CO1	Remember
14	What is LRC?	CO1	Remember
15	What is VRC?	CO1	Remember
16	What is Parity bit?	CO1	Remember
17	Define Redundancy?	CO1	Remember
18	What is Error Detection? What are its methods?	CO1	Remember
19	What are the types of errors?	CO1	Remember
20	What are the categories of Transmission media?	CO1	Remember
21	What are the different link types used to build a computer network?	CO1	Remember
22	What are the two classes of hardware building blocks?	CO1	Remember
23	What are the responsibilities of Application Layer?	CO1	Remember
24	What are the responsibilities of Presentation Layer?	CO1	Remember

25	What are the responsibilities of Session Layer?	CO1	Remember
26	What are the responsibilities of Transport Layer?	CO1	Understand
27	What are the responsibilities of Network Layer?	CO1	Remember
28	What are the responsibilities of Data Link Layer?	CO1	Understand
29	What are the concerns of the Physical Layer?	CO1	Understand
30	Which layer links the network support layers and user support layers?	CO1	Understand

EXPERIMENT - 7

7. Implement data encryption and data decryption

7. a. NAME OF THE EXPERIMENT: encrypting DES.

OBJECTIVE: Take a 64 bit playing text and encrypt the same using DES algorithm.

RESOURCE: Turbo C

PROGRAM LOGIC:

Data encryption standard was widely adopted by the industry in security products. Plain text is encrypted in blocks of 64 bits yielding 64 bits of cipher text. The algorithm which is parameterized by a 56 bit key has 19 distinct stages. The first stage is a key independent transposition and the last stage is exactly inverse of the transposition. The remaining stages are functionally identical but are parameterized by different functions of the key. The algorithm has been designed to allow decryption to be done with the same key as encryption.

PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

ALOGRITHM:

It is a simple type of substitution cipher, in this, each letter or word of a given text message is replaced by a letter some fixed number down the original alphabet.
We decide that fixed number, for example, if we select that number as 2 then A will be replaced by C, B will be replaced by D, and so on.
This fixed number here indicates the shift, which means the number of positions by which each letter of the text has to be moved down.

SOURCE CODE:

```
#include<stdio.h> int
main(){
char message[100], ch; int i,
key;
printf("Enter a message to encrypt: "); gets(message);
printf("Enter key: ");
scanf("%d", &key);
for(i = 0; message[i] != '\0'; ++i){ ch
= message[i];
if(ch>= 'a' &&ch<= 'z'){ ch
= ch + key;
if(ch> 'z'){
ch = ch - 'z' + 'a' - 1;
}
message[i] = ch;
}
else if(ch>= 'A' &&ch<= 'Z'){ ch
= ch + key;
if(ch> 'Z'){
ch = ch - 'Z' + 'A' - 1;
}
message[i] = ch;}
```

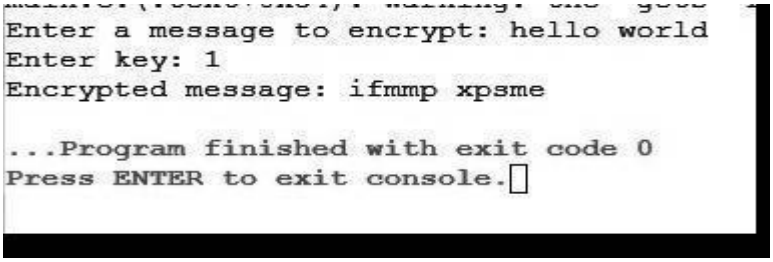
```
}
printf("Encrypted message: %s", message);
return 0;

}
```

RESULT:

The above program is executed and compiled successfully without any errors.

OUTPUT:



7. b. NAME OF THE EXPERIMENT: Decrypting DES.

OBJECTIVE: Write a program to break the above DES coding

RESOURCE: Turbo C

PROGRAM LOGIC:
Data encryption standard was widely adopted by the industry in security products. Plain text is encrypted in blocks of 64 bits yielding 64 bits of cipher text. The algorithm which is parameterized by a 56 bit key has 19 distinct stages. The first stage is a key independent transposition and the last stage is exactly inverse of the transposition. The remaining stages are functionally identical but are parameterized by different functions of the key. The algorithm has been designed to allow decryption to be done with the same key as encryption.

PROCEDURE: Go to debug -> run or press CTRL + F9 to run the program.

ALOGRITHM:

It is a simple type of substitution cipher, in this, each letter or word of a given text message is replaced by a letter some fixed number down the original alphabet.
We decide that fixed number, for example, if we select that number as 2 then A will be replaced by C, B will be replaced by D, and so on.
This fixed number here indicates the shift, which means the number of positions by which each letter of the text has to be moved down.

SOURCE CODE:

```
/*Write a program to break the above DES coding*/
#include<stdio.h> int
main(){
char message[100], ch; int i,
key;
printf("Enter a message to decrypt: "); gets(message);
```

```
printf("Enter key: ");
scanf("%d", &key);
for(i = 0; message[i] != '\0'; ++i){ ch
= message[i];
if(ch>= 'a' &&ch<= 'z'){ ch
= ch - key;
if(ch< 'a'){
ch = ch + 'z' - 'a' + 1;
}
message[i] = ch;
}
else if(ch>= 'A' &&ch<= 'Z'){ ch
= ch - key;
if(ch< 'A'){
ch = ch + 'Z' - 'A' + 1;
}
message[i] = ch;
} }
printf("Decrypted message: %s", message);
return 0;
}
```

RESULT:

The above program is executed and compiled successfully without any errors. OUTPUT:

```
Enter a message to decrypt: ifmmp xpsme
Enter key: 1
Decrypted message: hello world

...Program finished with exit code 0
Press ENTER to exit console.
```

Additional Pragramme

Program 1:- Write a program to find the shortest path between vertices using the bellman-ford algorithm

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What is Fixed Size Framing?	CO1	Understand
2	Define Character Stuffing?	CO1	Remember
3	What is Bit Stuffing?	CO1	Understand
4	What is Flow Control?	CO1	Remember
5	What is Error Control ?	CO1	Understand
6	What Automatic Repeat Request (ARQ)?	CO1	Remember
7	What is Stop-and-Wait Protocol?	CO1	Understand
8	What is Stop-and-Wait Automatic Repeat Request?	CO1	Understand
9	What is usage of Sequence Number in Relaible Transmission?	CO1	Understand
10	What is Pipelining ?	CO1	Remember
11	What is Sliding Window?	CO1	Understand
12	What is Piggy Backing?	CO1	Remember
13	What are the two types of transmission technology available?	CO1	Understand
14	What is subnet?	CO1	Understand
15	Difference between the communication and transmission.	CO1	Understand
16	What are the possible ways of data exchange?	CO1	Remember
17	What is SAP?	CO1	Understand
18	What do you meant by "triple X" in Networks?	CO1	Remember
19	What is frame relay, in which layer it comes?	CO1	Understand
20	What is terminal emulation, in which layer it comes?	CO1	Remember
21	What is Beaconing?	CO1	Understand
22	What is redirector?	CO1	Understand
23	What is NETBIOS and NETBEUI?	CO1	Remember
24	What is RAID?	CO1	Understand

25	What is passive topology?	CO1	Understand
26	What is Brouter?	CO1	Understand
27	What is cladding?	CO1	Remember
28	What is point-to-point protocol?	CO1	Understand
29	How Gateway is different from Routers?	CO1	Remember
30	What is attenuation?	CO1	Understand

EXPERIMENT - 8

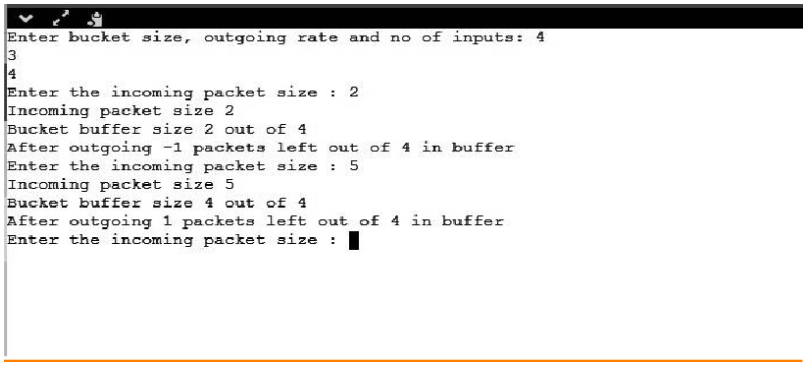
8. **Write a program for congestion control using Leaky bucket algorithm.**

SOURCE CODE:

```
#include<stdio.h> int
main(){
int incoming, outgoing, buck_size, n, store = 0; printf("Enter
bucket size, outgoing rate and no of inputs: "); scanf("%d %d
%d", &buck_size, &outgoing, &n);

while (n != 0) {
printf("Enter the incoming packet size : "); scanf("%d",
&incoming);
printf("Incoming packet size %d\n", incoming); if
(incoming <= (buck_size - store)){
store += incoming;
printf("Bucket buffer size %d out of %d\n", store, buck_size);
      } else {
printf("Dropped %d no of packets\n", incoming - (buck_size - store));
printf("Bucket buffer size %d out of %d\n", store, buck_size);
store = buck_size;
      }
store = store - outgoing;
printf("After outgoing %d packets left out of %d in buffer\n", store, buck_size); n-
-;
   }
}
```

OUTPUT:



Additional Progamme

Program 1:- Using TCP/IP sockets, write a client-server program to make the client send the file name and to make the server send back the contents of the requested file if present.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What is MAC address?	CO4	Understand
2	Difference between bit rate and baud rate.	CO4	Understand
3	What is Bandwidth?	CO4	Understand
4	What are the types of Transmission media?	CO4	Remember
5	What is Project 802?	CO4	Understand
6	What is Protocol Data Unit?	CO4	Remember
7	What are the different type of networking / internetworking devices?	CO4	Understand
8	What is ICMP?	CO4	Remember
9	What are the data units at different layers of the TCP / IP protocol suite?	CO4	Understand
10	What is difference between ARP and RARP?	CO4	Understand
11	What is the minimum and maximum length of the header in the TCP segment and IP datagram?	CO4	Understand
12	What is the range of addresses in the classes of internet addresses?	CO4	Understand
13	What is the difference between TFTP and FTP application layer protocols?	CO4	Understand
14	What are major types of networks and explain?	CO4	Remember
15	What are the important topologies for networks?	CO4	Understand
16	What is mesh network?	CO4	Remember
17	What is the difference between routable and non- routable protocols?	CO4	Understand
18	Why should you care about the OSI Reference Model?	CO4	Remember
19	What is logical link control?	CO4	Remember
20	What is Medium access Control	CO4	Remember

EXPERIMENT - 9

9. **Write a program for frame sorting technique used in buffers.**

Objective: to implement frame sorting technique used in buffers.

SOURCE CODE:

```
#include<stdio.h> #include<string.h>
#define FRAM_TXT_SIZ 3
#define MAX_NOF_FRAM 127
char str[FRAM_TXT_SIZ*MAX_NOF_FRAM];
struct frame // structure maintained to hold frames
{ char text[FRAM_TXT_SIZ]; int
seq_no;
}fr[MAX_NOF_FRAM], shuf_ary[MAX_NOF_FRAM]; int
assign_seq_no() //function which splits message
{ int k=0,i,j; //into frames and assigns sequence no
for(i=0; i < strlen(str); k++)
{ fr[k].seq_no = k;
for(j=0; j < FRAM_TXT_SIZ && str[i]!='\0'; j++)
fr[k].text[j] = str[i++];
}
printf("\nAfter assigning sequence numbers:\n");
for(i=0; i < k; i++)
printf("%d:%s ",i,fr[i].text);
return k; //k gives no of frames
}
void generate(int *random_ary, const int limit) //generate array of random nos
{ int r, i=0, j; while(i
< limit)
{ r = random() % limit;
for(j=0; j < i; j++)
if( random_ary[j] == r )
break;
if( i==j ) random_ary[i++] = r;} }
void shuffle( const int no_frames ) // function shuffles the frames
{
int i, k=0, random_ary[no_frames];
generate(random_ary, no_frames);
for(i=0; i < no_frames; i++) shuf_ary[i] =
fr[random_ary[i]]; printf("\n\nAFTER
SHUFFLING:\n"); for(i=0; i <
no_frames; i++)
printf("%d:%s ",shuf_ary[i].seq_no,shuf_ary[i].text);
}
```

```
void sort(const int no_frames) // sorts the frames
{int i,j,flag=1; struct frame hold;
for(i=0; i < no_frames-1 && flag==1; i++) // search for frames in sequence
{flag=0;
for(j=0; j < no_frames-1-i; j++) //(based on seq no.) and display
if(shuf_ary[j].seq_no > shuf_ary[j+1].seq_no)
{
hold = shuf_ary[j]; shuf_ary[j]
= shuf_ary[j+1]; shuf_ary[j+1]
= hold; flag=1;
}
}
}
int main()
{
int no_frames,i;
printf("Enter the message: ");
gets(str);
no_frames = assign_seq_no();
shuffle(no_frames); sort(no_frames);
printf("\n\nAFTER SORTING\n");
for(i=0;i<no_frames;i++)
printf("%s",shuf_ary[i].text);
printf("\n\n");
}
```

OUTPUT :

```
Enter the message: welcome to lab

After assigning sequence numbers:
0:wel 1:com 2:e t 3:o l 4:ab

AFTER SHUFFLING:
3:o l 1:com 2:e t 0:wel 4:ab

AFTER SORTING
welcome to lab
```

```
Enter the message: computer

After assigning sequence numbers:
0:com 1:put 2:er

AFTER SHUFFLING:
1:put 0:com 2:er

AFTER SORTING
computer
```

additional Pragramme

Program 1:- Using TCP/IP sockets, write a client-server program to make the client send the file name and to make the server send back the contents of the requested file if present.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What are functions of different layers?	CO4	Remember
2	Differentiate between TCP/IP Layers and OSI Layers	CO4	Analyze
3	Differentiate between TCP and UDP.	CO4	Analyze
4	Differentiate between Connectionless and connection oriented connection.	CO4	Understand
5	What is meant by subnet?	CO4	Understand
6	What is meant by Gateway?	CO4	Understand
7	What is an IP address?	CO4	Understand
8	What is MAC address?	CO4	Understand
9	Define Raw Socket	CO4	Understand
10	What is a fork command?	CO4	Remember
11	Why IP address is required when we have MAC address?	CO4	Understand
12	What is meant by port?	CO4	Remember
13	What are ephemerical port number and well known port numbers?	CO4	Understand
14	What is a socket?	CO4	Remember
15	What are the parameters of socket()?	CO4	Understand
16	Describe bind(), listen(), accept(),connect(), send() and recv().	CO4	Understand
17	What are system calls? Mention few of them.	CO4	Understand
18	What is IPC? Name three techniques.	CO4	Understand
19	What type of protocol is BGP?	CO4	Remember
20	What type of protocol is OSPF?	CO4	Understand

EXPERIMENT 10

Wireshark

Introduction to **Wire Shark**

Wireshark is an open-source network protocol analysis software program started by Gerald Combs in 1998. A global organization of network specialists and software developers support Wireshark and continue to make updates for new network technologies and encryption methods. Wireshark is absolutely safe to use. Government agencies, corporations, non-profits, and educational institutions use Wireshark for troubleshooting and teaching purposes. There isn't a better way to learn networking than to look at the traffic under the Wireshark microscope.

There are questions about the legality of Wireshark since it is a powerful packet sniffer. The Light side of the Force says that you should only use Wireshark on networks where you have permission to inspect network packets. Using Wireshark to look at packets without permission is a path to the Dark Side.

How does Wireshark work?

Wireshark is a packet sniffer and analysis tool. It captures network traffic on the local network and stores that data for offline analysis. Wireshark captures network traffic from Ethernet, Bluetooth, Wireless (IEEE.802.11), Token Ring, Frame Relay connections, and more.

Ed. Note: A "packet" is a single message from any network protocol (i.e., TCP, DNS, etc.)
Ed. Note 2: LAN traffic is in broadcast mode, meaning a single computer with Wireshark can see traffic between two other computers. If you want to see traffic to an external site, you need to capture the packets on the local computer.
Wireshark allows you to filter the log either before the capture starts or during analysis, so you can narrow down and zero into what you are looking for in the network trace. For example, you can set a filter to see TCP traffic between two IP addresses. You can set it only to show you the packets sent from one computer. The filters in Wireshark are one of the primary reasons it became the standard tool for packet analysis.

How to Download Wireshark

Downloading and installing Wireshark is easy. Step one is to check the official Wireshark Download page for the operating system you need. The basic version of Wireshark is free.

Wireshark for Windows

Wireshark comes in two flavors for Windows, 32 bit and 64 bit. Pick the correct version for your OS. The current release is 3.0.3 as of this writing. The installation is simple and shouldn't cause any issues.

Wireshark for Mac Wireshark is available on Mac as a Homebrew install.

To install Homebrew, you need to run this command at your Terminal prompt:

```
/usr/bin/ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Once you have the Homebrew system in place, you can access several open-source projects for your Mac. To install Wireshark run this command from the Terminal:

```
brew install wireshark
```

Homebrew will download and install Wireshark and any dependencies so it will run correctly.

Wireshark for Linux Installing Wireshark on Linux can be a little different depending on the Linux distribution. If you aren't running one of the following distros, please double-check the commands.

Ubuntu

From a terminal prompt, run these commands:

1. `sudo apt-get install wireshark`
2. `sudodpkg-reconfigure wireshark-common`
3. `sudoadduser $USER wireshark`

Those commands download the package, update the package, and add user privileges to run Wireshark.

Red Hat Fedora

From a terminal prompt, run these commands:

1. `sudodnf install wireshark-qt`
2. `sudousermod -a -G wireshark username`

The first command installs the GUI and CLI version of Wireshark, and the second adds permissions to use Wireshark.

Kali Linux

Wireshark is probably already installed! It's part of the basic package. Check your menu to verify. It's under the menu option "Sniffing & Spoofing."

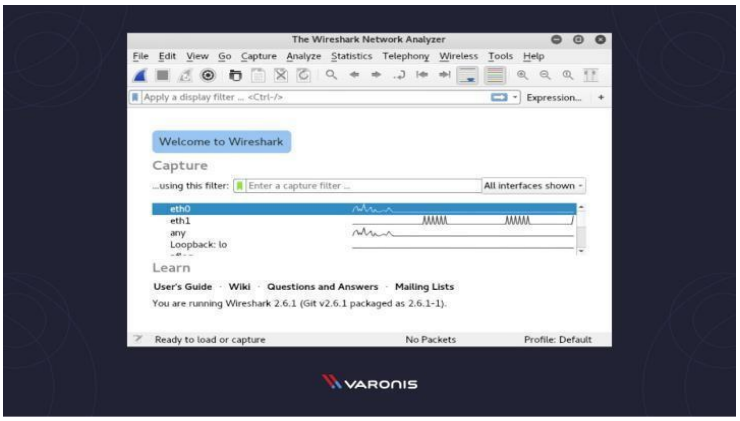
Data Packets on Wireshark

Now that we have Wireshark installed let's go over how to enable the Wireshark packet sniffer and then analyze the network traffic.

i. i) Capturing Data Packets on Wireshark

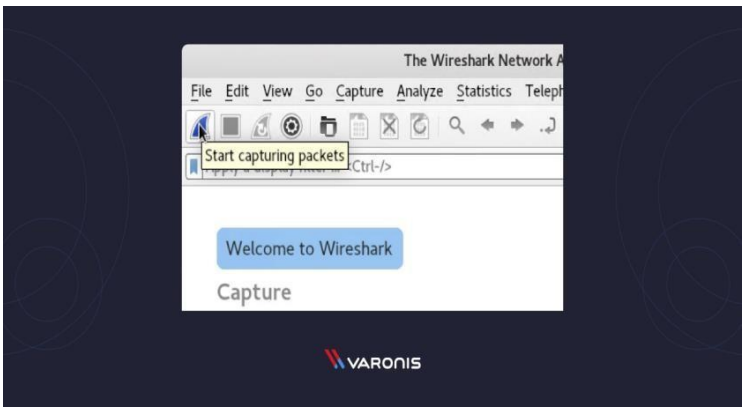
When you open Wireshark, you see a screen that shows you a list of all of the network connections you can monitor. You also have a capture filter field, so you only capture the network traffic you want to see.

You can select one or more of the network interfaces using "shift left-click." Once you have the

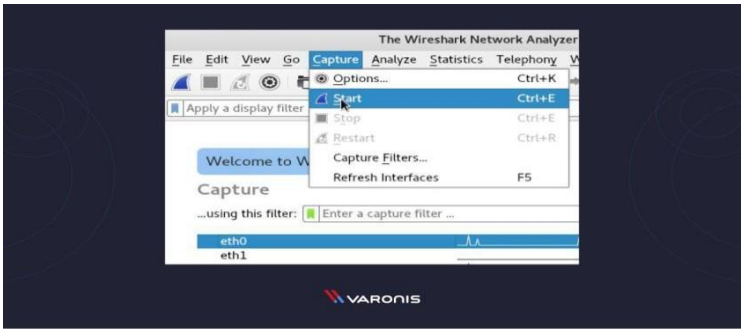


network interface selected, you can start the capture, and there are several ways to do that.

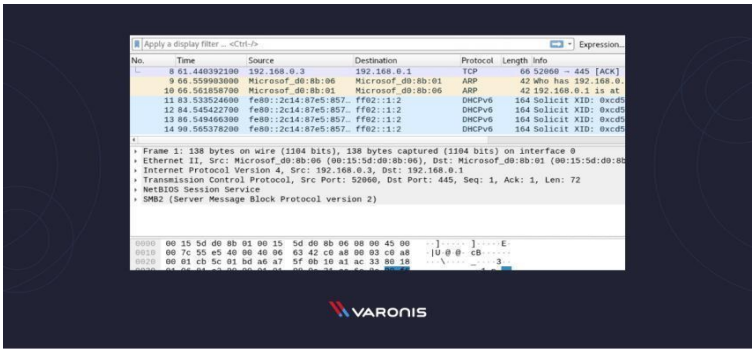
Click the first button on the toolbar, titled "Start Capturing Packets."



You can select the menu item Capture -> Start.



Or you could use the keystroke Control – E.
During the capture, Wireshark will show you the packets that it captures in real-time.



Once you have captured all the packets you need, you use the same buttons or menu options to stop the capture.

Best practice says that you should stop Wireshark packet capture before you do analysis.

iv) Analyzing Data Packets on Wireshark

Wireshark shows you three different panes for inspecting packet data. The Packet List, the top pane, is a list of all the packets in the capture. When you click on a packet, the other two panes change to show you the details about the selected packet. You can also tell if the packet is part of a conversation. Here are some details about each column in the top pane:

- **No.**: This is the number order of the packet that got captured. The bracket indicates that this packet is part of a conversation.
- **Time**: This column shows you how long after you started the capture that this packet got captured. You can change this value in the Settings menu if you need something different displayed.
- **Source**: This is the address of the system that sent the packet.
- **Destination**: This is the address of the destination of that packet.
- **Protocol**: This is the type of packet, for example, TCP, DNS, DHCPv6, or ARP.
- **Length**: This column shows you the length of the packet in bytes.
- **Info**: This column shows you more information about the packet contents, and will vary depending on what kind of packet it is.

Packet Details, the middle pane, shows you as much readable information about the packet as possible, depending on what kind of packet it is. You can right-click and create filters based on the highlighted text in this field.

The bottom pane, Packet Bytes, displays the packet exactly as it got captured in hexadecimal.

When you are looking at a packet that is part of a conversation, you can right-click the packet and select Follow to see only the packets that are part of that conversation.

Wireshark Filters

One of the best features of Wireshark is the Wireshark Capture Filters and Wireshark Display Filters. Filters allow you to view the capture the way you need to see it so you can troubleshoot the issues at hand. Here are several filters to get you started.

Wireshark Capture Filters

Capture filters limit the captured packets by the filter. Meaning if the packets don't match the filter, Wireshark won't save them. Here are some examples of capture filters:
host IP-*address*: this filter limits the capture to traffic to and from the IP address net 192.168.0.0/24: this filter captures all traffic on the subnet.

dst host IP-*address*: capture packets sent to the specified host.
port 53: capture traffic on port 53 only.

port not 53 and not arp: capture all traffic except DNS and ARP traffic

Wireshark Display Filters

Wireshark Display Filters change the view of the capture during analysis. After you have stopped the packet capture, you use display filters to narrow down the packets in the Packet List so you can troubleshoot your issue.
The most useful (in my experience) display filter is:

ip.src==*IP-address* and ip.dst==*IP-address*
This filter shows you packets from one computer (ip.src) to another (ip.dst). You can also use ip.addr to show you packets to and from that IP. Here are some others:

tcp.port eq 25: This filter will show you all traffic on port 25, which is usually SMTP traffic. icmp:

This filter will show you only ICMP traffic in the capture, most likely they are pings.

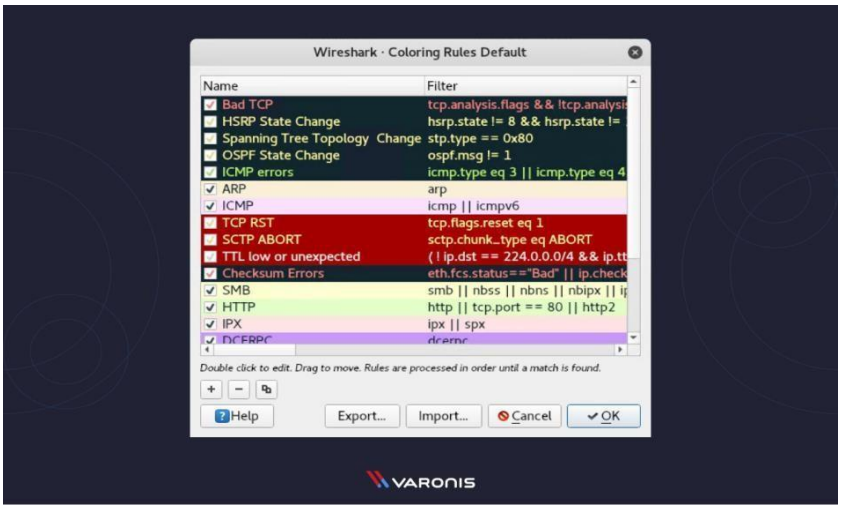
ip.addr != *IP_address*: This filter shows you all traffic except the traffic to or from the specified computer.
Analysts even build filters to detect specific attacks, like this filter to detect the Sasser worm:
ls_ads.opnum==0x09

Additional Wireshark Features

Beyond the capture and filtering, there are several other features in Wireshark that can make your life better.

Wireshark Colorization Options

You can setup Wireshark so it colors your packets in the Packet List according to the display filter, which allows you to emphasize the packets you want to highlight. Check out some examples here.



Wireshark Promiscuous Mode

By default, Wireshark only captures packets going to and from the computer where it runs. By checking the box to run Wireshark in Promiscuous Mode in the Capture Settings, you can capture most of the traffic on the LAN.

Wireshark Command Line

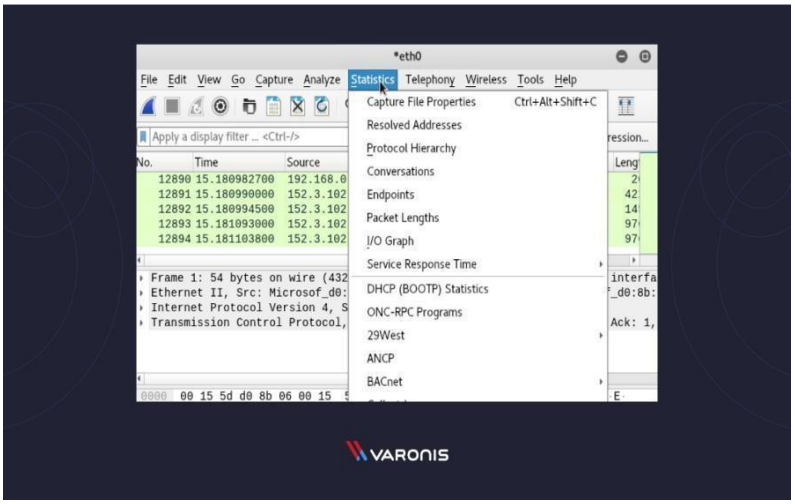
Wireshark does provide a Command Line Interface (CLI) if you operate a system without a GUI. Best practice would be to use the CLI to capture and save a log so you can review the log with the GUI.

Wireshark Commands

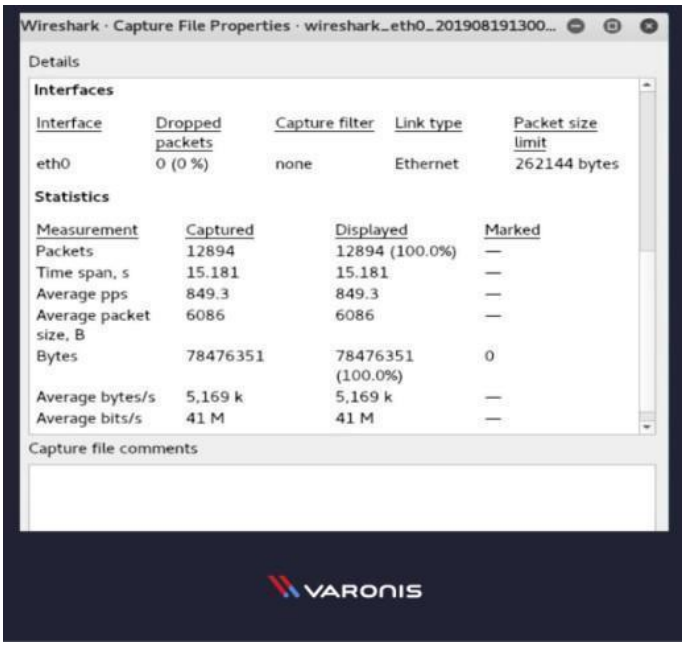
- wireshark : run Wireshark in GUI mode
- wireshark –h : show available command line parameters for Wireshark
- wireshark –a duration:300 –i eth1 –w wireshark. : capture traffic on the Ethernet interface 1 for 5 minutes. –a means automatically stop the capture, -i specifics which interface to capture

Metrics and Statistics

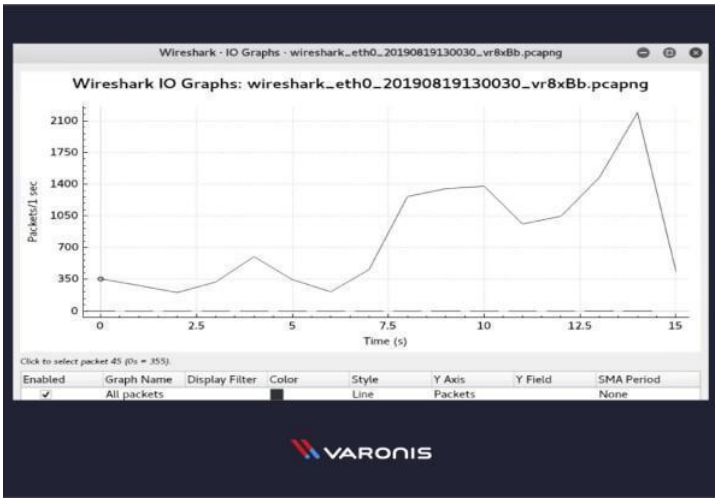
Under the Statistics menu item, you will find a plethora of options to show details about your capture.



Capture File Properties:



Wireshark I/O Graph:



i. **Packet Capture Using Wireshark**
ii. **Starting Wireshark**
iii. **Viewing Captured Traffic**
iv. **Analysis and Statistics &Filters.**

1. **Use Wireshark to perform a packet capture of network traffic**

In order to investigate your issues further, we would like to run an analysis of the traffic being sent between you and Salesforce. To do this, we will use the Wireshark application. Wireshark is a tool that allows packet traces to be monitored, captured and analysed. Please follow the steps below in order to obtain a capture of your network traffic using Wireshark.
Note: *You will require some administrator rights on your machine in order to complete these tests. If you are unsure, please contact your IT administrator.*

Installing the Wireshark package

Visit the Wireshark download site, and download the appropriate Wireshark package or installer for the operating system running on the system which is to be used for packet capture.

When installing, ensure all components are selected for installation, including the optional "Winpcap" application.

Once complete, start Wireshark via shortcut or start menu.

Capturing your traffic with Wireshark

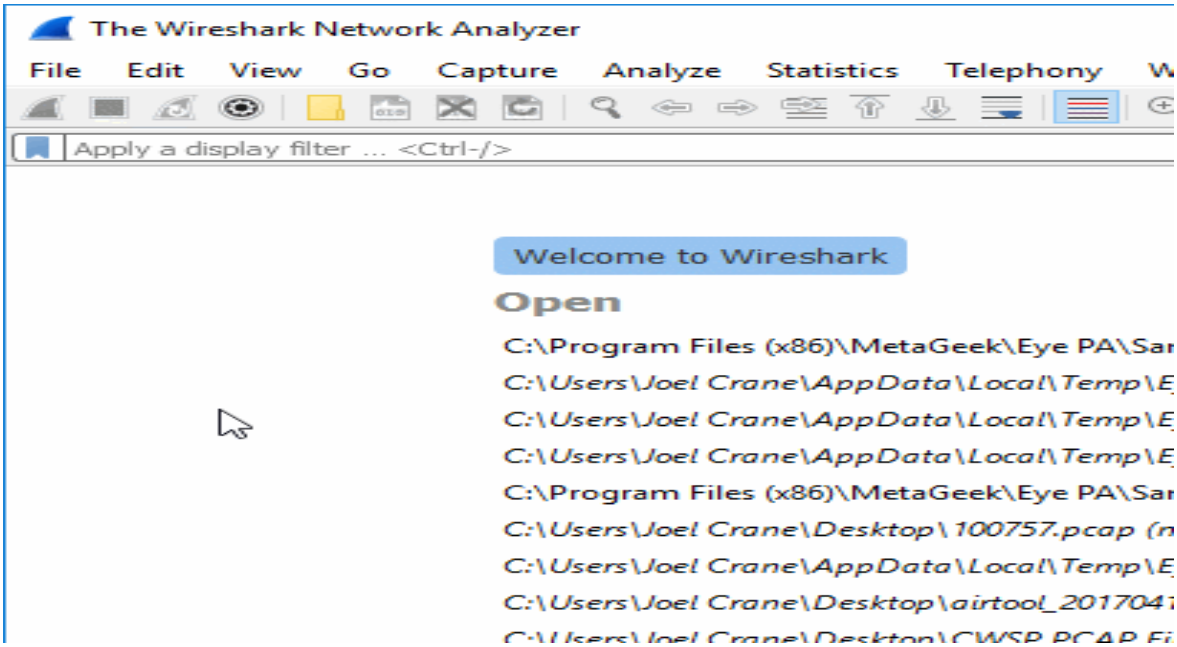
After starting Wireshark, do the following:

1. Select **Capture** | **Interfaces**
2. Select the interface on which packets need to be captured. This will usually be the interface where the Packet/s column is constantly changing, which would indicate the presence of live traffic). If you have multiple network interface cards (i.e. LAN card and Wi-Fi adapter) you may need to check with your IT administrator to determine the right interface.
3. Click the **Start** button to start the capture.
4. Recreate the problem. The capture dialog should show the number of packets increasing. Try to avoid running any other internet applications while capturing, closing other browsers, Instant messengers etc.
5. Once the problem which is to be analyzed has been reproduced, click on **Stop**. It may take a few seconds for Wireshark to display the packets captured.
6. Save the packet trace in the default format. Click on the **File** menu option and select **Save As**. By default Wireshark will save the packet trace in libpcap format. This is a filename with a.pcap extension.

Returning the information to Salesforce support

Forward the resulting .pcap file to your support representative, either by email, or attaching it to your open case. Please also include the following information:

- Your external IP address (get this from http://www.whatismyip.com/)
- The internal IP address of the local machine where traffic is being captured
- A click path of the steps you took to reproduce, including links to each page/record accessed

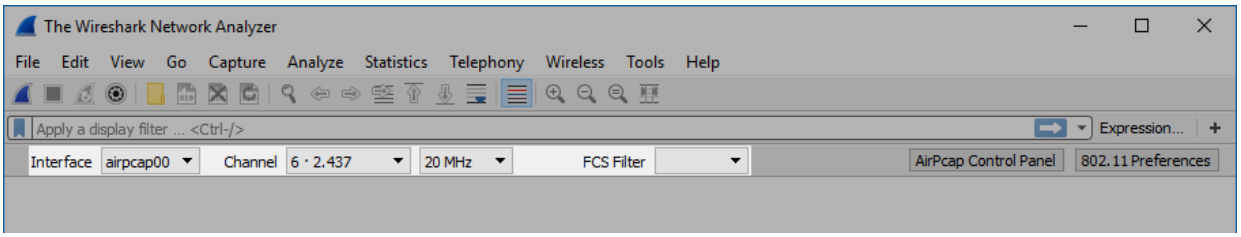


In some cases, you may want to perform packet captures with Wireshark. One case might be when you want to perform a packet capture on channel 12 or 13.

Set up the Packet Capture

1. Click **View** > **Wireless Toolbar.** The Wireless Toolbar will appear just below the Main toolbar.

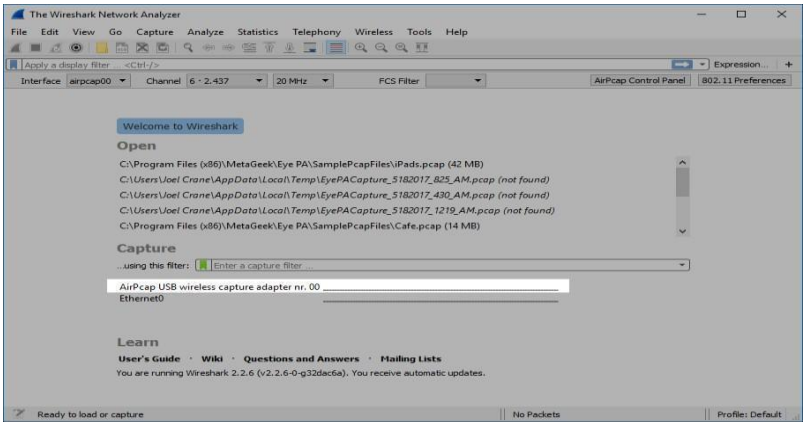
2. Use the **Wireless Toolbar** to configure the desired channel and channel width.



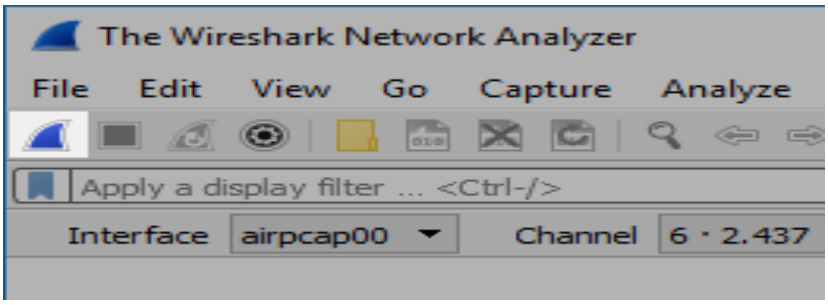
3. Under **Capture**, click on **AirPcap USB wireless capture adapter** to select the capture interface.

*Note: If the AirPcap isn't listed, press **F5** to refresh the list of available packet capture interfaces.*

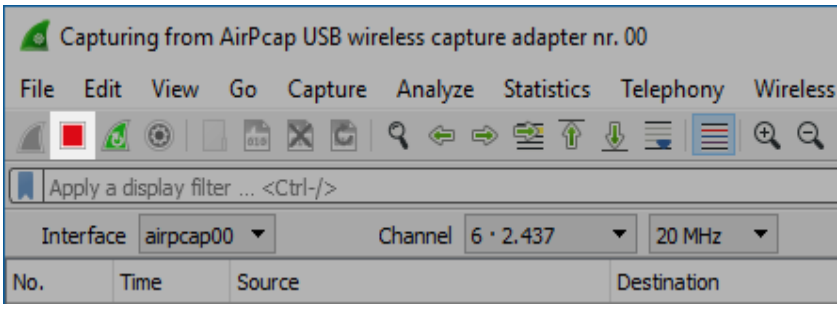
Note: The AirPcap has been discontinued by RiverBed and is 802.11n only.



4. Click the **Start Capture** button to begin the capture.

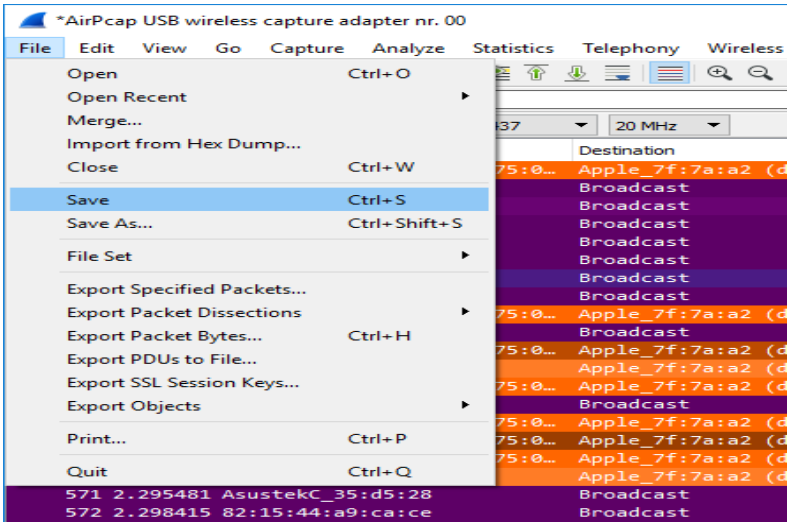


5. When you are finished capturing, click the **Stop** button.



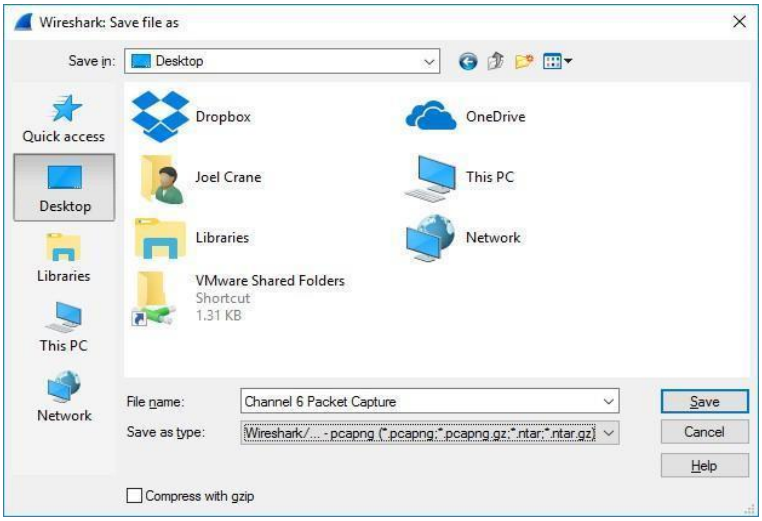
Saving the Capture

1. To save the capture, click **File** > **Save**.



2. Name the file, and click **Save**.

Note: ***.Pcap*** *and* ***.Pcap-ng*** *are good filetypes to use for the capture if you plan to use Eye P.A. to open the capture.*



3. Eye P.A. can now open the capture file.

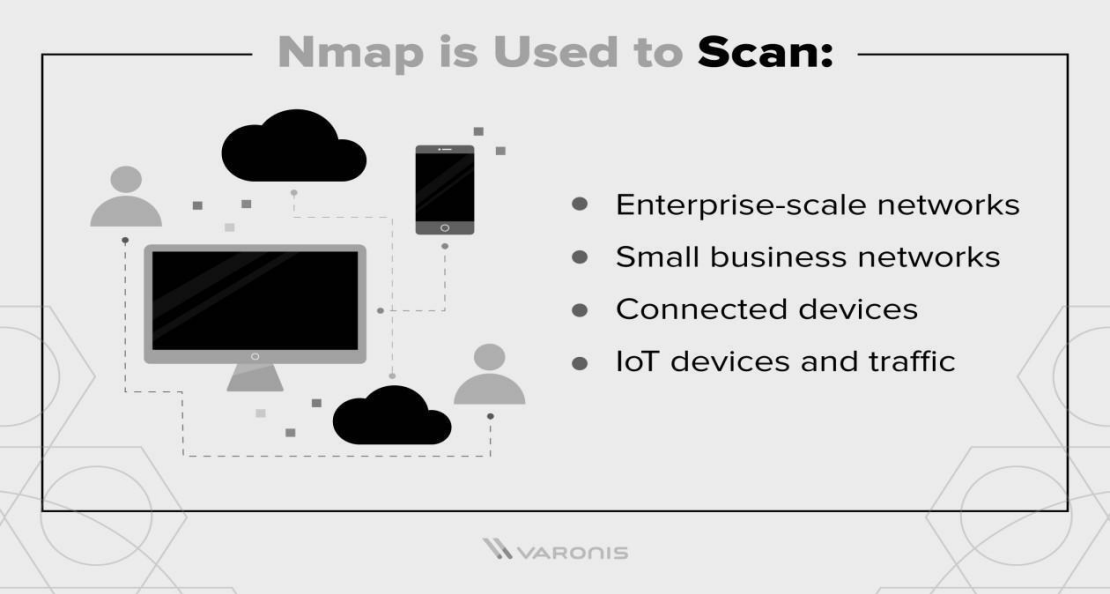
VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What Is Wireshark?	CO5	Understand
2	How Would You Setup Wireshark To Monitor Packets Passing Through An Internet Router?	CO5	Understand
3	Can Wireshark Be Setup On A Cisco Router?	CO5	Understand
4	Is It Possible To Start Wireshark From Command Line On Windows?	CO5	Understand
5	A User Is Unable To Ping A System On The Network. How Can Wireshark Be Used To Solve The Problem?	CO5	Understand
6	Which Wireshark Filter Can Be Used To Check All Incoming Requests To A Http Web Server?	CO5	Understand
7	Which Wireshark Filter Can Be Used To Monitor Outgoing Packets From A Specific System On The Network?	CO5	Understand
8	What's Up With The Name Change? Is Wireshark A Fork?	CO5	Understand
9	What Kind Of Shark Is Wireshark?	CO5	Understand
10	What Do You Think Of Wireshark?	CO5	Understand
11	Question eleven. How To Remove Wireshark Antivirus From My Computer?	CO5	Understand
12	How Do I Use Wireshark To Find A Password In My Network?	CO5	Understand
13	Why Does Wireshark Not Detect My Wireless Cards?	CO5	Understand
14	How Do I Modify Wireshark Packets On The Fly?	CO5	Understand
15	Why Don't The Packets I'm Capturing Have Vlan Tags?	CO5	Understand
16	How Can I Capture Packets WithCrc Errors?	CO5	Understand
17	What Causes Wireshark Error?	CO5	Understand
18	What Exactly Does Wireshark Do?	CO5	Understand
19	Can't Install Wireshark?	CO5	Understand
20	What does Cisco IOS stand for?	CO5	Remembering
21	Name three common Cisco networking devices.	CO5	Remembering
22	What is the purpose of a Cisco router?	CO5	Remembering

23	What command is used to display the routing table on a Cisco router?	CO5	Remembering
24	Explain the difference between a switch and a router.	CO5	Understanding
25	Describe how VLANs improve network performance.	CO5	Understanding
26	What is the function of the Spanning Tree Protocol (STP) in Cisco switches?	CO5	Understanding
27	How would you configure a basic IP address on a Cisco router interface?	CO5	Applying
28	Show how to configure a VLAN on a Cisco switch.	CO5	Applying
29	Apply ACL (Access Control List) to restrict access on a Cisco router.	CO5	Applying
30	Analyze why a routing loop might occur in a Cisco network.	CO5	Analyzing

Experiment 11

What is Nmap?

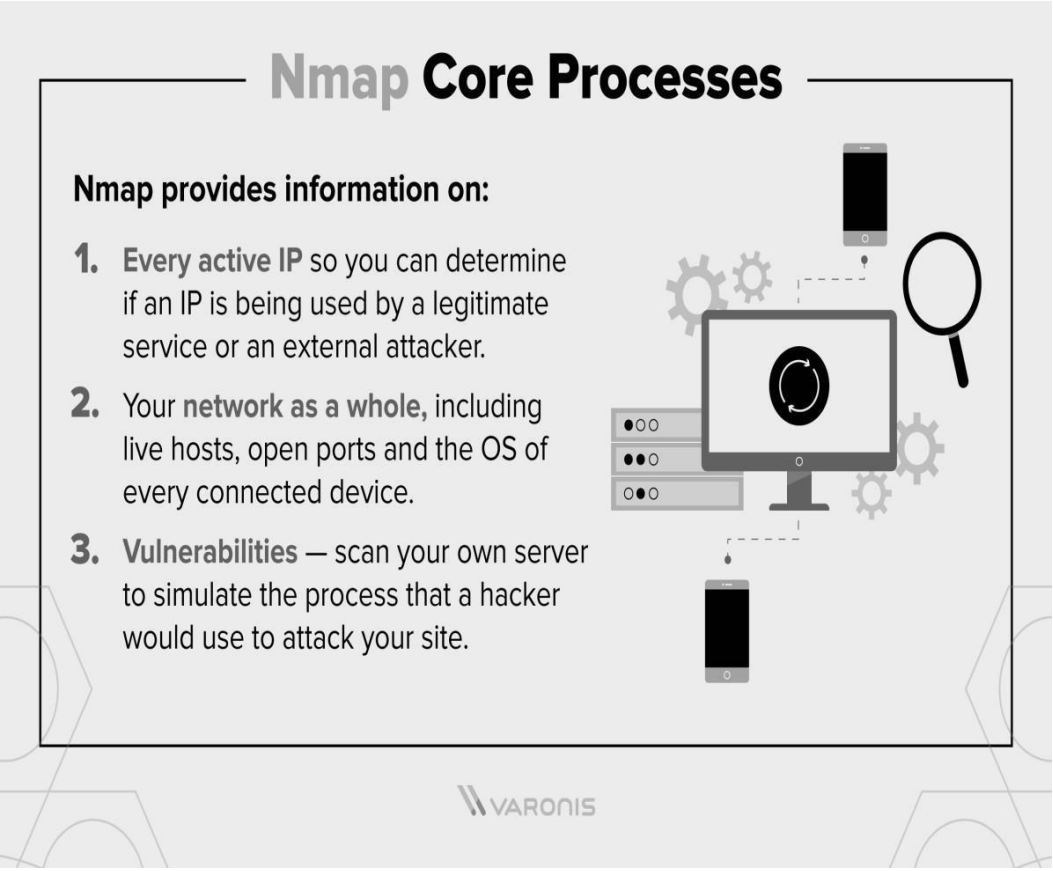


At its core, Nmap is a network scanning tool that uses IP packets to identify all the devices connected to a network and to provide information on the services and operating systems they are running.

The program is most commonly used via a command-line interface (though GUI front-ends are also available) and is available for many different operating systems such as Linux, Free BSD, and Gentoo. Its popularity has also been bolstered by an active and enthusiastic user support community.

Nmap was developed for enterprise-scale networks and can scan through thousands of connected devices. However, in recent years Nmap is being increasingly used by smaller companies. The rise of the IoT, in particular, now means that the networks used by these companies have become more complex and therefore harder to secure.
This means that Nmap is now used in many website monitoring tools to audit the traffic between web servers and IoT devices. The recent emergence of IoT botnets, like Mirai, has also stimulated interest in Nmap, not least because of its ability to interrogate devices connected via the UPnP protocol and to highlight any devices that may be malicious.

What Does Nmap Do?

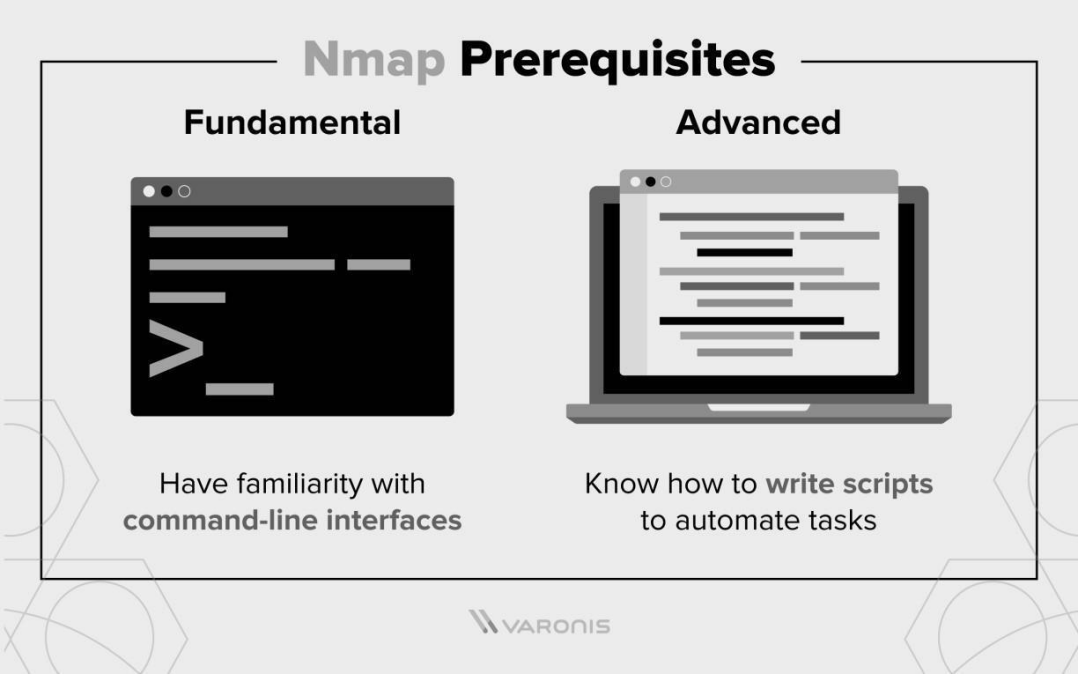


At a practical level, Nmap is used to provide detailed, real-time information on your networks, and on the devices connected to them.

The primary uses of Nmap can be broken into three core processes. First, the program gives you detailed information on every IP active on your networks, and each IP can then be scanned. This allows administrators to check whether an IP is being used by a legitimate service, or by an external attacker.

Secondly, Nmap provides information on your network as a whole. It can be used to provide a list of live hosts and open ports, as well as identifying the OS of every connected device. This makes it a valuable tool in ongoing system monitoring, as well as a critical part of pentesting. Nmap can be used alongside the Metasploit framework, for instance, to probe and then repair network vulnerabilities.
Thirdly, Nmap has also become a valuable tool for users looking to protect personal and business websites. Using Nmap to scan your own web server, particularly if you are hosting your website from home, is essentially simulating the process that a hacker would use to attack your site. “Attacking” your own site in this way is a powerful way of identifying security vulnerabilities.

How To Use Nmap



Nmap is straightforward to use, and most of the tools it provides are familiar to system admins from other programs. The advantage of Nmap is that it brings a wide range of these tools into one program, rather than forcing you to skip between separate and discrete network monitoring tools.

In order to use Nmap, you need to be familiar with command-line interfaces. Most advanced users are able to write scripts to automate common tasks, but this is not necessary for basic network monitoring.

How To Install Nmap

The process for installing Nmap is easy but varies according to your operating system. The Windows, Mac, and Linux versions of the program can be downloaded here.

- For Windows, Nmap comes with a custom installer (namp<version>setup.exe). Download and run this installer, and it automatically configures Nmap on your system.
- On Mac, Nmap also comes with a dedicated installer. Run the Nmap-<version>mpkg file to start this installer. On some recent versions of macOS, you might see a warning that Nmap is an "unidentified developer", but you can ignore this warning.
- Linux users can either compile Nmap from source or use their chosen package manager. To use apt, for instance, you can run Nmap –version to check if Nmap is installed, and sudo apt-get install Nmap to install it.

Nmap Tutorial and Examples

Once you've installed Nmap, the best way of learning how to use it is to perform some basic network scans.

How To Run a Ping Scan

One of the most basic functions of Nmap is to identify active hosts on your network. Nmap does this by using a ping scan. This identifies all of the IP addresses that are currently online without sending any packers to these hosts.

To run a ping scan, run the following command:

1. # nmap -sp 192.100.1.1/24

This command then returns a list of hosts on your network and the total number of assigned IP addresses. If you spot any hosts or IP addresses on this list that you cannot account for, you can then run further commands (see below) to investigate them further.

How To Run A Host Scan

A more powerful way to scan your networks is to use Nmap to perform a host scan. Unlike a ping scan, a host scan actively sends ARP request packets to all the hosts connected to your network. Each host then responds to this packet with another ARP packet containing its status and MAC address.

To run a host scan, use the following command:

1. # nmap -sp<target IP range>

This returns information on every host, their latency, their MAC address, and also any description associated with this address. This can be a powerful way of spotting suspicious hosts connected to your network.

If you see anything unusual in this list, you can then run a DNS query on a specific host, by using:

1. # namp -sL<IP address>

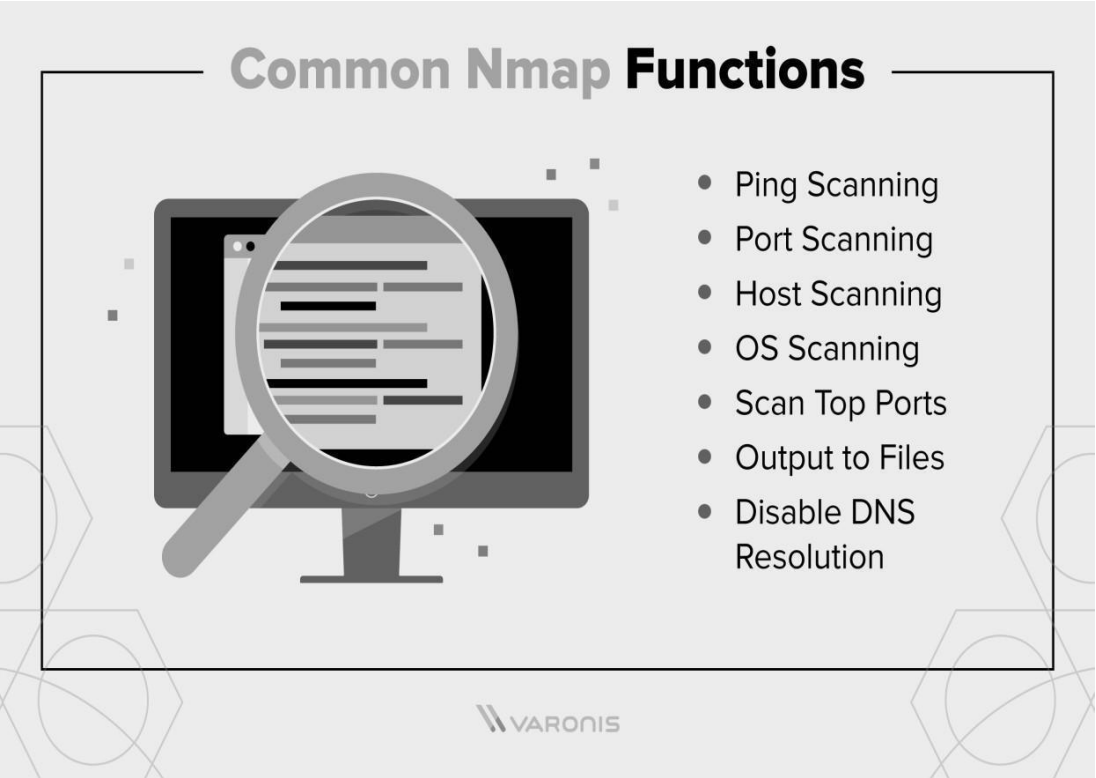
This returns a list of names associated with the scanned IP. This description provides information on what the IP is actually for.

How To Use Nmap in Kali Linux

Using Nmap in Kali Linux can be done in an identical way to running the program on any other flavor of Linux.

That said, there are advantages to using Kali when running Nmap scans. Most modern distros of Kali now come with a fully-features Nmap suite, which includes an advanced GUI and results viewer (Zenmap), a flexible data transfer, redirection, and debugging tool (Ncat), a utility for comparing scan results (Ndiff), and a packet generation and response analysis tool (Nping).

Nmap Commands



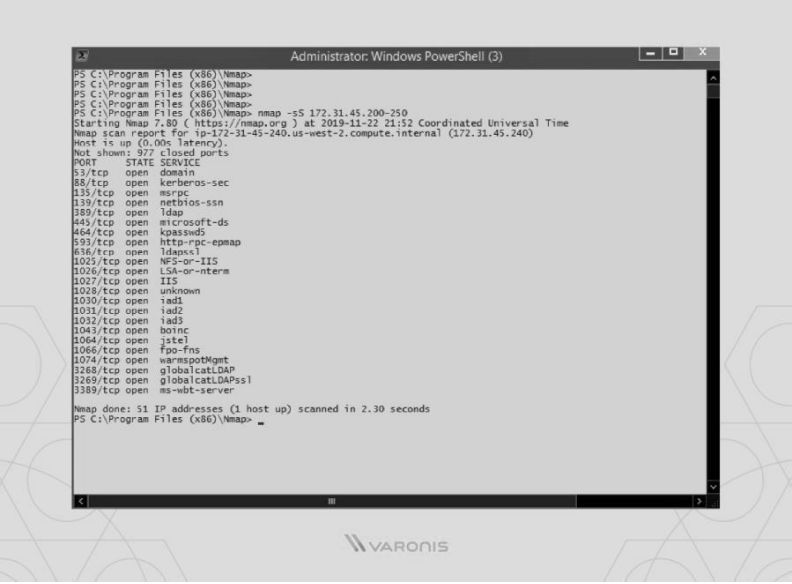
Most of the common functions of Nmap can be executed using a single command, and the program also uses a number of 'shortcut' commands that can be used to automate common tasks.

Here is a quick run-down:

1. Ping Scanning

As mentioned above, a ping scan returns information on every active IP on your network. You can execute a ping scan using this command:

2. Port Scanning



There are several ways to execute port scanning using Nmap. The most commonly used are these:

1. # sS TCP SYN scan 2.
3. # sT TCP connect scan 4.
5. # sU UDP scans 6.
7. # sY SCTP INIT scan
8.
9. # sN TCP NULL

The major differences between these types of scans are whether they cover TCP or UDP ports and whether they execute a TCP connection. Here are the basic differences:

- The most basic of these scans is the sS TCP SYN scan, and this gives most users all the information they need. It scans thousands of ports per second, and because it doesn't complete a TCP connection it does not arouse suspicion.
- The main alternative to this type of scan is the TCP Connect scan, which actively queries each host, and requests a response. This type of scan takes longer than a SYN scan, but can return more reliable information.
- The UDP scan works in a similar way to the TCP connect scan but uses UDP packets to scan DNS, SNMP, and DHCP ports. These are the ports most frequently targeted by hackers, and so this type of scan is a useful tool for checking for vulnerabilities.
- The SCTP INIT scan covers a different set of services: SS7 and SIGTRAN. This type of scan can also be used to avoid suspicion when scanning an external network because it doesn't complete the full SCTP process.
- The TOP NULL scan is also a very crafty scanning technique. It uses a loophole in the TCP system that can reveal the status of ports without directly querying them, which means that you can see their status even where they are protected by a firewall.

3. Host Scanning

Host scanning returns more detailed information on a particular host or a range of IP addresses. As mentioned above, you can perform a host scan using the following command:

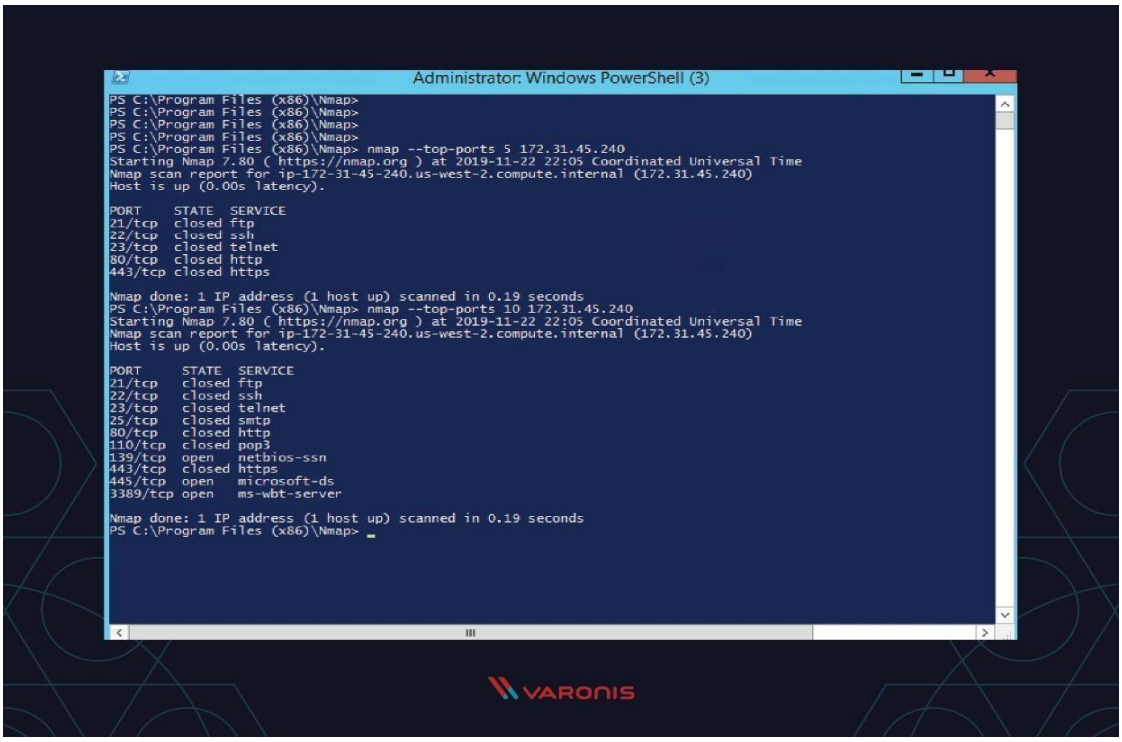
1. # nmap -sp<target IP range>

4. OS Scanning

OS scanning is one of the most powerful features of Nmap. When using this type of scan, Nmap sends TCP and UDP packets to a particular port, and then analyze its response. It compares this response to a database of 2600 operating systems, and return information on the OS (and version) of a host.
To run an OS scan, use the following command:

1. nmap -O <target IP>

5. Scan The Most Popular Ports



If you are running Nmap on a home server, this command is very useful. It automatically scans a number of the most 'popular' ports for a host. You can run this command using:

1. nmtop-ports 20 192.168.1.106

Replace the "20" with the number of ports to scan, and Nmap quickly scans that many ports. It returns a concise output that details the status of the most common ports, and this lets you quickly see whether you have any unnecessarily open ports.

6. Output to a File

If you want to output the results of your Nmap scans to a file, you can add an extension to your commands to do that. Simply add:

1. -oN output.txt

To your command to output the results to a text file, or:

1. -oX output.xml To
output to an XML.

7. Disable DNS Name Resolution

Finally, you can speed up your Nmap scans by using the -n parameter to disable reverse DNS resolution. This can be extremely useful if you want to scan a large network. For example, to turn off DNS resolution for the basic ping scan mentioned above, add -n:

1. # nmap -sp -n 192.100.1.1/24

Nmap FAQ

The commands above cover most of the basic functionality of Nmap. You might still have some questions though, so let's run through the most common ones.

Q: What Are Some Nmap Alternatives?

There are some alternatives to Nmap, but most of them are focused on providing specific, niche functionality that the average system administrator does need frequently. MASSCAN, for instance, is much faster than Nmap but provides less detail. Umit, by contrast, allows you to run several scans at once.
In reality, however, Nmap provides all the functionality and speed that the average user requires, especially when used alongside other similarly popular tools like NetCat (which can be used to manage and control network traffic) and ZenMap (which provides a GUI for Nmap)

Q: How Does Nmap Work?

Nmap builds on previous network auditing tools to provide quick, detailed scans of network traffic. It works by using IP packets to identify the hosts and IPs active on a network and then analyze these packets to provide information on each host and IP, as well as the operating systems they are running.

Q: Is Nmap Legal?

Yes. If used properly, Nmap helps protect your network from hackers, because it allows you to quickly spot any security vulnerabilities in your systems.

Whether port scanning on external servers is legal is another issue. The legislation in this area is complex and varies by territory. Using Nmap to scan external ports can lead to you being banned by your ISP, so make sure you research the legal implications of using the program before you start using it more widely.

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Is it possible to use nmap without root access? If yes, then how?	CO5	Understand
2	What are some of the most common ways that people use Nmap?	CO5	Understand
3	Do you know what NSE stands for?	CO5	Understand
4	What kind of information can be collected with Nmap?	CO5	Understand
5	Why do we need a tool like Nmap when there are other tools available?	CO5	Understand
6	What's the difference between TCP connect scanning and SYN Stealth Scanning? Which one would you recommend using in certain situations?	CO5	Understand
7	What are the various types of port states? Which one represents an open port?	CO5	Understand
8	What is OS fingerprinting? How is it done?	CO5	Understand
9	Should you always trust a host discovery scan? Explain your answer.	CO5	Understand
10	What is the best way to detect multiple hosts on a single subnet?	CO5	Understand
11	What is Nmap?	CO5	Understand
12	Can you explain how Nmap works?	CO5	Understand
13	How does the discovery phase work in Nmap?	CO5	Understand
14	Is it possible to use nmap without root access? If yes, then how?	CO5	Understand
15	What are some of the most common ways that people use Nmap?	CO5	Understand
16	Do you know what NSE stands for?	CO5	Understand
17	What kind of information can be collected with Nmap?	CO5	Understand
18	Why do we need a tool like Nmap when there are other tools available?	CO5	Understand
19	What's the difference between TCP connect scanning and SYN Stealth Scanning? Which one would you recommend using in certain situations?	CO5	Understand
20	What are the various types of port states? Which one represents an open port?	CO5	Understand
21	Compare the advantages of using OSPF over RIP in a Cisco network.	CO5	Analyzing

22	Identify the cause of a “down” interface status on a Cisco switch port.	CO5	Analyzing
23	Evaluate the security implications of enabling Telnet vs SSH on Cisco devices.	CO5	Evaluating
24	Assess the effectiveness of using VLANs to segment network traffic.	CO5	Evaluating
25	How Do I Modify Wireshark Packets On The Fly?	CO5	Understand
26	Why Don’t The Packets I’m Capturing Have Vlan Tags?	CO5	Understand
27	Critique the use of static routing versus dynamic routing protocols in different network scenarios.	CO5	Evaluating
28	Design a small Cisco network topology for a branch office with two VLANs and inter-VLAN routing.	CO5	Creating
29	Propose a security policy using Cisco features to protect a network from unauthorized access.	CO5	Creating
30	Develop a configuration script for Cisco routers to implement OSPF routing.	CO5	Creating

EXPERIMENT - 12

11. **Operating System Detection using Nmap**

OS Detection

One of Nmap's best-known features is remote OS detection using TCP/IP stack fingerprinting. Nmap sends a series of TCP and UDP packets to the remote host and examines practically every bit in the responses. After performing dozens of tests such as TCP ISN sampling, TCP options support and ordering, IP ID sampling, and the initial window size check, Nmap compares the results to its `nmap-os-db` database of more than 2,600 known OS fingerprints and prints out the OS details if there is a match. Each fingerprint includes a freeform textual description of the OS, and a classification which provides the vendor name (e.g. Sun), underlying OS (e.g. Solaris), OS generation (e.g. 10), and device type (general purpose, router, switch, game console, etc). Most fingerprints also have a Common Platform Enumeration (CPE) representation, like `cpe:/o:linux:linux_kernel:2.6`.

If Nmap is unable to guess the OS of a machine, and conditions are good (e.g. at least one open port and one closed port were found), Nmap will provide a URL you can use to submit the fingerprint if you know (for sure) the OS running on the machine. By doing this you contribute to the pool of operating systems known to Nmap and thus it will be more accurate for everyone.

OS detection enables some other tests which make use of information that is gathered during the process anyway. One of these is TCP Sequence Predictability Classification. This measures approximately how hard it is to establish a forged TCP connection against the remote host. It is useful for exploiting source-IP based trust relationships (rlogin, firewall filters, etc) or for hiding the source of an attack. This sort of spoofing is rarely performed any more, but many machines are still vulnerable to it. The actual difficulty number is based on statistical sampling and may fluctuate. It is generally better to use the English classification such as “worthy challenge” or “trivial joke”. This is only reported in normal output in verbose (`-v`) mode. When verbose mode is enabled along with `-O`, IP ID sequence generation is also reported. Most machines are in the “incremental” class, which means that they increment the ID field in the IP header for each packet they send. This makes them vulnerable to several advanced information gathering and spoofing attacks.

Another bit of extra information enabled by OS detection is a guess at a target's uptime. This uses the TCP timestamp option (RFC 1323) to guess when a machine was last rebooted. The guess can be inaccurate due to the timestamp counter not being initialized to zero or the counter overflowing and wrapping around, so it is printed only in verbose mode.

OS detection is enabled and controlled with the following options:

`-O` (Enable OS detection)

Enables OS detection, as discussed above. Alternatively, you can use `-A` to enable OS detection along with other things.

`--osscan-limit` (Limit OS detection to promising targets)

OS detection is far more effective if at least one open and one closed TCP port are found. Set this option and Nmap will not even try OS detection against hosts that do not meet this criteria. This can save substantial time, particularly on `-Pn` scans against many hosts. It only matters when OS detection is requested with `-O` or `-A`.

`--osscan-guess`; `--fuzzy` (Guess OS detection results)

When Nmap is unable to detect a perfect OS match, it sometimes offers up near-matches as possibilities. The match has to be very close for Nmap to do this by default. Either of these (equivalent) options make Nmap guess more aggressively. Nmap will still tell you when an imperfect match is printed and display its confidence level (percentage) for each guess.

`--max-os-tries` (Set the maximum number of OS detection tries against a target)

When Nmap performs OS detection against a target and fails to find a perfect match, it usually repeats the attempt. By default, Nmap tries five times if conditions are favorable for OS fingerprint submission, and twice when conditions aren't so good. Specifying a lower `--max-os-tries` value (such as 1) speeds Nmap up, though you miss out on retries which could potentially identify the OS. Alternatively, a high value may be set to allow even more retries when conditions are favorable. This is rarely done, except to generate better fingerprints for submission and integration into the Nmap OS database.

EXPERIMEN

T - 13

Do the following using NS2Simulator

i. Simulate to Find the Number of Packets Dropped

ii. Simulate to Find the Number of Packets Dropped by TCP/UDP

iii. Simulate to Find the Number of Packets Dropped due to Congestion

iv. Simulate to Compare Data Rate& Throughput.

v. Simulate to Plot Congestion for Different Source/Destination

vi. Simulate to Determine the Performance with respect to Transmission of Packets

SIMULATION USING NS-2

i)Introduction to NS-2:

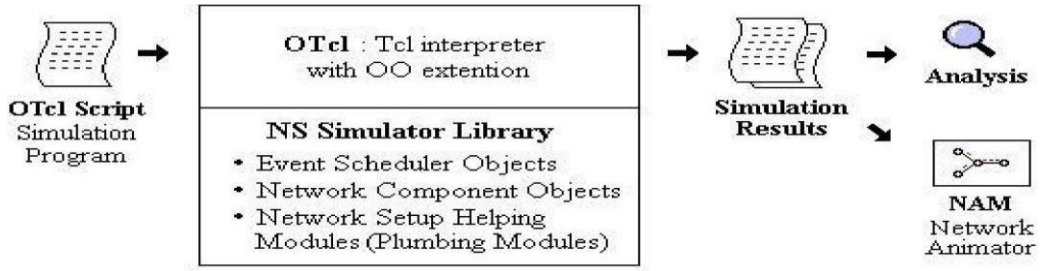
NS2 is an open-source simulation tool that runs on Linux. It is a discreet event simulator targeted at networking research and provides substantial support for simulation of routing, multicast protocols and IP protocols, such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks.

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



TCL – Tool Command Language

Tcl is a very simple programming language. If you have programmed before, you can learn enough to write interesting Tcl programs within a few hours. This page provides a quick overview of the main features of Tcl. After reading this you'll probably be able to start writing simple Tcl scripts on your own; however, we recommend that you consult one of the many available Tcl books for more complete information.

Basic syntax

Tcl scripts are made up of *commands* separated by newlines or semicolons.
Commands all have the same basic form illustrated by the following example: expr
20 + 10
This command computes the sum of 20 and 10 and returns the result, 30. You can try out this example and all the others in this page by typing them to a Tcl application such as tclsh; after a command completes, tclsh prints its result.
Each Tcl command consists of one or more *words* separated by spaces. In this example there are four words: expr, 20, +, and 10. The first word is the name of a command and the other words are *arguments* to that command. All Tcl commands consist of words, but different commands treat their arguments differently. The expr command treats all of its arguments together as an arithmetic expression, computes the result of that expression, and returns the result as a string. In the expr command the division into words isn't significant: you could just as easily have invoked the same command as
expr 20+10

However, for most commands the word structure is important, with each word used for a distinct purpose.
All Tcl commands return results. If a command has no meaningful result then it returns an empty string as its result.

Variables

Tcl allows you to store values in variables and use the values later in commands. The set command is used to write and read variables. For example, the following command modifies the variable x to hold the value 32: set x 32
The command returns the new value of the variable. You can read the value of a variable by invoking set with only a single argument: set x
You don't need to declare variables in Tcl: a variable is created automatically the
first time it is set. Tcl variables don't have types: any variable can hold any value.
To use the value of a variable in a command, use *variable substitution* as in the following example:
expr $x*3

When a $ appears in a command, Tcl treats the letters and digits following it as a variable name, and substitutes the value of the variable in place of the name. In this example, the actual argument received by the expr command will be 32*3 (assuming that variable x was set as in the previous example). You can use variable substitution in any word of any command, or even multiple times within a word:

```
set cmd expr set
x 11
$cmd $x*$x
```

Command substitution

You can also use the result of one command in an argument to another command.
This is called *command substitution*:

```
set a 44
set b [expr $a*4]
```

When a [appears in a command, Tcl treats everything between it and the matching] as a nested Tcl command. Tcl evaluates the nested command and substitutes its result into the enclosing command in place of the bracketed text. In the example above the second argument of the second set command will be 176.

Quotes and braces

Double-quotes allow you to specify words that contain spaces. For example, consider the following script:

```
set x 24
set y 18
set z "$x + $y is [expr $x + $y]"
```

After these three commands are evaluated variable z will have the value 24 + 18 is 42. Everything between the quotes is passed to the set command as a single word. Note that (a) command and variable substitutions are performed on the text between the quotes, and (b) the quotes themselves are not passed to the command. If the quotes were not present, the set command would have received 6 arguments, which would have caused an error.
Curly braces provide another way of grouping information into words. They are different from quotes in that no substitutions are performed on the text between the curly braces:

```
set z {$x + $y is [expr $x + $y]}
```

This command sets variable z to the value "$x + $y is [expr $x + $y]".

Control structures

Tcl provides a complete set of control structures including commands for conditional execution, looping, and procedures. Tcl control structures are just commands that take Tcl scripts as arguments. The example below creates a Tcl procedure called power, which raises a base to an integer power:

```
proc power {base p} { set
result 1
while {$p > 0} {
set result [expr $result * $base] set
p [expr $p - 1]
}return $result}
```

This script consists of a single command, proc. The proc command takes three arguments: the name of a procedure, a list of argument names, and the body of the procedure, which is a Tcl script. Note that everything between the curly brace at the end of the first line and the curly brace on the last line is passed verbatim to proc as a single argument. The proc command creates a new

Tcl command named power that takes two arguments. You can then invoke power with commands like the following:

power 2 6

power 1.15 5

When power is invoked, the procedure body is evaluated. While the body is executing it can access its arguments as variables: base will hold the first argument and p will hold the second.

The body of the power procedure contains three Tcl commands: set, while, and return. The while command does most of the work of the procedure. It takes two arguments, an expression ($p > 0) and a body, which is another Tcl script. The while command evaluates its expression argument using rules similar to those of the C programming language and if the result is true (nonzero) then it evaluates the body as a Tcl script. It repeats this process over and over until eventually the expression evaluates to false (zero). In this case the body of the while command multiplied the result value by base and then decrements p. When p reaches zero the result contains the desired power of base. The return command causes the procedure to exit with the value of variable result as the procedure's result.

Where do commands come from?

As you have seen, all of the interesting features in Tcl are represented by commands. Statements are commands, expressions are evaluated by executing commands, control structures are commands, and procedures are commands.

Tcl commands are created in three ways. One group of commands is provided by the Tcl interpreter itself. These commands are called *builtin commands*. They include all of the commands you have seen so far and many more (see below). The builtin commands are present in all Tcl applications.

The second group of commands is created using the Tcl extension mechanism. Tcl provides APIs that allow you to create a new command by writing a *command procedure* in C or C++ that implements the command. You then register the command procedure with the Tcl interpreter by telling Tcl the name of the command that the procedure implements. In the future, whenever that particular name is used for a Tcl command, Tcl will call your command procedure to execute the command. The builtin commands are also implemented using this same extension mechanism; their command procedures are simply part of the Tcl library.

When Tcl is used inside an application, the application incorporates its key features into Tcl using the extension mechanism. Thus the set of available Tcl commands varies from application to application. There are also numerous extension packages that can be incorporated into any Tcl application. One of the best known extensions is Tk, which provides powerful facilities for building graphical user interfaces. Other extensions provide object-oriented programming, database access, more graphical capabilities, and a variety of other features. One of Tcl's greatest advantages for building integration applications is the ease with which it can be extended to incorporate new features or communicate with other resources.

The third group of commands consists of procedures created with the proc command, such as the power command created above. Typically, extensions are used for lower-level functions where C programming is convenient, and procedures are used for higher-level functions where it is easier to write in Tcl.

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

NS Simulator Preliminaries.

Initialization and termination aspects of the ns simulator.
Definition of network nodes, links, queues and topology.
Definition of agents and of applications.
The nam visualization tool.
Tracing and random variables.

Features of NS2

NS2 can be employed in most unix systems and windows. Most of the NS2 code is in C++. It uses TCL as its scripting language, Otcl adds object orientation to TCL.NS(version 2) is an object oriented, discrete event driven network simulator that is freely distributed and open source.

• Traffic Models: CBR, VBR, Web etc

• Protocols: TCP, UDP, HTTP, Routing algorithms,MACetc

• Error Models: Uniform, burstyetc

• Misc: Radio propagation, Mobility models , Energy Models

• Topology Generation tools

• Visualization tools (NAM), Tracing

Structure of NS

● NS is an object oriented discrete event simulator
– Simulator maintains list of events and executes one event after another
– Single thread of control: no locking or race conditions
● Back end is C++ event scheduler

– Protocols mostly
– Fast to run, more control
• Front end is OTCL
Creating scenarios, extensions to C++ protocols
fast to write and change

Platforms

It can be employed in most unixsystems(FreeBSD, Linux, Solaris) and Windows.

Source code

Most of NS2 code is in C++

Scripting language

It uses TCL as its scripting language OTcl adds object orientation to TCL.

Protocols implemented in NS2

Transport layer(Traffic Agent) – TCP, UDP Network
layer(Routing agent)
Interface queue – FIFO queue, Drop Tail queue, Priority queue
Logic link contol layer – IEEE 802.2, AR

How to use NS2 Design Simulation – Determine simulation scenario Build ns-2 script using tcl.
Run simulation

Simulation with NS2

Define objects of simulation. Connect
the objects to each other
Start the source applications. Packets are then created and are transmitted through network. Exit the simulator after a certain fixed time.

NS programming Structure

● Create the event scheduler

● Turn on tracing

● Create network topology

● Create transport connections

● Generate traffic

● Insert errors

ii& iii) Simulate a three-node point-to-point network with a duplex link between them. Set the queue size and vary the bandwidth and find the number of packets dropped.

STEPS:

Step1: Select the hub icon on the toolbar and drag it onto the working window.
Step2: Select the host icon on the toolbar and drag it onto the working window. Repeat this for another host icon.
Step3: Select the link icon on the toolbar and drag it on the screen from host (node 1) to the hub and again from host(node 2) to the hub. Here the hub acts as node 3 in the point-to-point network. This leads to the creation of the 3-node point-to-point network topology. Save this topology as a .tpl file.
Step4: Double-click on host(node 1), a host dialog box will open up. Click on Node editor and you can see the different layers- interface, ARP, FIFO, MAC, TCPDUMP, Physical layers. Select MAC and then select full-duplex for switches and routers and half duplex for hubs, and in log Statistics, select Number of Drop Packets, Number of Collisions, Throughput of incoming packets and Throughput of outgoing packets. Select FIFO and set the queue size to 50 and press OK. Then click on Add. Another dialog box pops up. Click on the Command box and type the Command according to the following syntax:
stg [-t duration(sec)] [-p port number]HostIPaddr
and click OK.
Step 5: Double-click on host (node 2), and follow the same step as above with only change in command according to the following syntax:
rtg [-t] [-w log] [-p port number]
and click OK.
Step 6: Double click on the link between node 1 and the hub to set the bandwidth to some initial value say, 10 Mbps. Repeat the same for the other node.
Step 7: Click on the E button (Edit Property) present on the toolbar in order to save the changes made to the topology. Now click on the R button (RunSimulation). By doing so a user can run/pause/continue/stop/abort/disconnect/reconnect/submit a simulation. No simulation settings can be changed in this mode.

Step 8: Now go to Menu->Simulation->Run. Executing this command will submit he current simulation job to one available simulation server managed by the dispatcher. When the simulation server is executing, the user will see the time knot at the bottom of the screen move. The time knot reflects the current virtual time (progress) of the simulation case.

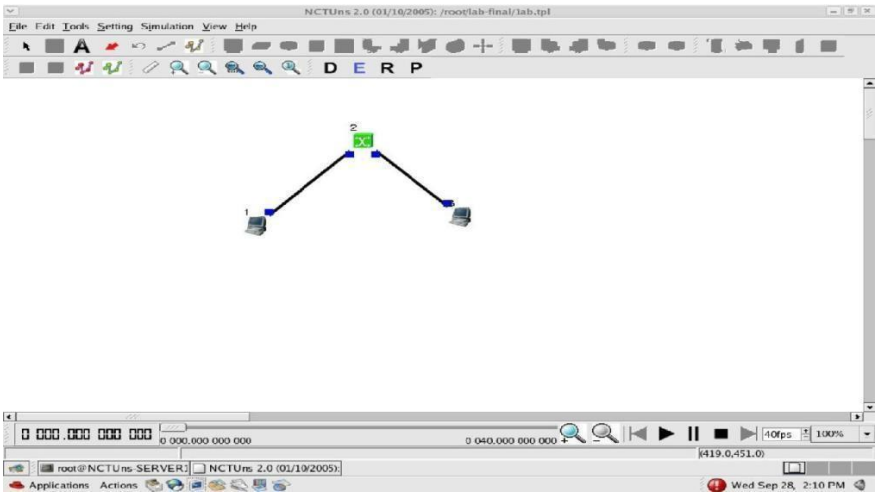
Step 9: To start the playback, the user can left-click the start icon(|>) of the time bar located at the bottom. The animation player will then start playing the recorded packet animation.

Step 10: Change the bandwidth say, 9 Mbps, and run the simulation and compare the two results.

Step 11: To view the results, go to the filename. results folder**.**

Note: To get the syntax of any command, double click on the host icon. Host dialog boxes appear and then choose App. Usage.

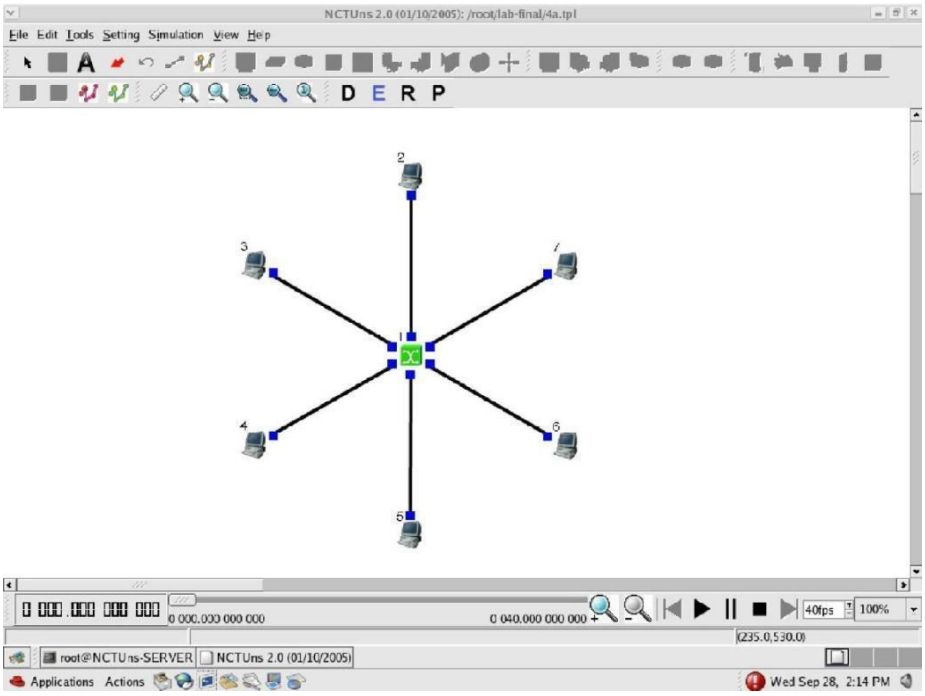
The screenshot below explain the topology.



iv)Simulate the transmission of ping messages over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

STEPS:

Step 1: Click on the subnet icon on the toolbar and then click on the screen of the working window.
Step 2: Select the required number of hosts and a suitable radius between the host and the switch.
Step 3: In the edit mode, get the IP address of one of the hosts say, host 1 and then for the other host say, host2 set the drop packet and no: of collisions statistics
as described in the earlier experiments.
Step 4: Now run the simulation.
Step 5: Now click on any one of the hosts and click on command console and ping the destination node.
ping IP Address of the host
Note: The no: of drop packets are obtained only when the traffic is more in the network. For checking the no of packets dropped press ctrl+C
The screenshot of the topology is shown below:

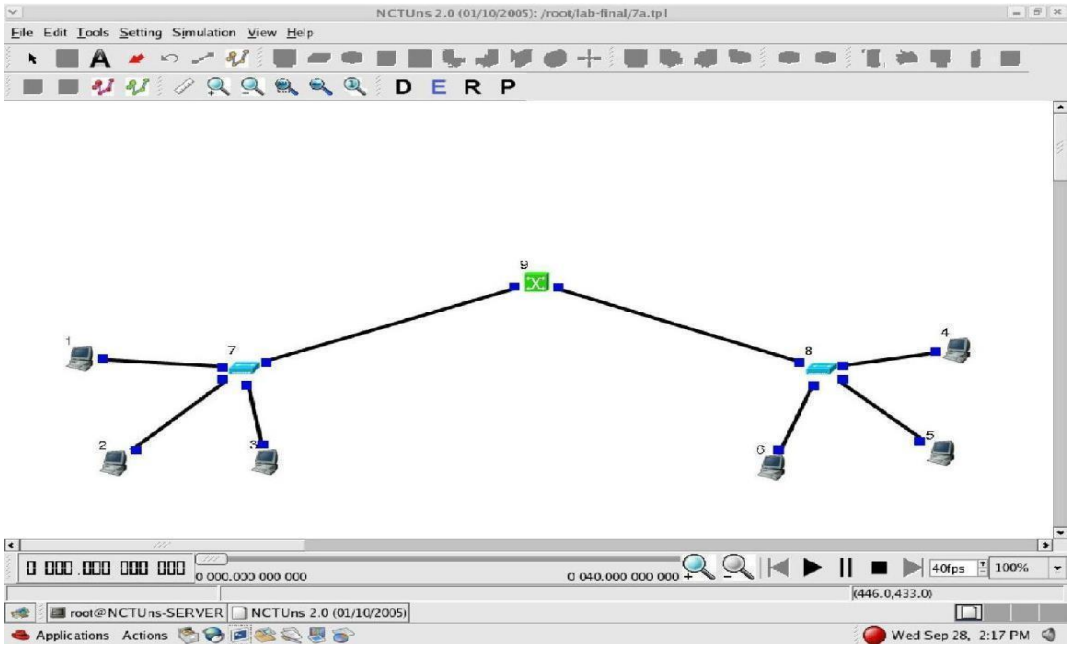


vi) Simulate an Ethernet LAN using N nodes and set multiple traffic nodes and plot congestion window for different source/destination.

STEPS:

Step 1: Connect one set of hosts with a hub and another set of hosts also through a hub and connect these two hubs through a switch. This forms an Ethernet LAN.
Step 2: Setup multiple traffic connections between the hosts on one hub and hosts on another hub using the following command:
stcp [-p port] [-l writesize] hostIPaddrrtcp [-p port] [-l readsize]
Step 3: Setup the collision log at the destination hosts in the MAC layer as described in the earlier experiments.
Step 4: To plot the congestion window go to Menu->Tools->Plot Graph->File->open->filename.results->filename.coll.log
Step 5: View the results in the filename.results.

The screenshot of the topology is shown below:



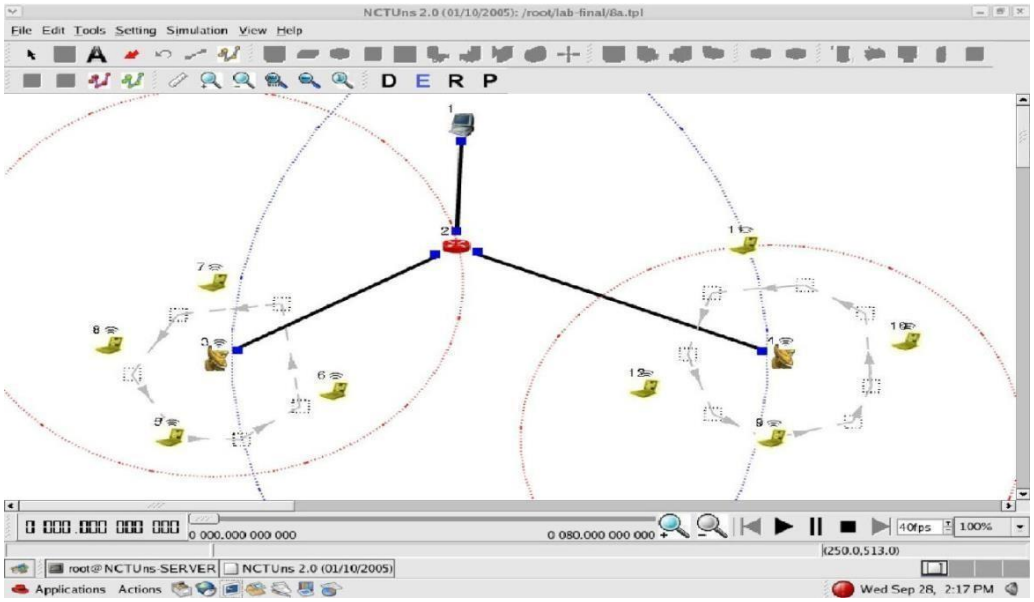
vii) Simulate simple ESS and with transmitting nodes in wireless LAN by simulation and determine the performance with respect to transmission of packets.

STEPS:

Step 1: Connect a host and two WLAN access points to a router.
Step 2: Setup multiple mobile nodes around the two WLAN access points and set the path for each mobile node.
Step 3: Setup a ttcp connection between the mobile nodes and host using the following command:
Mobile Host 1

ttcp –t –u –s –p 3000 IPAddrOf Receiver
Mobile Host 1
ttcp –t –u –s –p 4000 IPAddrOf Receiver
Host(Receiver)
ttcp –r –u –s –p 3000 ttcp –r –u –s –p 4000
Step 4: Setup the input throughput log at the destination host.
Step 5: To set the transmission range go to Menu->Settings->WLAN mobile node->Show transmission range.
Step 5: View the results in the filename. results.

Screenshot:



VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	What are the various types of port states? Which one represents an open port?	CO5	Understand
2	A User Is Unable To Ping A System On The Network. How Can Wireshark Be Used To Solve The Problem?	CO5	Understand
3	What's the difference between host discovery scans and ping scans?	CO5	Understand
4	What is OS fingerprinting? How is it done?	CO5	Understand
5	What type of scan uses half-open connections to determine if a port is open or closed?	CO5	Understand
6	What advantages does Scapy have over Nmap?	CO5	Understand
7	What is the best way to detect remote operating systems running on remote hosts?	CO5	Understand
8	What is the purpose of a zombie host?	CO5	Understand
9	Should you always trust a host discovery scan? Explain your answer.	CO5	Understand
10	Name the factors that affect the reliability of the network?	CO5	Understand
11	In which cases will a Ping sweep fail?	CO5	Understand
12	What is a network simulator?	CO5	Remembering
13	Name two popular network simulators used in research.	CO5	Remembering
14	Define the term "packet" in networking.	CO5	Remembering
15	What does NS-3 stand for?	CO5	Remembering
16	Explain how a network simulator helps in analyzing network performance.	CO5	Understanding
17	Describe the difference between a wired and wireless network simulation.	CO5	Understanding
18	Why is simulation preferred over real-world testing in network research?	CO5	Understanding
19	How would you set up a simple topology in a network simulator like NS-2 or NS-3?	CO5	Applying
20	Given a network scenario, how would you use a simulator to measure throughput?	CO5	Applying
21	Apply a routing protocol in a network simulator and describe the process.	CO5	Applying
22	Analyze the effect of varying node mobility on packet delivery ratio in a wireless network simulation.	CO5	Analyzing

23	Compare the results of two different routing protocols simulated on the same network topology.	CO5	Analyzing
24	Identify bottlenecks in a simulated network using throughput and delay statistics.	CO5	Analyzing
25	Evaluate the accuracy of simulation results compared to real-world network measurements.	CO5	Evaluating
26	Critique the limitations of using network simulators for large-scale network testing.	CO5	Evaluating
27	Assess the benefits and drawbacks of discrete event simulation in network modeling.	CO5	Evaluating
28	Design a new experiment using a network simulator to test the impact of congestion control algorithms.	CO5	Creating
29	Propose enhancements to a network simulator to better model 5G network scenarios.	CO5	Creating
30	Develop a custom script in a network simulator to simulate IoT device communication.	CO5	Creating

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VIVA QUESTIONS

1. How does Nmap perform host discovery?

Nmap offers a variety of methods to perform host discovery. Some common techniques include:

ICMP Echo Request (ping): Sends ICMP echo requests to target hosts and waits for replies ARP: Sends ARP requests to the local network and listens for ARP replies
SYN/ACK packets: Sends SYN or ACK packets and analyzes the responses
UDP: Sends UDP packets and checks for responses or an ICMP "destination unreachable" error

2. What is Nmap and what is it used for?

Nmap, or Network Mapper, is an open-source security tool that helps IT professionals discover hosts, services, and vulnerabilities on a network. By using Nmap, administrators can stay informed about security risks within their networks and take timely action to safeguard the organization's assets.

3. Explain the primary functions of Nmap. Some of the primary functions of Nmap include:

Host discovery: Finding active hosts in the network
Port scanning: Identifying open and closed ports on target hosts
Version detection: Determining the service and application versions running on target hosts OS detection: Identifying the operating systems of target hosts
Scriptable interaction: Running custom scripts using the Nmap Scripting Engine (NSE]

4. What are some common Nmap scan techniques?

Popular Nmap scan techniques include:

SYN scan: Sends SYN packets and waits for SYN/ACK or RST responses
Connect scan: Uses the standard TCP three-way handshake process to scan the target ACK scan: Utilizes ACK packets to determine if a port is filtered or unfiltered
UDP scan: Sends UDP packets to the target host and checks for responses or ICMP errors SCTP INIT scan: Sends SCTP INIT packets to target hosts and awaits for INIT-ACK responses

5. What is the Nmap Scripting Engine (NSE)?

The Nmap Scripting Engine (NSE) allows users to write and execute custom scripts to enhance Nmap's functionality. These custom scripts can be used for various purposes such as vulnerability detection, network discovery, and more, significantly extending Nmap's capabilities.

6. Do you know what NSE stands for?

NSE stands for Nmap Scripting Engine. The NSE is a powerful engine that allows users to extend the functionality of Nmap by writing their own scripts. These scripts can be used to perform a variety of tasks, such as network discovery, port scanning, and vulnerability analysis.

7. What kind of information can be collected with Nmap?

Nmap can be used to collect a variety of information about a target network or system. This information can include things like the network layout, the types of devices and services that are running, and the open ports and vulnerabilities that are present. Nmap can also be used to perform more sophisticated attacks, like denial of service attacks or password guessing.

8. Why do we need a tool like Nmap when there are other tools available?

Nmap is a powerful tool that can be used for a variety of tasks, including network exploration, security auditing, and network troubleshooting. It is unique in its ability to scan large networks quickly and efficiently. Additionally, Nmap can be used to identify hosts and services on a network, as well as to determine which ports are open on a given host.

9. What's the difference between TCP connect scanning and SYN Stealth Scanning? Which one would you recommend using in certain situations?
TCP connect scanning is the most basic form of port scanning, and simply tries to establish a connection with the target host on the specified port. If the connection is successful, then the port is considered open. SYN Stealth Scanning is a more advanced form of port scanning that uses a SYN packet to initiate the connection. If the target host responds with a SYN/ACK packet, then the port is considered open. If the target host responds with a RST packet, then the port is considered closed. In general, SYN Stealth Scanning is a more reliable form of port scanning, and is recommended for most situations.

10. What are the various types of port states? Which one represents an open port?

There are four types of port states: open, closed, filtered, and unfiltered. An open port is one that is ready and willing to accept connections. A closed port is one that is not accepting connections. A filtered port is one that is being blocked by a firewall. An unfiltered port is one that cannot be reached for some reason.

11. What's the difference between host discovery scans and ping scans?
A host discovery scan is used to find out which hosts are up and running on a network, while a ping scan is used to check if a host is responsive.

12. What type of scan uses half-open connections to determine if a port is open or closed?
A SYN scan uses half-open connections to determine if a port is open or closed. This type of scan is also known as a "half-open" or "stealth" scan.

13. What advantages does Scapy have over Nmap?
Scapy is a much more powerful and flexible tool than Nmap. It can be used for a wider range of tasks, including network discovery, scanning, tracerouting, and even attacks. It is also easier to use and customize than Nmap.

14. What is the purpose of a zombie host?

A zombie host is a computer that has been infected with a malware that allows it to be controlled remotely by a hacker. Hackers can use zombie hosts to launch attacks on other computers or networks, or to steal sensitive information.

15. In which cases will a Ping sweep fail?
A Ping sweep will fail if the target host is not online, if it is behind a firewall that is blocking ICMP traffic, or if the network is configured to not respond to ICMP requests.

16. What is the best way to detect multiple hosts on a single subnet?
The best way to detect multiple hosts on a single subnet is to use a tool like Nmap. Nmap can quickly scan a subnet and return a list of all active hosts. This is a very useful tool for network administrators who need to keep track of all devices on a network.

17. What is the best way to detect remote operating systems running on remote hosts?

The best way to detect remote operating systems running on remote hosts is to use Nmap. Nmap is a network exploration and security auditing tool that can be used to identify hosts and services on a network, as well as to determine what operating systems those hosts are running. By running Nmap

against a remote host, you can fingerprint the operating system that host is running and determine what type of system it is.

18. What is the best way to find out unused IP addresses on networks?
The best way to find unused IP addresses on a network is to use Nmap to scan for open ports. If there are no open ports, then the IP address is likely unused.

19. What is OS fingerprinting? How is it done?
OS fingerprinting is a process of identifying what operating system is running on a given host, based on analyzing the host's responses to various network probes. This can be done manually, by looking at the responses and trying to identify patterns, or automatically, by using a tool like Nmap that can compare the responses to a database of known operating systems.

20. Should you always trust a host discovery scan? Explain your answer.
No, you should not always trust a host discovery scan. The results of a host discovery scan can be spoofed, which means that the host may not actually be where it says it is. Additionally, some hosts may be configured to not respond to certain types of host discovery scans, which means that they may not show up in the scan result