KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

COMPUTER NETWORKS LABORATORY

Subject Code :KG23ACM310

Regulation : KGR23

Academic Year : 2025-2026

III B. TECH I SEMESTER

COMPUTER SCIENCE AND ENGINEERING

KG REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

Affiliated o JNTUH, Chilkur,(V), Moinabad(M) R. R Dist, TS-501504

KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

VISION AND MISSION OF THE INSTITUTION

VISION:

To become an institution which is internationally recognized for its holistic approach to engineering, innovative teaching and learning culture, research and entrepreneurial ecosystem, and sustainable social impact in the community.

MISSION:

• To offer undergraduate and post-graduate programs which are supported through industry relevant curriculum and innovative teaching and learning processes that would help students succeed in their professional careers.
• To provide faculty and students with an ecosystem that fosters innovation, research, entrepreneurship, and international exposure through strategic partnerships with governmentorganizations and collaboration with industries.
• To provide holistic learning environment to students which will contribute to their personal and professional growth and enable them to become leaders in their respective fields.
• To contribute to the development of the region by using our technological expertise to work with nearby communities and support them in their social and economic development.

KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION:

To be recognized as a department of excellence by stimulating a learning environment in which students and faculty will thrive and grow to achieve their professional, institutional and societal goals.

MISSION:

• To provide high quality technical education to students that will enable life-long learning and build expertise in advanced technologies in Computer Science and Engineering.
• To promote research and development by providing opportunities to solve complex engineering problems in collaboration with industry and government agencies.
• To encourage professional development of students that will inculcate ethical values and leadership skills through entrepreneurship while working with the community to address societalissues.

PROGRAM EDUCATIONAL OBJECTIVES

PEO 1: Graduates will provide solutions to difficult and challenging issues in their profession by applying computer science and engineering theory and principles.
PEO 2: Graduates have successful careers in computer science and engineering fields or will be able to successfully pursue advanced degrees.
PEO 3: Graduates will communicate effectively, work collaboratively and exhibit high levels of professionalism, moral and ethical responsibility.
PEO 4: Graduates will develop the ability to understand and analyze engineering issues in a broader perspective with ethical responsibility towards sustainable development.

KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM OUTCOMES

PO I: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO II: 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.
PO III: 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.
PO IV: 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.
PO V: 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.
PO VI: 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.
PO VII: 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.
PO VIII: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO IX: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO X: 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.
PO XI: 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.
PO XII: 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.

KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM SPECIFIC OUTCOMES

PSO1: The Computer Science and Engineering graduates are able to analyze, design, develop, test and apply management principles, mathematical foundations in the development of intelligentsystems with computational solutions, make them to expert in designing the secure application and hardware prototype
PSO2: The graduating student will be analyze the contemporary research issues in different areasof computer science & engineering and explore research gaps, analyze and carry out research in the specialized/emerging areas.
PSO3: Develop their skills to solve problems in the broad area of programming concepts and appraise environmental and social issues with ethics and manage different projects in multi- disciplinary field to conducive in cultivating skills for successful career, entrepreneurship and higher studies.

KG REDDY College of Engineering & Technology
AN AUTONOMOUS INSTITUTION

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COMPUTER NETWORKS LABORATORY

Course Code:KG23ACM310 **L T P C**

B.Tech. III Year I Sem. **0 0 2 1**

Course Objectives: The objectives of this course for the students are to:

1. To understand the working principle of various communication protocols.
2. To understand the network simulator environment and visualize a network topology and observe its performance
3. To analyze the traffic flow and the contents of protocol frames
4. To analyze network traffic and monitor network performance
5. To simulate and analyze the performance of different network topologies

Course Outcomes:

1 . Implement data link layer farming methods. (K2)
2. Analyze error detection and error correction codes. (K3)
3. Apply and analyze routing and congestion issues in network design. (K5)
4. Demonstrate Encoding and Decoding techniques used in presentation layer. (K5)
5. Design and implement solutions using various network tools. (K6)

List of Experiments

S.no	Name of The Experiment
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, CRC-16 and CRC CCIP
3	Develop a simple data link layer that performs the flow control using the sliding window protocol, and loss recovery usi Go Back-N mechanism.
4	Implement Dijkstra'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	Wireshark i. Packet Capture Using Wire shark ii. Starting Wire shark iii. Viewing Captured Traffic iv. Analysis and Statistics & Filters.
11	How to run Nmap scan
12	Operating System Detection using Nmap Do the following using NS2 Simulator i.NS2 Simulator-Introduction ii Simulate to Find the Number of Packets Dropped iii. Simulate to Find the Number of Packets Dropped by TCP/UDP iv. Simulate to Find the Number of Packets Dropped due to Congestion

	v. Simulate to Compare Data Rate & Throughput. vi. Simulate to Plot Congestion for Different Source/Destination vii Simulate to Determine the Performance with respect to Transmission of Packets

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

REFERENCE BOOKS: 1. An Engineering Approach to Computer Networks, S. Keshav, 2nd Edition, Pearson Education. 2. Data Communications and Networking – Behrouz A. Forouzan. 3rd Edition, TMH

.

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

CODE :

//CHARACTER STUFFING

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<process.h>
void main(){
int i=0,j=0,n,pos;
char a[20],b[50],ch;
printf("enter string:\n");
scanf("%s",a);
n=strlen(a);
printf("enter position\n");
scanf("%d",&pos);
if(pos>n){
printf("invalid position,Enter again:");
scanf("%d",&pos);
}
printf("enter the character\n");
ch=getche();
b[0]='d';
b[1]='l';
b[2]='e';
b[3]='s';
b[4]='t';
b[5]='x';
j=6;
while(i<n){
if(i==pos-1){
b[j]='d';
b[j+1]='l';
b[j+2]='e';
b[j+3]=ch;
b[j+4]='d';
b[j+5]='l';
b[j+6]='e';
j=j+7;
}
if(a[i]=='d'&&a[i+1]=='l'&&a[i+2]=='e'){
b[j]='d';
b[j+1]='l';
b[j+2]='e';
j=j+3;
}
b[j]=a[i];
i++;
 j++;
 }
 b[j]='d';
```

```
b[j+1]='l';
b[j+2]='e';
b[j+3]='e';
b[j+4]='t';
 b[j+5]='x';
b[j+6]='\0';
printf("\nframe after stuffing:\n");
printf("%s",b);
getch();
}
```

OUTPUT:

```
enter string:
hello
enter position
3
enter the character
a
frame after stuffing:
dlestxhedleadlellodleetx
```

//BIT STUFFING

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main(){
int a[20],b[30],i,j,k,count,n;
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++;
```

```
count=1;
}
printf("After stuffing the frame is:");
for(i=0;i<j;i++)
printf("%d",b[i]);
getch();
}
```

OUTPUT:

```
enter frame length:9
enter input frame(0's&1's only):0
1
0
1
1
1
1
1
1
After stuffing the frame is:0101111101
```

EXPERIMENT 2: Write a program to compute CRC code for the polynomials CRC-12, CRC-16 and CRC CCIP

CODE:

```
#include<stdio.h>
#include<conio.h>
int main(void){
int data[50],div[16],rem[16];
int datalen,divlen,i,j,k;
int ch;
printf("Enter the data: ");
i=0;
while((ch=fgetc(stdin))!='\n'){
if(ch=='1')
data[i]=1;
else if(ch=='0')
data[i]=0;
else
continue;
i++;
}
datalen=i;
printf("Enter the divisor: ");
i=0;
while((ch=fgetc(stdin))!='\n'){
if(ch=='1')
div[i]=1;
else if(ch=='0')
    div[i]=0;
```

```
else
continue;
i++;
}
divlen=i;
for(i=datalen;i<datalen+divlen-1;i++)
data[i]=0;
for(i=0;i<divlen;i++)
rem[i]=data[i];
k=divlen;
while(k<=datalen){
if(rem[0]==1){
for(i=0;i<divlen;i++)
rem[i]=rem[i]^div[i];
}else{
for(i=0;i<divlen;i++)
rem[i]=rem[i];
}
for(i=0;i<divlen-1;i++)
rem[i]=rem[i+1];
if(k<datalen)
rem[i]=data[k];
k++;
}
j=1;
for(i=datalen;i<datalen+divlen-1;i++)
data[i]=rem[j++];
printf("\nThe data to be sent is:\n");
for(i=0;i<datalen+divlen-1;i++)
printf("%d",data[i]);
getch();
return 0;
}
```

OUTPUT:

Enter the data: 10101111
Enter the divisor: 1011

The data to be sent is:
10101111100

EXPERIMENT 3: Develop a simple data link layer that performs the flow control using the sliding window

CODE:

```
#include <stdio.h>

int main(){
   int w,f,i;
   printf("Enter window size:");
   scanf("%d",&w);
```

```
    printf("Enter number of frames to transmit:");
    scanf("%d",&f);
    int frames[f];
    printf("\nEnter %d frames:",f);
    for (i=0;i<f;i++) {
       scanf("%d",&frames[i]);
    }
    printf("\nWith 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 the sender waits for acknowledgement sent by the
  receiver.\n",w);
    for (i=0;i<f;i++) {
       printf("%d",frames[i]);
       if ((i+1)%w==0||i==f-1){
          printf("\nAcknowledgement of the above frames is received by sender.\n\n");
       }
    }
    return 0;
  }
```

OUTPUT:

```
Enter window size:3
Enter number of frames to transmit:5
Enter 5 frames:12
5
8
9
46
With sliding window protocol, the frames will be sent in the following manner (assuming no corruption
     of frames):

After sending 3 frames, at each stage the sender waits for acknowledgement sent by the receiver.
1258
Acknowledgement of the above frames is received by sender.

946
Acknowledgement of the above frames is received by sender.
```

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

CODE:

```
#include <stdio.h>
#define INF 9999
#define MAX 10

void dijkstra(int G[MAX][MAX], int n, int start) {
   int cost[MAX][MAX], dist[MAX], pred[MAX], visited[MAX] = {0};
   int i, j, count, mindist, nextnode;

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

```
            cost[i][j] = (G[i][j] == 0) ? INF : G[i][j];

    for (i = 0; i < n; i++) {
        dist[i] = cost[start][i];
        pred[i] = start;
    }

    dist[start] = 0; visited[start] = 1;

    for (count = 1; count < n - 1; count++) {
        mindist = INF;
        for (i = 0; i < n; i++)
            if (!visited[i] && dist[i] < mindist)
                mindist = dist[i], nextnode = i;

        visited[nextnode] = 1;
        for (i = 0; i < n; i++)
            if (!visited[i] && mindist + cost[nextnode][i] < dist[i])
                dist[i] = mindist + cost[nextnode][i], pred[i] = nextnode;
    }

    for (i = 0; i < n; i++)
        if (i != start) {
            printf("\nDistance to node %d = %d\nPath: %d", i, dist[i], i);
            for (j = i; j != start; j = pred[j])
                printf(" <- %d", pred[j]);
            printf("\n");
        }
}

int main() {
    int G[MAX][MAX], n, start;
    printf("Enter number of vertices: ");
    scanf("%d", &n);
    printf("Enter adjacency matrix (0 for no edge):\n");
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &G[i][j]);
    printf("Enter starting node (0 to %d): ", n - 1);
    scanf("%d", &start);
    dijkstra(G, n, start);
    return 0;
}
```

OUTPUT:

```
Enter number of vertices: 5
Enter adjacency matrix (0 for no edge):
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0
Enter starting node (0 to 4): 0
```

```
Distance to node 1 = 10
Path: 1 <- 0

Distance to node 2 = 50
Path: 2 <- 3 <- 0
Distance to node 3 = 30
Path: 3 <- 0
Distance to node 4 = 60
Path: 4 <- 2 <- 3 <- 0
```

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

```
#include<stdio.h>
int a[10][10],n;
void adj(int k);

int main(){
   int i,j,root;
   printf("Enter number of nodes:");
   scanf("%d",&n);
   printf("Enter adjacency matrix (0 or 1):\n");
   for(i=1;i<=n;i++){
      for(j=1;j<=n;j++){
         printf("Connection from %d to %d:",i,j);
         scanf("%d",&a[i][j]);
      }
   }
   printf("Enter root node:");
   scanf("%d",&root);
   adj(root);
   return 0;
}

void adj(int k){
   int j;
   printf("Adjacent nodes of root node %d are:\n",k);
   for(j=1;j<=n;j++){
      if(a[k][j]==1 || a[j][k]==1)
         printf("%d\t",j);
   }
   printf("\n");
}
```

OUTPUT:

```
Enter number of nodes:5
Enter adjacency matrix (0 or 1):
Connection from 1 to 1:1
Connection from 1 to 2:0
Connection from 1 to 3:1
Connection from 1 to 4:0
Connection from 1 to 5:0
Connection from 2 to 1:0
```

```
Connection from 2 to 2:1
Connection from 2 to 3:1
Connection from 2 to 4:0
Connection from 2 to 5:1
Connection from 3 to 1:0
Connection from 3 to 2:0
Connection from 3 to 3:1
Connection from 3 to 4:1
Connection from 3 to 5:0
Connection from 4 to 1:0
Connection from 4 to 2:1
Connection from 4 to 3:1
Connection from 4 to 4:1
Connection from 4 to 5:1
Connection from 5 to 1:0
Connection from 5 to 2:0
Connection from 5 to 3:1
Connection from 5 to 4:0
Connection from 5 to 5:1
Enter root node:2
Adjacent nodes of root node 2 are:
2      3      4      5
```

EXPERIMENT 6: Implement distance vector routing algorithm for obtaining routing tables at each node

```
#include<stdio.h>
```

Code:

```
// Structure to represent the routing table of each router
struct RoutingTable {
    unsigned distance[20];  // Distance to each destination node
    unsigned nextHop[20];   // Next hop to reach each destination node
} rt[10];

int main() {
    int costMatrix[20][20];  // Cost matrix input by user
    int numNodes;            // Number of routers/nodes in the network
    int i, j, k;             // Loop counters
    int updated;             // Flag to check if any update is made in the routing table

    // Read the number of routers
    printf("\nEnter the number of nodes (routers): ");
    scanf("%d", &numNodes);

    // Read the cost matrix
    printf("\nEnter the cost matrix:\n");
    for (i = 0; i < numNodes; i++) {
        for (j = 0; j < numNodes; j++) {
            scanf("%d", &costMatrix[i][j]);

            // Distance from a node to itself is 0
```

```
            if (i == j) {
                costMatrix[i][j] = 0;
            }

            // Initialize routing table
            rt[i].distance[j] = costMatrix[i][j]; // Initial distance
            rt[i].nextHop[j] = j;               // Initial next hop (direct connection)
        }
    }

    // Apply Distance Vector Routing algorithm (Bellman-Ford logic)
    do {
        updated = 0;  // Assume no updates

        for (i = 0; i < numNodes; i++) {
            for (j = 0; j < numNodes; j++) {
                for (k = 0; k < numNodes; k++) {
                    // Check if the path from i to j via k is shorter
                    if (rt[i].distance[j] > costMatrix[i][k] + rt[k].distance[j]) {
                        rt[i].distance[j] = costMatrix[i][k] + rt[k].distance[j];
                        rt[i].nextHop[j] = k; // Update next hop
                        updated = 1;          // Mark that an update occurred
                    }
                }
            }
        }
    } while (updated);  // Repeat until no updates

    // Print final routing tables
    for (i = 0; i < numNodes; i++) {
        printf("\nRouting Table for Router %d:\n", i + 1);
        printf("Destination\tNext Hop\tDistance\n");
        for (j = 0; j < numNodes; j++) {
            printf("     %d\t\t   %d\t\t   %d\n", j + 1, rt[i].nextHop[j] + 1, rt[i].distance[j]);
        }
    }
    printf("\n");
    return 0;
}
```

OUTPUT:

```
Enter the number of nodes (routers): 3

Enter the cost matrix:
1 2 3
4 5 6
6 7 8
Routing Table for Router 1:
Destination     Next Hop        Distance
    1                   1               0
    2                   2               2
    3                   3               3
```

Routing Table for Router 2:

Destination	Next Hop	Distance
1	1	4
2	2	0
3	3	6

Routing Table for Router 3:

Destination	Next Hop	Distance
1	1	6
2	2	7
3	3	0

EXPERIMENT 7: Implement data encryption and data decryption

// Simple C program to encrypt and decrypt a string

Code:

```
#include <stdio.h>

int main()
{
   int i, x;
   char str[100];

   printf("\nPlease enter a string: ");
   fgets(str, sizeof(str), stdin);  // safer alternative to gets()

   printf("\nPlease choose one of the following options:\n");
   printf("1 = Encrypt the string\n");
   printf("2 = Decrypt the string\n");
   printf("Enter your choice: ");
   scanf("%d", &x);

   // using switch-case statements
   switch (x)
   {
      case 1:
         for (i = 0; str[i] != '\0'; i++)
            str[i] = str[i] + 3;  // key for encryption is +3
         printf("\nEncrypted string: %s\n", str);
         break;

      case 2:
         for (i = 0; str[i] != '\0'; i++)
            str[i] = str[i] - 3;  // key for decryption is -3
         printf("\nDecrypted string: %s\n", str);
         break;

      default:
         printf("\nInvalid choice!\n");
   }

   return 0;
}
```

OUTPUT:

```
Please enter a string: hello

Please choose one of the following options:
1 = Encrypt the string
2 = Decrypt the string
Enter your choice: 1
Encrypted string: khoor
```

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

CODE:

```
#include <stdio.h>
int main() {
    int bucket_size, out_rate, n, input[10], i, store = 0;
    printf("Enter bucket size: ");
    scanf("%d", &bucket_size);
    printf("Enter output rate: ");
    scanf("%d", &out_rate);
    printf("Enter number of packets: ");
    scanf("%d", &n);

    printf("Enter packets at each second:\n");
    for(i = 0; i < n; i++)
        scanf("%d", &input[i]);

    for(i = 0; i < n; i++) {
        store += input[i];
        if(store > bucket_size) {
            printf("Bucket overflow! %d packets lost\n", store - bucket_size);
            store = bucket_size;
        }
        printf("At second %d: %d sent, %d remaining\n", i + 1, out_rate, store - out_rate);
        store -= out_rate;
        if(store < 0) store = 0;
    }
    return 0;
}
```

OUTPUT:

```
Enter bucket size: 10
Enter output rate: 4
Enter number of packets: 5
Enter packets at each second:
5 8 2 1 3
Bucket overflow! 3 packets lost
At second 1: 4 sent, 1 remaining
At second 2: 4 sent, 5 remaining
At second 3: 4 sent, 3 remaining
```

At second 4: 4 sent, 0 remaining
At second 5: 4 sent, 0 remaining\

EXPERIMENT 9: Write a program for frame sorting techniques used in buffers

Code:

```
#include <stdio.h>
  struct Frame {
    int id;
    char data[10];
};

int main() {
    int n, i, j;
    struct Frame temp, f[10];

    printf("Enter number of frames: ");
    scanf("%d", &n);

    for(i = 0; i < n; i++) {
        printf("Enter Frame ID and Data: ");
        scanf("%d %s", &f[i].id, f[i].data);
    }

    // Sorting frames by ID (Bubble Sort)
    for(i = 0; i < n-1; i++)
        for(j = 0; j < n-i-1; j++)
            if(f[j].id > f[j+1].id) {
                temp = f[j];
                f[j] = f[j+1];
                f[j+1] = temp;
            }

    printf("\nFrames after sorting:\n");
    for(i = 0; i < n; i++)
        printf("Frame %d : %s\n", f[i].id, f[i].data);

    return 0;
}
```

OUTPUT:

Enter number of frames: 3
Enter Frame ID and Data: 3 A
Enter Frame ID and Data: 1 B
Enter Frame ID and Data: 2 C

Frames after sorting:
Frame 1 : B
Frame 2 : C
Frame 3 : A

EXPERIMENT 10 : **Wireshark**

Aim: Wireshark

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

Solution:

What is Wireshark?
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.

Who uses Wireshark?
Government, educational institutions, corporations, small businesses, non-profits

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:

sudo apt-get install wireshark
sudodpkg-reconfigure wireshark-common
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:

sudodnf install wireshark-qt
sudousermod -a -G wireshark username

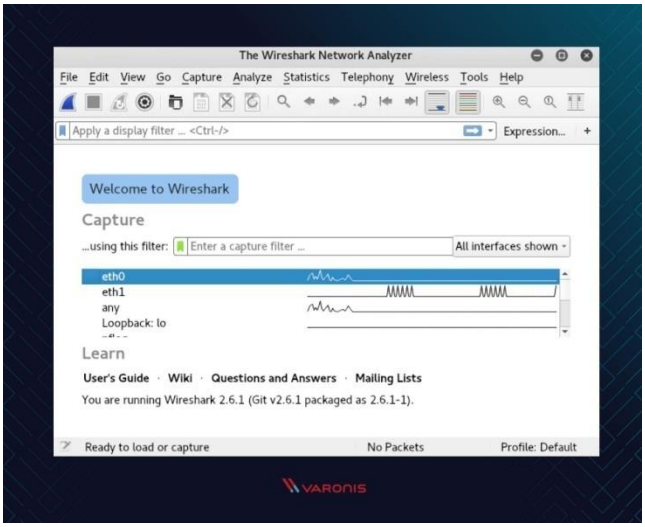
The first command installs the GUI and CLI version of Wireshark, and the second adds permissions to use Wireshark.

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.

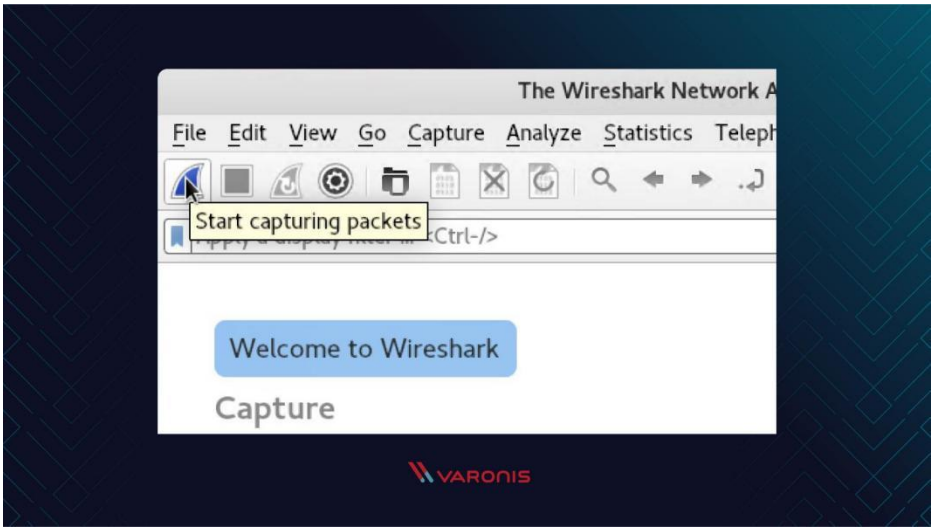
Capturing Data Packets on Wireshark
When you open Wireshark, you see a screen that shows you a list of all of the network connections you can monitor. You also have a capture filter field, so you only capture the network traffic you want to see.



Wireshark capture filter screenshot

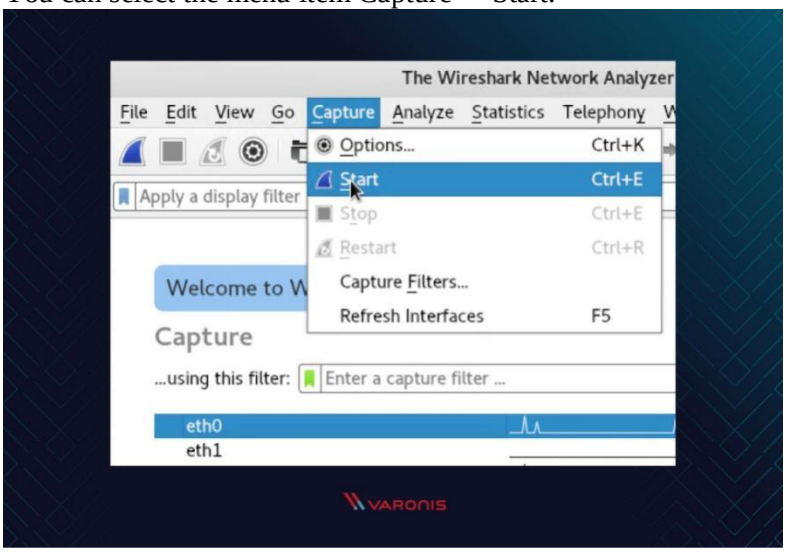
You can select one or more of the network interfaces using "shift left-click." Once you have the network interface selected, you can start the capture, and there are several ways to do that.

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



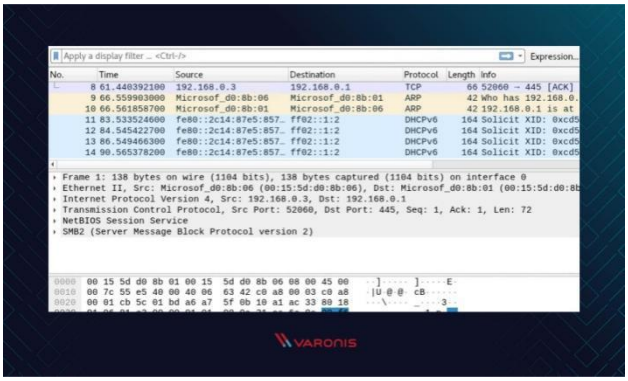
Wireshark how to start capturing screenshot

You can select the menu item Capture -> Start.



Wireshark capture packets screenshot
Or you could use the keystroke Control – E.

During the capture, Wireshark will show you the packets that it captures in real-time.



Wireshark how to stop capture screenshot

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.

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:

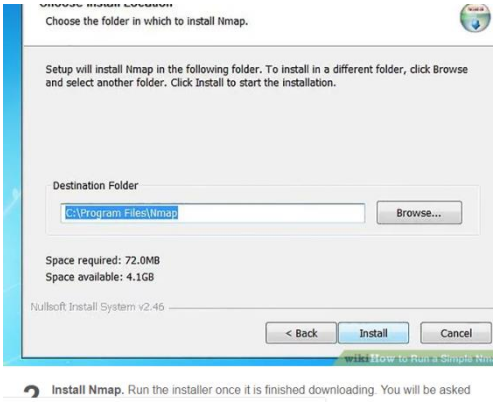
ls_ads.opnum==0x09

EXPERIMENT 11: How to run Nmap scan

Aim: How to run Nmap scan

Step 1: Download the Nmap installer.

This can be found for free from the developer's website. It is highly recommended that you download directly from the developer to avoid any potential viruses or fake files. Downloading the Nmap installer includes Zenmap, the graphical interface for Nmap which makes it easy for newcomers to perform scans without having to learn command lines.TheZenmap program is available for Windows, Linux, and Mac OS X. You can find the installation files for all operating systems on the Nmap website.



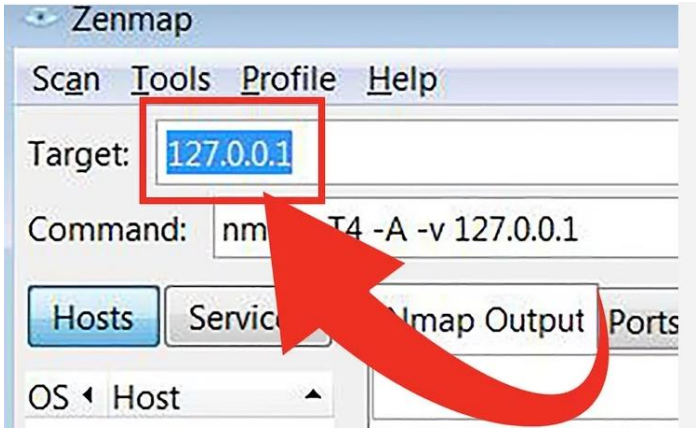
Step 2: Install Nmap.

Run the installer once it is finished downloading. You will be asked which components you would like to install. In order to get the full benefit of Nmap, keep all of these checked. Nmap will not install adware or spyware.

Step3 :Run the "Nmap – Zenmap" GUI program.

If you left your settings at default during installation, you should be able to see an icon for it on your desktop. If not, look in your Start menu. Opening Zenmap will start the program

.

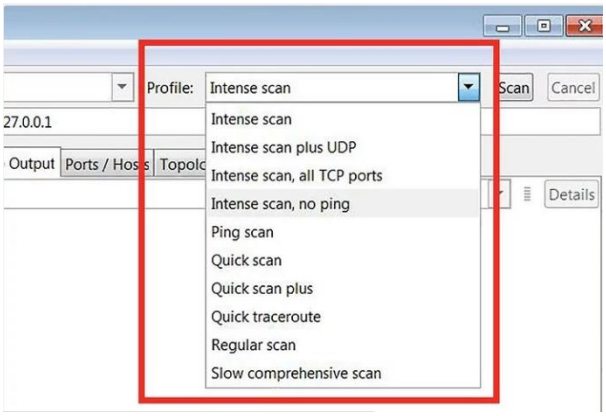


Step4:

Enter in the target for your scan.

The Zenmap program makes scanning a fairly simple process. The first step to running a scan is choosing your target. You can enter a domain (example.com), an IP address (127.0.0.1), a network (192.168.1.0/24), or a combination of those.

Depending on the intensity and target of your scan, running an Nmap scan may be against the terms of your internet service provider, and may land you in hot water. Always check your local laws and your ISP contract before performing Nmap scans on targets other than your own network.

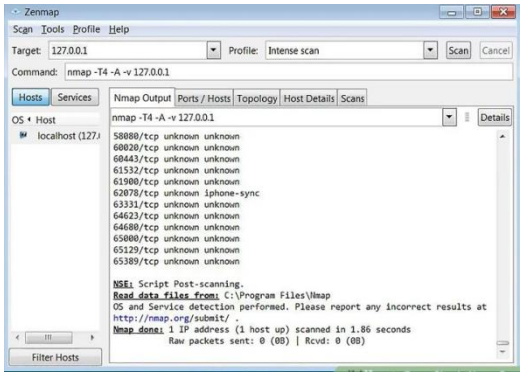


Step 5: Choose your Profile.

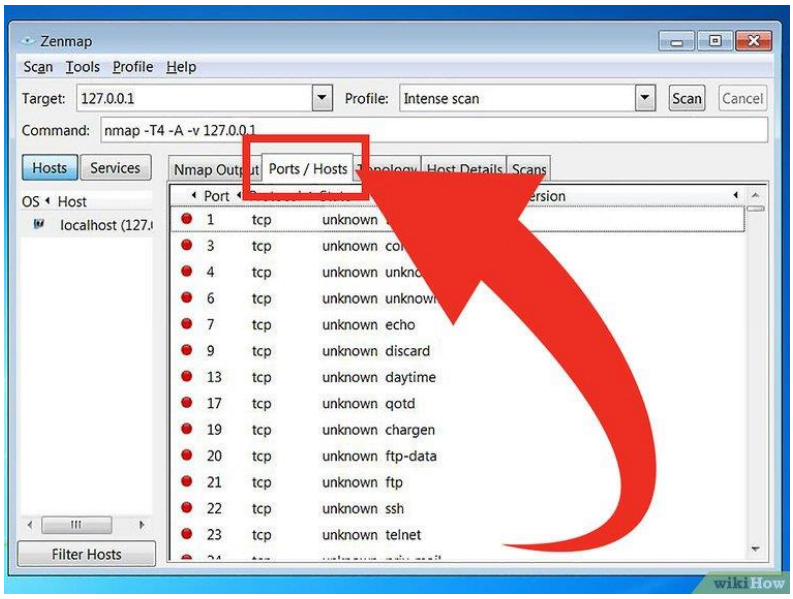
Profiles are preset groupings of modifiers that change what is scanned. The profiles allow you to quickly select different types of scans without having to type in the modifiers on the command line. Choose the profile that best fits your needs:[1]
Intense scan - A comprehensive scan. Contains Operating System (OS) detection, version detection, script scanning, traceroute, and has aggressive scan timing. This is considered an intrusive scan.
Ping scan - This scan simply detects if the targets are online, it does not scan any ports.
Quick scan - This is quicker than a regular scan due to aggressive timing and only scanning select ports.
Regular scan - This is the standard Nmap scan without any modifiers. It will return ping and return open ports on the target.

Step 6:Click Scan to start scanning.

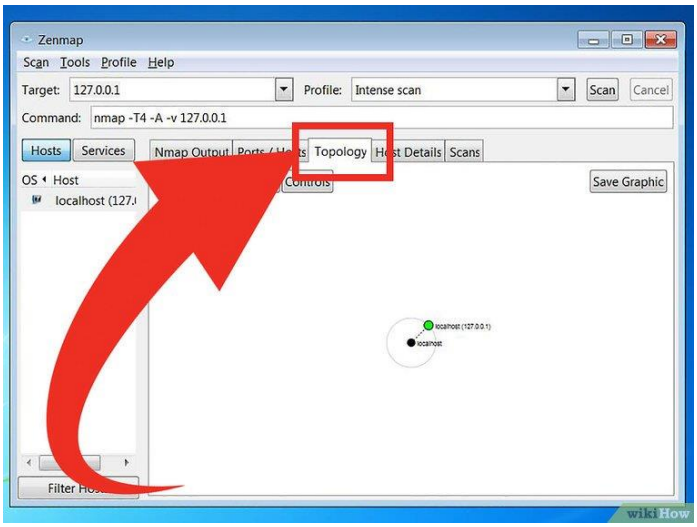
The active results of the scan will be displayed in the Nmap Output tab. The time the scan takes will depend on the scan profile you chose, the physical distance to the target, and the target's network configuration.



Step 7:Read your results. Once the scan is finished, you'll see the message "Nmap done" at the bottom of the Nmap Output tab. You can now check your results, depending on the type of scan you performed. All of the results will be listed in the main Nmap Output tab, but you can use the other tabs to get a better look at specific data.Ports/Hosts - This tab will show the results of your port scan, including the services for those ports.
Ports/Hosts - This tab will show the results of your port scan, including the services for those ports.

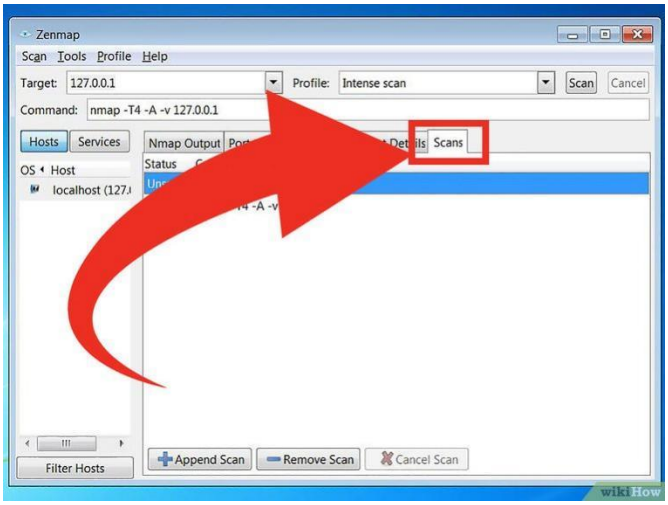


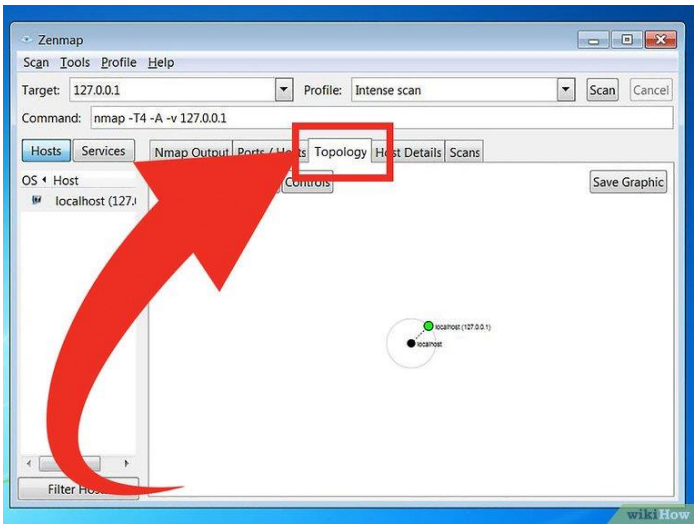
Topology - This shows the traceroute for the scan you performed. You can see how many hops your data goes through to reach the target



Host Details - This shows a summary of your target learned through scans, such as the number of ports, IP addresses, hostnames, operating systems, and more.

Scans - This tab stores the commands of your previously-run scans. This allows you to quickly re-scan with a specific set of parameters.





EXPERIMENT 12:

Aim : Operating System Detection using Nmap

Operating system (OS) detection can be enable with -O . On the other side -A parameter also provides operating system information. To use operating system detection nmap command should be run with root privileges because lower layer network manipulation will be done by nmap.

$ sudonmap -O localhost

```
ismail@ubu1:~$ nmap -O localhost
TCP/IP fingerprinting (for OS scan) requires root privileges.
QUITTING!
ismail@ubu1:~$ sudo nmap -O localhost
[sudo] password for ismail:

Starting Nmap 7.12 ( https://nmap.org ) at 2016-12-31 17:03 +03
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000070s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 995 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
53/tcp   open  domain
80/tcp   open  http
631/tcp  open  ipp
3306/tcp open  mysql
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3.19 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.19, Linux 3.8 - 4.4
Network Distance: 0 hops
```

1. Do the following using NS2 Simulator

i.NS2 Simulator-Introduction
ii. Simulate to Find the Number of Packets Dropped
iii.Simulate to Find the Number of Packets Dropped by TCP/UDP
iv.Simulate to Find the Number of Packets Dropped due to Congestion
v.Simulate to Compare Data Rate& Throughput.
vi.Simulate to Plot Congestion for Different Source/Destination
vii.Simulate to Determine the Performance with respect to Transmission of Packets

i.NS2 Simulator-Introduction:

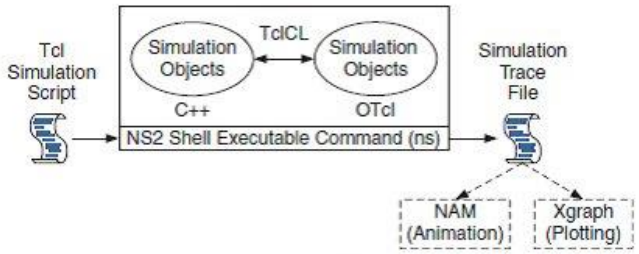
NS2 stands for Network Simulator Version 2. It is an open-source event-driven simulator designed specifically for research in computer communication networks.

Features of NS2

1. It is a discrete event simulator for networking research.
2. It provides substantial support to simulate bunch of protocols like TCP, FTP, UDP, https and DSR.
3. It simulates wired and wireless network.
4. It is primarily Unix based.
5. Uses TCL as its scripting language.
6. Otcl: Object oriented support
7. Tclcl: C++ and otcl linkage
8. Discrete event scheduler

Basic Architecture

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events. The C++ and the OTcl are linked together using TclCL



Basic architecture of NS.

ii. Simulate to Find the Number of Packets Dropped

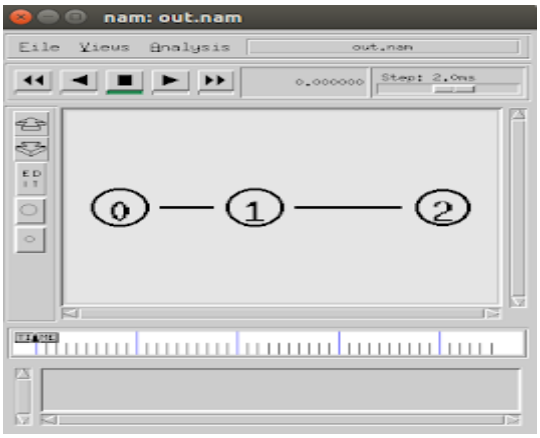
```
#Create a new Simulation Instance
set ns [new Simulator]
set bandwidth 1.75Mb
#Turn on the Trace and the animation files
set f [open out.tr w]
set nf [open out.nam w]
$ns trace-all $f
$ns namtrace-all $nf
#Define the finish procedure to perform at the end of the simulation
proc finish {} {
 global f nf ns
 $ns flush-trace
 close $f
 close $nf
 exec namout.nam&
 #exec awk -f 1.awk out.tr &
 exit 0
}
#Create the nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
#Label the nodes
$n0 label "TCP Source"
$n1 label "UDP Source"
$n2 label "Sink"
#Set the color
$ns color 1 red
$ns color 2 yellow
#Create the Topology
$ns duplex-link $n0 $n1 $bandwidth 10ms DropTail
$ns duplex-link $n1 $n2 1.75Mb 20ms DropTail
#Attach a Queue of size N Packets between the nodes n1 n2
$ns queue-limit $n1 $n2 10
#Make the Link Orientation
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n1 $n2 orient right
#Create a UDP Agent and attach to the node n1
set udp0 [new Agent/UDP]
```

```
$ns attach-agent $n0 $udp0
#Create a CBR Traffic source and attach to the UDP Agent
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
#Specify the Packet Size and interval
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
#Create a Null Agent and attach to the node n2
set null0 [new Agent/Null]
$ns attach-agent $n2 $null0
#Connect the CBR Traffic source to the Null agent
$ns connect $udp0 $null0
#Create a TCP agent and attach to the node n0
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
#Create a FTP source and attach to the TCP agent
set ftp0 [new Application/FTP]
#Attach the FTP source to the TCP Agent
$ftp0 attach-agent $tcp0
#Create a TCPSink agent and attach to the node n2
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
#Specify the Max file Size in Bytes
$ftp0 set maxPkts_ 1000
#Connect the TCP Agent with the TCPSink
$ns connect $tcp0 $sink
$udp0 set class_ 1
$tcp0 set class_ 2
#Schedule the Events
$ns at 0.1 "$cbr0 start"
$ns at 1.0 "$ftp0 start"
$ns at 4.0 "$ftp0 stop"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```

Now run the code using

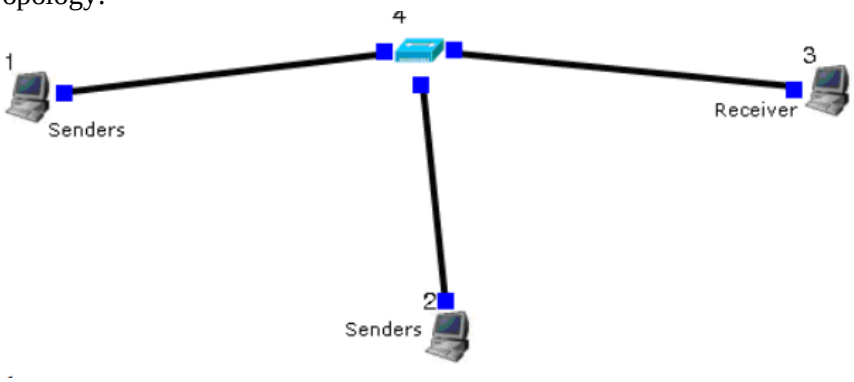
ns filename.tcl

OUTPUT:



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

AIM: Simulate a four-node point-to-point network and connect the link as follows:
Apply a TCP agent between n0 to n3 and apply a UDP agent between n1 and n3. Apply relevant applications over TCP and UDP agents changing the parameters and determine the number of packets sent by two agents.
Topology:-



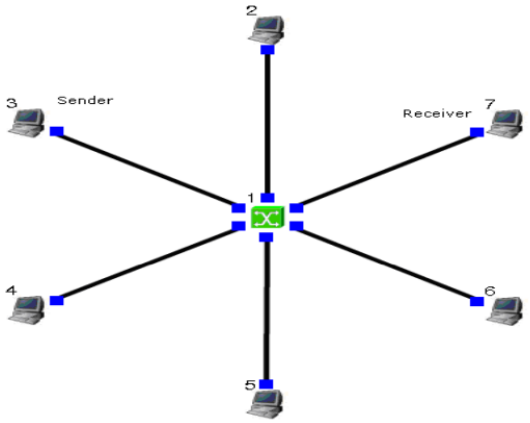
Sender:- stcp –p 3000 –l 1024 1.0.1.3 stg –u 1024 1.0.1.3
Receiver:- rtcp –p 3000 –l 1024 rtg –u 3000
Parameters:- Throughput of incoming and outgoing Packets
Step1: Drawing topology
1. Select/click the HOST icon on the toolbar and click the left mouse button on the editor, to place a host on the editor. Repeat the above procedure and place two other hosts “HOST2” and “HOST3” on the editor
2. Select/click the HUB (or SWITCH) icon on the toolbar and click the left mouse button on the editor, to place a HUB (or SWITCH) on the editor. NETWORKS LABORATORY 2016-2017 Dept. of CSE, GCEM, Bangalore PAGE 17 SEMESTER-VII
3. Click on the LINK icon on the toolbar and connect HOST1 to HUB, HOST2 to HUB and HUB to HOST3 4. Click on the “E” icon on the toolbar to save the current topology e.g: file2.tpl (Look for the ******.tpl extension.) NOTE: Changes cannot / (should not) be done after selecting the “E” icon.
Step2: Configuration
1. Double click the left mouse button while cursor is on HOST1 to open the HOST window. 2. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. stcp –p 21 –l 1024 1.0.1.3 3. Click OK button on the command window to exit 4. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 5. Select LOG STATISTICS and select checkbox for output throughput in the MAC window 6. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 7.Double click the left mouse button while cursor is on HOST2 to open the HOST window. 8. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. stg –u 1024 100 1.0.1.3
9. Click OK button on the command window to exit 10. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 11. Select LOG STATISTICS and select checkbox for output throughput in the MAC window 12. Click OK button on the MAC window to exit and once again click on the OK buttonon the HOST window to exit. 13. Double click the left mouse button while cursor is on HOST3 to open the HOST window. 14. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. rtcp –p 21 –l 1024 15. Click OK button on the command window to exit. 16. Also add the following command on HOST3 rtg –u –w log1 17. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 18. Select LOG STATISTICS and select checkbox for input and output throughput in the MAC window 19. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit.
Step3: Simulate
i. Click “R” icon on the tool bar ii. Select Simulation in the menu bar and click/ select RUN in the dropdown list to execute the simulation. iii. To start playback select “►” icon located at the bottom right corner of the editor. iv. To view results, Open up new TERMINAL window, move to file2.results folder and open input and output throughput log files in separate TERMINAL window. Caution: file2 is the hypothetical name given to this simulation. (Refer Step 1.4)

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

EXPERIMENT 13:

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

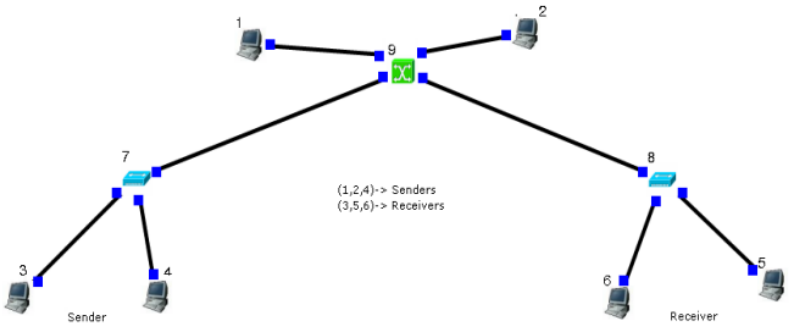
Topology:-



Sender:- stcp –p 2000 –l 1024 1.0.1.4
Receiver:- rtcp –p 2000 –l 1024
Command Console:- Goto tools->
simulation time and change Simulation time to 100. During run mode, double click host 2 and then click command console. And execute the following command. ping 1.0.1.4 Parameters:- Drop Packets and Collision Packets.
Step1: Drawing topology 1. Select/click the SUBNET icon on the toolbar and click the left mouse button on the editor, to place a SUBNET on the editor. 2. A pop up window appears requesting the number of nodes and radius for the subnet Set number of nodes=6; Set radius of subnet >150 3. Click on the "E" icon on the toolbar to save the current topology e.g: file4.tpl (Look for the ******.tpl extension.) NOTE: Changes cannot / (should not) be done after selecting
Step2: Configuration 4. Double click the left mouse button while cursor is on a HOST to open the HOST window. 5. Click NODE EDITOR Button on the HOST window and select the INTERFACE tab (1st tab) from the modal window that pops up. 6. Determine the IP address of the selected host. 7. Click OK button on the INTERFACE window to exit and once again click on the OK button on the HOST window to exit. 8. Repeat the above step for 2 other HOSTS 9. Also click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 10. Select LOG STATISTICS and select checkbox for drop and collision log statistics in the MAC window 11. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 12. Repeat steps 6 to 9 for the other hosts selected at step 5. 13. Select G_Setting from the menu bar and select Simulation from the drop down list Set simulation time>600sec Step3: Simulate i. Click "R" icon on the tool bar ii. Select Simulation in the menu bar and click/ select RUN in the dropdown list to execute the simulation. iii. During simulation, open a new terminal window.
iv. Type ping IP address of a host in the subnet at the command prompt. v. To view results, Open up new TERMINAL window, move to file4.results folder and open drop and collision log files in separate TERMINAL window. Caution: file4 is the hypothetical name given to this simulation. (Refer Step 1.3)

V.Simulate to Compare Data Rate & Throughput AIM:

Simulate an Ethernet LAN using N nodes (6-10), change error rate and data rate and compare throughput.
Topology:-



Sender:- stcp –p 2000 –l 1024 1.0.1.4

Receiver:- rtcp –p 2000 –l 1024 Double click on receiver link and change BER to 0.000001, Run Again.

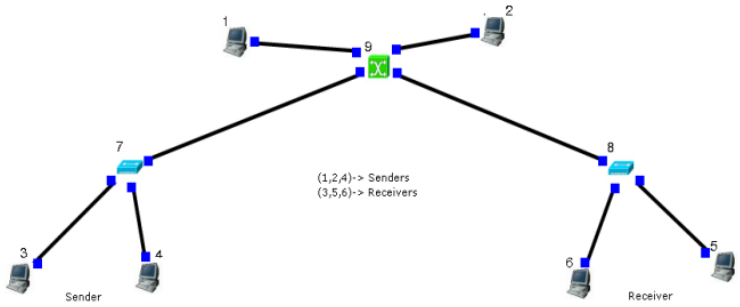
Parameters:- Throughput of outgoing Packets Step1: Drawing topology 1. Select/click the HOST icon on the toolbar and click the left mouse button on the editor, to place HOST1 on the editor. Repeat the above procedure and place 5 other hosts "HOST2", "HOST3", "HOST4", "HOST5", and "HOST6"on the editor. 2. Select/click the HUB icon on the toolbar and click the left mouse button on the editor, to place HUB1 on the editor. Repeat the above procedure and place another host "HUB2" on the editor 3. Click on the LINK icon on the toolbar and connect HOST1, HOST2 and HOST3 to HUB1, HOST4, HOST5 and HOST6 to HUB2. 4. Select/click the SWITCH icon on the toolbar and click the left mouse button on the editor, to place SWITCH1 on the editor. 5. Click on the LINK icon on the toolbar and connect HUB1 to SWITCH1 and HUB2 to SWITCH1. 6. Click on the "E" icon on the toolbar to save the current topology e.g: file5.tpl (Look for the ******.tpl extension.) NOTE: Changes cannot / (should not) be done after selecting the "E" icon. Step2: Configuration 1. Double click the left mouse button while cursor is on HOST1 to open the HOST window. 2. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. stcp –p 21 –l 1024 1.0.1.4 3. Click OK button on the command window to exit and once again click on the OK button on the HOST window to exit. 4. Repeat this step at HOST 2 and HOST3, but use different commands stcp –p 21 –l 1024 1.0.1.5 at HOST2 stcp –p 21 –l 1024 1.0.1.6 at HOST3 5. Double click the left mouse button while cursor is on HOST4 to open the HOST window. 6. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. rtcp –p 21 –l 1024 7. Click OK button on the command window to exit. 8. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 9. Select LOG STATISTICS and select checkbox for output throughput in the MAC window 10. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit.

11. Repeat this step at HOST 5 and HOST6, but use different commands rtcp –p 21 –l 1024 at HOST5 rtcp –p 21 –l 1024 at HOST6 12. Double click the left mouse button while cursor is on HOST5 to open the HOST window. 13. Click NODE EDITOR Button on the HOST5 window and select the PHYSICAL tab from the modal window that pops up. 14. Change Bit Error Rate 15. Click OK button on the PHYSICAL window to exit and once again click on the OK button to return to the HOST window 16. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 17. Select LOG STATISTICS and select checkbox for output throughput in the MAC window 18. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 19. Repeat this step HOST6, Change Bandwidth this time while undoing the change in Bit Error Rate, also select the output throughput at HOST6. Step3: Simulate i. Click "R" icon on the tool bar ii. Select Simulation in the menu bar and click/ select RUN in the dropdown list to execute the simulation. iii. To start playback select "►" icon located at the bottom right iv. To view results, Open up new TERMINAL window, move to file5.results folder and open output throughput log files in separate TERMINAL window. Caution: file5 is the hypothetical name we gave to this simulation (Refer Step 1.7)

Vi Simulate to Plot Congestion for Different Source/Destination

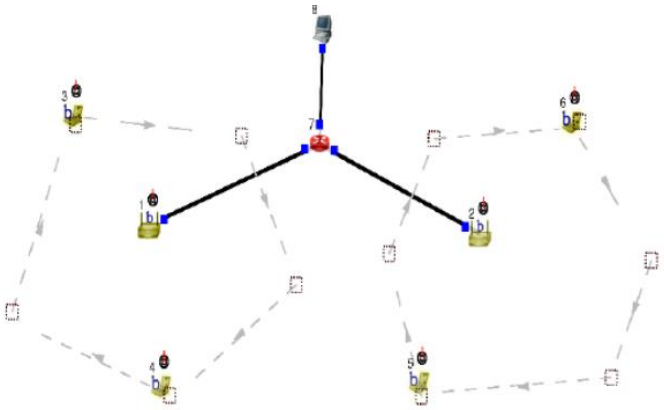
AIM: Simulate an Ethernet LAN using N nodes and set multiple traffic nodes and plot congestion window for different source/destination.

Topology:-



Sender:- stcp –p 2000 –l 1024 1.0.1.4
Receiver:- rtcp –p 2000 –l 1024
Parameters:- Receiver side Collision Packets and Drop Packets Step1: Drawing topology
1. Select/click the HOST icon on the toolbar and click the left mouse button on the editor, to place HOST1 on the editor. Repeat the above procedure and place 3 other hosts “HOST2”, “HOST3”, “HOST4”, “HOST5”, and “HOST6”on the editor.
2. Select/click the HUB icon on the toolbar and click the left mouse button on the editor, to place HUB1 on the editor. Repeat the above procedure and place another host “HUB2” on the editor
3. Click on the LINK icon on the toolbar and connect HOST1, HOST2 and HOST3 to HUB1, HOST4, HOST5 and HOST6 to HUB2.
4. Select/click the SWITCH icon on the toolbar and click the left mouse button on the NETWORKS LABORATORY 2016-2017 Dept. of CSE, GCEM, Bangalore PAGE 26 SEMESTER-VII editor, to place SWITCH1 the editor.
5. Click on the LINK icon on the toolbar and connect HUB1 to SWITCH1 and HUB2 to SWITCH1. 6. Click on the “E” icon on the toolbar to save the current topology e.g: file7.tpl (Look for the *******.tpl extension.) NOTE: Changes cannot / (should not) be done after selecting the “E” icon.
Step2: Configuration
1. Double click the left mouse button while cursor is on HOST1 to open the HOST window.
2. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. stcp –p 21 –l 1024 1.0.1.4
3. Click OK button on the command window to exit and once again click on the OK button on the HOST window to exit. 4. Repeat this step at HOST 2 and HOST3, but use different commands stcp –p 23 –l 1024 1.0.1.5 at HOST2 stcp –p 25 –l 1024 1.0.1.6 at HOST3 5. Double click the left mouse button while cursor is on HOST4 to open the HOST window. 6. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. rtcp –p 21 –l 1024 7. Click OK button on the command window to exit. 8. Click NODE EDITOR Button on the HOST window and select the MAC tab from the modal window that pops up. 9. Select LOG STATISTICS and select checkbox for Number of drop and collisions packets in the MAC window 10. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 11. Repeat this step at HOST 5 and HOST6, but use different commands rtcp –p 23 –l 1024 at HOST5 rtcp –p 25 –l 1024 at HOST6
12. Double click the left mouse button while cursor is on HOST5 to open the HOST window. 13. Click NODE EDITOR Button on the HOST5 window and select the MAC tab from the modal window that pops up. 14. Select LOG STATISTICS and select checkbox for Number of drop and collisions packets in the MAC window 15. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 16. Also select the drop and collisions at HOST6. Step3: Simulate i. Click “R” icon on the tool bar ii. Select Simulation in the menu bar and click/ select RUN in the dropdown list to execute the simulation. iii. To start playback select “►” icon located at the bottom right corner of the editor. iv. To plot congestion window select Tools in the menu bar and select PLOT GRAPH in the drop down list. v. In the Graph window, select File->OPEN, move to file7.results folder and the drop and collision log file. vi. To open another Graph window, Select File->New tab on the drop down list to open up to a maximum of 6 windows vii. To view results, Open up new TERMINAL window, move to file7.results folder and open input and output throughput log files in separate TERMINAL window.
Caution: file7 is the hypothetical name given to this simulation.

Vii Simulate to Determine the Performance with respect to Transmission of Packets
AIM: Simulate simple ESS and with transmitting nodes in wireless LAN by simulation and determine the performance with respect to transmission of packets.
Topology:-



Click on “access point”. Goto wireless interface and tick on “show transmission range and then click OK.
Double click on Router -> Node Editor and then Left stack -> throughput of Incoming packets Right stack -> throughput of Outgoing packets Select mobile hosts and access points then click on. Tools -> WLAN mobile nodes-> WLAN Generate infrastructure. Subnet ID: Port number of router
(2) Gateway ID: IP address of router Mobile Host 1 ttcp –t –u –s –p 3000 1.0.1.1 Mobile Host 1 ttcp –t –u –s –p 3001 1.0.1.1 NETWORKS LABORATORY 2016-2017 Dept. of CSE, GCEM, Bangalore PAGE 29 SEMESTER-VII Host(Receiver) ttcp –r –u –s –p 3000 ttcp –r –u –s –p 3001 Run and then play to plot the graph.
Step1: Drawing topology 1. Select/click the HOST icon on the toolbar and click the left mouse button on the editor, to place HOST1 on the editor.
2. Select/click the ROUTER icon on the toolbar and click the left mouse button on the editor, to place ROUTER1 on the editor.
3. Select/click the WIRELESS ACCESS POINT(802.11b) icon on the toolbar and click the left mouse button on the editor, to place ACCESS POINT 1 on the editor. Repeat this procedure and place ACCESS POINT 2 on the editor.
4. Select/click the MOBILE NODE (infrastructure mode) icon on the toolbar and click the left mouse button on the editor, to place MOBILE NODE 1 on the editor. Repeat this procedure and place MOBILE NODE 2, MOBILE NODE3 and MOBILE NODE 4 on the editor.
5. Click on the LINK icon on the toolbar and connect ACCESS POINT1 to ROUTER1 and ACCESS POINT2 to ROUTER1 6. Click on the “Create a moving path” icon on the toolbar and draw moving path across MOBILE NODE 1 and 2, Repeat for MOBILE NODE 3and 4 (Accept the default speed value 10 and close the window, Click the right mouse button to terminate the path). To create Subnet 7. Select wireless subnet icon in the toolbar now select MOBILE NODE1, MOBILE NODE2 and ACCESS POINT1 by clicking on left mouse button, and clicking right mouse button will create a subnet. 8. Repeat the above step for MOBILE NODE3, MOBILE NODE4 and ACCESS POINT2. 9. Click on the “E” icon on the toolbar to save the current topology e.g: file8.tpl (Look for the ******.tpl extension.) NOTE: Changes cannot / (should not) be done after selecting the “E” icon. Step2: Configuration 1. Double click the left mouse button while cursor is on HOST1 to open the HOST window. NETWORKS LABORATORY 2016-2017 Dept. of CSE, GCEM, Bangalore PAGE 30 SEMESTER-VII 2. Select Add button on the HOST window to invoke the command window and provide the following command in the command textbox. ttcp –r –u –s –p 8001 3. Click OK button on the command window to exit 4. Repeat this step and add the following commands at HOST1 ttcp –r –u –s –p 8002 ttcp –r –u –s –p 8003 ttcp –r –u –s –p 8004 5. Click NODE EDITOR Button on the HOST1 window and select the MAC tab from the modal window that pops up. 6. Select LOG STATISTICS and select checkbox for Input throughput in the MAC window 7. Click OK button on the MAC window to exit and once again click on the OK button on the HOST window to exit. 8. Double click the left mouse button while cursor is on MOBILE NODE 1 to open the MOBILE NODE window. 9. Select Application tab and select Add button to invoke the command window and provide the following command in the command textbox. ttcp –t –u –s –p 80011.0.2.2 (host’s ip address) 10. Click NODE EDITOR Button on the MOBILE NODE1 window and select the MAC tab from the nodal window that pops up. 11. Select LOG STATISTICS and select checkbox for Output throughput in the MAC window 12. Click OK button on the MAC window to exit and once again click on the OK button on the MOBILE NODE1 window to exit. 13. Repeat the above steps (step 8 to step12) for the MOBILE NODE2,3 and 4 and add the following commands at MOBILE NODE2:- ttcp –t –u –s –p 8002 1.0.2.2 MOBILE NODE3:- ttcp –t –u –s –p 8003 1.0.2.2 MOBILE NODE4:- ttcp –t –u –s –p 8004 1.0.2.2 14. Double

click the left mouse button while cursor is on ROTER1 to open the ROUTER window. NETWORKS LABORATORY 2016-2017 Dept. of CSE, GCEM, Bangalore PAGE 31 SEMESTER-VII 15. Click NODE EDITOR Button on the ROUTER1 window and you can see three stacks. two stacks for two ACCESS POINTS and another stack for HOST1 which is connected to the ROUTER1. 16. Select the MAC tab of ACCESS POINT1 and Select LOG STATISTICS and select checkbox for Input throughput in the MAC window. Click OK button on the MAC window to exit. 17. Select the MAC tab of ACCESS POINT2 and Select LOG STATISTICS and select checkbox for Input throughput in the MAC window. Click OK button on the MAC window to exit. 18. Select the MAC tab of HOST1 and Select LOG STATISTICS and select checkbox for Output throughput in the MAC window. Click OK button on the MAC window to exit. Step3: Simulate i. Click "R" icon on the tool bar ii. Select Simulation in the menu bar and click/select RUN in the dropdown list to execute the simulation. iii. To start playback select "►" icon located at the bottom right corner of the editor. iv. MOBILE NODE's start moving across the paths already drawn.

Caution: file8 is the hypothetical name given to this simulation.