

JAWAHARLAL COLLEGE OF ENGINEERING AND TECHNOLOGY

JAWHAR GARDENS, LAKKIDI, MAGALAM (PO), PALAKKAD



# CSL332 – NETWORKING LAB

# LABORATORY MANUAL

# **B.TECH - CSE**

SEMESTER-VI (2019 Regulation-KTU)

**Department of Computer Science & Engineering** 



# JAWAHARLAL COLLEGE OF ENGINEERING & TECHNOLOGY

Approved by AICTE - ISO 9001:2015 Certified, Affiliated to APJ Abdul Kalam Technological University, Kerala Jawahar Gardens, Lakkidi, Mangalam, Palakkad District,Ottapalam, Kerala – 679301



JAWAHARLAL COLLEGE OF ENGINEERING AND TECHNOLOGY

JAWHAR GARDENS, LAKKIDI, MAGALAM (PO), PALAKKAD

# **INSTITUTE VISION**

Emerge as a centre of excellence for professional education to produce high quality engineers and entrepreneurs for the development of the region and the Nation.

# **INSTITUTE MISSION**

- To become an ultimate destination for acquiring latest and advanced knowledge in the multidisciplinary domains.
- To provide high quality education in engineering and technology through innovative teaching-learning practices, research and consultancy, embedded with professional ethics.
- To promote intellectual curiosity and thirst for acquiring knowledge through outcome based education.
- To have partnership with industry and reputed institutions to enhance the employability skills of the students and pedagogical pursuits.
- To leverage technologies to solve the real life societal problems through community services.





JAWHAR GARDENS, LAKKIDI, MAGALAM (PO), PALAKKAD



# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

# **VISION**

To produce competent professionals with research and innovative skills, by providing them with most conducive environment for quality academic and research oriented undergraduate and postgraduate education along with moral values committed to build a vibrant nation.

# **MISSION**

- Provide a learning environment to develop creativity and problem solving skills in a professional manner.
- Expose to latest technologies and tools used in the field of computer science.
- Provide a platform to explore the industries to understand the work culture and expectation of an organization.
- Enhance Industry Institute Interaction program to develop the entrepreneurship skills.
- Develop research interest among students which will impart a better life for the society and the nation.



JAWHAR GARDENS, LAKKIDI, MAGALAM (PO), PALAKKAD



# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Graduates in Computer Science and Engineering will be able to:

- **PEO1:** Provide high quality knowledge in computer science and engineering required for a computer professional to identify and solve problems in various application domains.
- **PEO2:** Persist with the ability in innovative ideas in computer support systems and transmit the knowledge and skills for research and advanced learning.
- **PEO3:** Manifest the motivational capabilities and turn on social and economic commitment to community services.

# PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Use fundamental knowledge of mathematics to solve problems using suitable analysis methods, data structure and algorithms.
- **PSO2:** Interpret the basic concepts and methods of computer systems and technical specifications to provide accurate solutions.
- **PSO3:** Apply theoretical and practical proficiency with a wide area of programming knowledge, design new ideas and innovations towards research.



# JAWHAR GARDENS, LAKKIDI, MAGALAM (PO), PALAKKAD



# The following are the Program Outcomes of Engineering Students:

	in the first and the first of t
PO1	<b>Engineering knowledge</b> : Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	<b>Problem analysis</b> : Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	<b>Design/development of solutions</b> : 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	<b>Conduct investigations of complex problems</b> : 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	<b>Modern tool usage</b> : 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	<b>The engineer and society</b> : 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	<b>Environment and sustainability</b> : Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	<b>Ethics</b> : Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	<b>Individual and team work</b> : Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	<b>Communication</b> : 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	<b>Project management and finance</b> : 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	<b>Life-long learning</b> : 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.

#### PROGRAM 1: FAMILIARITY WITH BASICS OF NETWORK CONFIGURATION FILES AND NETWORKING

#### commands in Linux - ifconfig, netstat, ping, arp, telnet, ftp, finger. Program Objective:

Understanding and using of commands like ifconfig, netstat, ping, arp, telnet, ftp, finger, traceroute, whois

#### **Program Description:**

UNIX utilities are commands that, generally, perform a single task. It may be as simple as printing the date and time, or a complex as finding files that match many criteria throughout a directory hierarchy

# IFCONFIG

The Unix command **ifconfig** (short for interface **config**urator) serves to configure and control TCP/IP network interfaces from a command line interface (CLI).

Common uses for ifconfig include setting an interface's IP address and netmask, and disabling or enabling a given interface. On some Unix-like operating systems, ifconfig is used to configure, or view the configuration of, a network interface.

Type ipconfig/all to see detailed configuration information for all network adapters configured on the computer.

The **ipconfig** (short for IP Configuration) is a basic, yet popular, Windows network command-line utility used to display the TCP/IP network configuration of a computer. If you are familiar with Linux, this tool is similiar to ifconfig. This tool is often used for troubleshooting network connectivity issues. With ipconfig, you can identify the types of network adapaters on your computer, the computer's IP address, the IP addresses of the DNS (Domain Name System) servers being used, and much more.

ifconfig -a

Displays the configuration of all interfaces, both active and inactive.

# NETSTAT

**netstat** (**net**work **stat**istics) is a command-line tool that displays network connections (both incoming and outgoing), routing tables, and a number of network interface statistics.

It is used for finding problems in the network and to determine the amount of traffic on the network as a performance measurement.

# Parameters

Parameters used with this command must be prefixed with a hyphen (-) rather than a slash (/).

-a : Displays all active TCP connections and the TCP and UDP ports on which the computer is listening.

-e : Displays ethernet statistics, such as the number of bytes and packets sent and received. This parameter can be combined with -s.

-f : Displays fully qualified domain names <FQDN> for foreign addresses.

-i : Displays network interfaces and their statistics (not available under Windows)

-n : Displays active TCP connections, however, addresses and port numbers are expressed numerically and no attempt is made to determine names.

-o : Displays active TCP connections and includes the process ID (PID) for each connection.

-p Linux: Process : Show which processes are using which sockets

#### PING

**Ping** is a computer network tool used to test whether a particular host is reachable across an IP network; it is also used to self-test the network interface card of the computer, or as a speed test. It works by sending ICMP "echo request" packets to the target host and listening for ICMP "echo response" replies. Ping does not estimate the round-trip time, as it does not factor in the user's connection speed, but instead is used to record any packet loss, and print a statistical summary when finished.

The word *ping* is also frequently used as a verb or noun, where it is usually incorrectly used to refer to the round-trip time, or measuring the round-trip time. The ping command can be used to test end-to-end connectivity between two host devices. It measures the round-trip time for a message to get from source to destination. The ping command can be used to verify the connectivity between two network devices that are IP (Internet Protocol) based.

#### ARP

In computer networking, the **Address Resolution Protocol** (**ARP**) is the method for finding a host's link layer (hardware) address when only its Internet Layer (IP) or some other Network Layer address is known.

ARP has been implemented in many types of networks; it is not an IP-only or Ethernet-only protocol. It can be used to resolve many different network layer protocol addresses to interface hardware addresses, although, due to the overwhelming prevalence of IPv4 and Ethernet, ARP is primarily used to translate IP addresses to Ethernet MAC addresses.

# How to Use ARP to Find a MAC Address?

In Windows, Linux, and other operating systems, the command line utility ARP (Address Resolution Protocol) shows local MAC address information stored in the ARP cache. However, it only works within the small group of computers on a local area network (LAN), not across the internet.

ARP is intended to be used by system administrators, and it is not typically a useful way to track down computers and people on the internet.

TCP/IP computer networks use both the IP addresses and MAC addresses of connected client devices. While the IP address changes over time, the MAC address of a network adapter always stays the same.

Start by pinging the device you want the MAC to address for:

ping 192.168.86.45

The ping command establishes a connection with the other device on the network and should show a result like this:

Pinging 192.168.86.45 with 32 bytes of data: Reply from 192.168.86.45: bytes=32 time=290ms TTL=128 Reply from 192.168.86.45: bytes=32 time=3ms TTL=128 Reply from 192.168.86.45: bytes=32 time=176ms TTL=128Reply from 192.168.86.45: bytes=32 time=3ms TTL=128 Use the following ARP command to get a list that shows the MAC address of the device you pinged:

#### arp -a

The results may look something like this but probably with many other entries:

Interface: 192.168.86.38 --- 0x3 Internet Address Physical Address Type 192.168.86.1 70-3a-cb-14-11-7a dynamic 192.168.86.45 98-90-96-B9-9D-61 dynamic 192.168.86.255 ff-ff-ff-ff-ff static 224.0.0.22 01-00-5e-00-00-16 static 224.0.0.251 01-00-5e-00-00-fb static Find the device's IP address in the list. The MAC address is shown right next to it. In this example, the IP address is 192.168.86.45, and its MAC address is 98-90-96-B9-9D-61.

# TELNET

**Telnet** (**Tel**ecommunication **net**work) is a network protocol used on the Internet or local area network (LAN) connections. In Linux, the telnet command is used to create a remote connection with a system over a TCP/IP network.

Typically, telnet provides access to a command-line interface on a remote machine.

The term *telnet* also refers to software which implements the client part of the protocol. Telnet clients are available for virtually all platforms.

#### **Protocol details:**

Telnet is a client-server protocol, based on a reliable connection-oriented transport. Typically this protocol is used to establish a connection to TCP port 23

# FTP

# File Transfer Protocol (FTP):

FTP is a network protocol used to transfer data from one computer to another through a network such as the Internet.FTP is a file transfer protocol for exchanging and manipulating files over a TCP computer network. An FTP client may connect to an FTP server to manipulate files on that server.FTP runs over TCP. It defaults to listen on port 21 for incoming connections from FTP clients. A connection to this port from the FTP Client forms the control stream on which commands are passed from the FTP client to the FTP server and on occasion from the FTP server to the FTP client. FTP uses out-of-band control, which means it uses a separate connection for control and data. Thus, for the actual file transfer to take place, a different connection is required which is called the data stream.

To establish an FTP connection to a remote system, use the ftp command with the remote system's IP address:

ftp [IP]

For instance, connecting to a remote server with the IP address 192.168.100.9:

ftp 192.168.100.9

#### FINGER:

In computer networking, the **Name/Finger protocol** and the **Finger user information protocol** are simple network protocols for the exchange of human-oriented status and user information. Finger command is a user information lookup command which gives details of all the users logged

in. This tool is generally used by system administrators. It provides details like login name, user name, idle time, login time, and in some cases their email address even.

finger -p ch Display information about the user ch. Output appears similar to the following:

Login name: admin In real life: Computer Hope On since Feb 11 23:37:16 on pts/7 from domain.computerhope.com 28 seconds Idle Time Unread mail since Mon Feb 12 00:22:52 2001

# **TRACEROUTE:**

**traceroute** is a computer network tool used to determine the route taken by packets across an IP network . An IPv6 variant, **traceroute6**, is also widely available. Traceroute is often used for network troubleshooting. By showing a list of routers traversed, it allows the user to identify the path taken to reach a particular destination on the network. This can help identify routing problems or firewalls that may be blocking access to a site. Traceroute is also used by penetration testers to gather information about network infrastructure and IP ranges around a given host. It can also be used when downloading data, and if there are multiple mirrors available for the same piece of data, one can trace each mirror to get a good idea of which mirror would be the fastest to use.

In other words, traceroute command in Linux prints the route that a packet takes to reach the host. This command is useful when you want to know about the route and about all the hops that a packet takes.

The traceroute command in Windows is *tracert*. On a Linux system, the command is *tracerouting*. A typical tracert on a Windows machine would look like the following. tracert www.google.com Tracing route to www.google.com [74.125.227.179] over a maximum of 30 hops: 1 1 ms <1 ms 1 ms 192.168.1.1 2 7 ms 6 ms 6 ms 10.10.1.2 3 7 ms 8 ms 7 ms 10.10.1.45 4 9 ms 8 ms 8 ms 10.10.25.45 5 9 ms 10 ms 9 ms 10.10.85.99 6 11 ms 51 ms 10 ms 10.10.64.2 7 11 ms 10 ms 10 ms 10.10.5.88 8 11 ms 10 ms 11 ms 216.239.46.248 9 12 ms 12 ms 12 ms 72.14.236.98 10 18 ms 18 ms 18 ms 66.249.95.231 11 25 ms 24 ms 24 ms 216.239.48.4 12 48 ms 46 ms 46 ms 72.14.237.213 13 50 ms 50 ms 50 ms 72.14.237.214 14 48 ms 48 ms 48 ms 64.233.174.137

15 47 ms 47 ms 46 ms dfw06s32-in-f19.1e100.net [74.125.227.179] Trace complete.

For all additional options of traceroute, check the manual page in the terminal with the man command:man traceroute

#### WHO IS:

**WHOIS** (pronounced "**who is**"; not an acronym) is a query/response protocol which is widely used for querying an official database in order to determine the owner of a domain name, an IP address, or an autonomous system number on the Internet. WHOIS lookups were traditionally made using a command line interface, but a number of simplified web-based tools now exist for looking up domain ownership details from different databases. WHOIS normally runs on TCP port 43.

The WHOIS system originated as a method that system administrators could use to look up information to contact other IP address or domain name administrators (almost like "white pages").

whois 216.58.206.46

The following results may also be obtained via:

https://whois.arin.net/rest/nets;q=216.58.206.46?showDetails=true&showARIN=false&showNo nArinTopLevelNet=false&ext=netref2

.192.0 - 216.58.223.255	
•	.192.0 - 216.58.223.255

CIDR: 216.58.192.0/19

NetName: GOOGLE

NetHandle: NET-216-58-192-0-1

Parent: NET216 (NET-216-0-0-0)

NetType: Direct Allocation

OriginAS: AS15169

Organization: Google LLC (GOGL)

RegDate: 2012-01-27

Updated: 2012-01-27

Ref: https://whois.arin.net/rest/net/NET-216-58-192-0-1

OrgName: Google LLC

OrgId: GOGL

Address: 1600 Amphitheatre Parkway

City: Mountain View

StateProv: CA

PostalCode: 94043

Country: US

RegDate: 2000-03-30

Updated: 2017-12-21

Ref: <u>https://whois.arin.net/rest/org/GOGL</u> ...

whois google.com

Domain Name: GOOGLE.COM

Registry Domain ID: 2138514\_DOMAIN\_COM-VRSN

Registrar WHOIS Server: whois.markmonitor.com

Registrar URL: http://www.markmonitor.com

Updated Date: 2011-07-20T16:55:31Z

Creation Date: 1997-09-15T04:00:00Z

Registry Expiry Date: 2020-09-14T04:00:00Z

Registrar: MarkMonitor Inc.

Registrar IANA ID: 292

Registrar Abuse Contact Email: abusecomplaints@markmonitor.com

Registrar Abuse Contact Phone: +1.2083895740

Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited

Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited

Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited

Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited

Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited

Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited

Name Server: NS1.GOOGLE.COM

Name Server: NS2.GOOGLE.COM

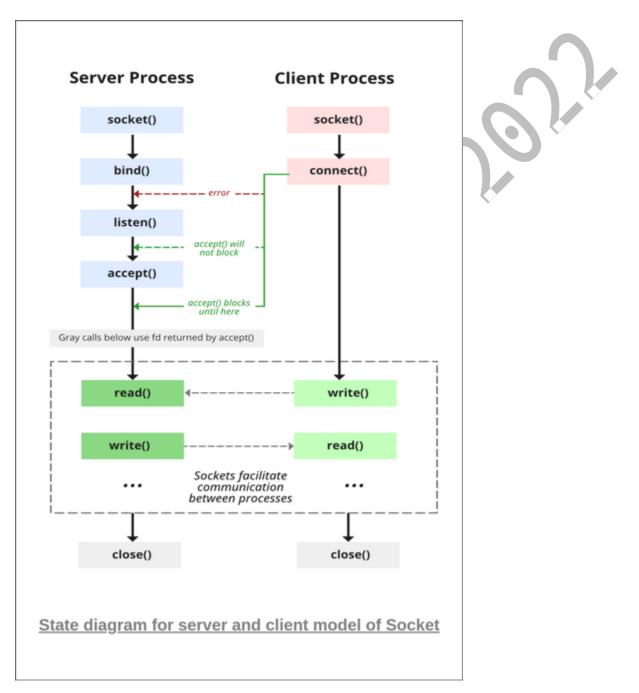
Name Server: NS3.GOOGLE.COM

Name Server: NS4.GOOGLE.COM

# PROGRAM 2 - SOCKET PROGRAMMING IN C/C++

#### What is socket programming?

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.



#### **Stages for server**

#### 1. Socket creation:

int sockfd = socket(domain, type, protocol)

- **sockfd:** socket descriptor, an integer (like a file-handle)
- **domain:** integer, specifies communication domain. We use AF\_ LOCAL as defined in the POSIX standard for communication between processes on the same host. For communicating between processes on different hosts connected by IPV4, we use AF\_INET and AF\_I NET 6 for processes connected by IPV6.
- **type:** communication type SOCK\_STREAM: TCP(reliable, connection oriented) SOCK\_DGRAM: UDP(unreliable, connectionless)
- **protocol:** Protocol value for Internet Protocol(IP), which is 0. This is the same number which appears on protocol field in the IP header of a packet.(man protocols for more details)

**2. Setsockopt:** This helps in manipulating options for the socket referred by the file descriptor sockfd. This is completely optional, but it helps in reuse of address and port. Prevents error such as: "address already in use".

int setsockopt(int sockfd, int level, int optname, const void \*optval, socklen\_t optlen);

#### 3. Bind:

int bind(int sockfd, const struct sockaddr \*addr, socklen\_t addrlen);

After creation of the socket, bind function binds the socket to the address and port number specified in addr(custom data structure). In the example code, we bind the server to the localhost, hence we use INADDR\_ANY to specify the IP address.

# 4. Listen:

int listen(int sockfd, int backlog);

It puts the server socket in a passive mode, where it waits for the client to approach the server to make a connection. The backlog, defines the maximum length to which the queue of pending connections for sockfd may grow. If a connection request arrives when the queue is full, the client may receive an error with an indication of ECONNREFUSED.

# 5. Accept:

int new\_socket= accept(int sockfd, struct sockaddr \*addr, socklen\_t \*addrlen); It extracts the first connection request on the queue of pending connections for the listening socket, sockfd,

creates a new connected socket, and returns a new file descriptor referring to that socket. At this point, connection is established between client and server, and they are ready to transfer data.

#### **Stages for Client**

- Socket connection: Exactly same as that of server's socket creation
- **Connect:** The connect() system call connects the socket referred to by the file descriptor sockfd to the address specified by addr. Server's address and port is specified in addr.

int connect(int sockfd, const struct sockaddr \*addr, socklen\_t addrlen);

#### Implementation

Here we are exchanging one hello message between server and client to demonstrate the client/server model.

• Server.c

// Server side C/C++ program to demonstrate Socket

// programming

#include <netinet/in.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <unistd.h>

#define PORT 8080

int main(int argc, char const\* argv[])

{

int server\_fd, new\_socket, valread;

struct sockaddr\_in address;

int opt = 1;

int addrlen = sizeof(address);

char buffer[1024] = { 0 };

char\* hello = "Hello from server";

// Creating socket file descriptor

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0))

== 0) {

perror("socket failed");

exit(EXIT\_FAILURE);

}

// Forcefully attaching socket to the port 8080

```
if (setsockopt(server_fd, SOL_SOCKET,
```

```
SO_REUSEADDR | SO_REUSEPORT, & opt,
```

sizeof(opt))) {

perror("setsockopt");

exit(EXIT\_FAILURE);

}

```
address.sin_family = AF_INET;
address.sin_addr.s_addr = INADDR_ANY;
address.sin_port = htons(PORT);
```

// Forcefully attaching socket to the port 8080

if (bind(server\_fd, (struct sockaddr\*)&address,

sizeof(address))

< 0) {

perror("bind failed");

```
exit(EXIT_FAILURE);
```

}

```
if (listen(server_fd, 3) < 0) {
```

perror("listen");

exit(EXIT\_FAILURE);

#### }

```
if ((new_socket
```

```
= accept(server_fd, (struct sockaddr*)&address,
```

```
(socklen_t*)&addrlen))
```

< 0) {

```
perror("accept");
```

```
exit(EXIT_FAILURE);
```

```
}
```

```
valread = read(new_socket, buffer, 1024);
```

```
printf("%s\n", buffer);
```

```
send(new_socket, hello, strlen(hello), 0);
```

```
printf("Hello message sent\n");
```

return 0;

}

# client.c

// Client side C/C++ program to demonstrate Socket

// programming

#include <arpa/inet.h>

#include <stdio.h>

```
#include <string.h>
```

#include <sys/socket.h>

#include <unistd.h>

```
#define PORT 8080
```

int main(int argc, char const\* argv[])

```
{ int sock = 0, valread;
```

```
struct sockaddr_in serv_addr;
```

```
char* hello = "Hello from client";
```

```
char buffer[1024] = { 0 };
```

```
if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
```

```
printf("\n Socket creation error \n");
```

return -1;

```
}
```

```
serv_addr.sin_family = AF_INET;
```

```
serv_addr.sin_port = htons(PORT);
```

```
// Convert IPv4 and IPv6 addresses from text to binary
```

// form

```
if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)
```

<=0) {

printf(

"\nInvalid address/ Address not supported \n");

```
return -1;
```

```
}
```

if (connect(sock, (struct sockaddr\*)&serv\_addr,

```
sizeof(serv_addr))
```

```
< 0) {
    printf("\nConnection Failed \n");
    return -1;
}
send(sock, hello, strlen(hello), 0);
printf("Hello message sent\n");
valread = read(sock, buffer, 1024);
```

printf("%s\n", buffer); return 0;}

# **Compiling:**

gcc client.c -o client gcc server.c -o server

# **Output:**

Client:Hello message sent Hello from server Server:Hello from client Hello message sent

# PROGRAM 3 - TCP CHAT SERVER/CLIENT

#### Aim

To implement a chat server and client in java using TCP sockets.

#### Algorithm

#### <u>Server</u>

- 1. Create a server socket
- 2. Wait for client to be connected.
- 3. Read Client's message and display it
- 4. Get a message from user and send it to client
- 5. Repeat steps 3-4 until the client sends "end"
- 6. Close all streams
- 7. Close the server and client socket
- 8. Stop

# <u>Client</u>

- 1. Create a client socket and establish connection with the server
- 2. Get a message from user and send it to server
- 3. Read server's response and display it
- 4. Repeat steps 2-3 until chat is terminated with "end" message
- 5. Close all input/output streams
- 6. Close the client socket

7. Stop

# Program

# // TCP Chat Server--tcpchatserver.java

import java.io.\*;

import java.net.\*;

class tcpchatserver

{

public static void main(String args[])throws Exception

{

#### PrintWriter toClient;

BufferedReader fromUser, fromClient;

try

```
ServerSocket Srv = new ServerSocket(5555);
System.out.print("\nServer started\n");
Socket Clt = Srv.accept();
System.out.println("Client connected");
```

```
toClient = new PrintWriter(new BufferedWriter(new
OutputStreamWriter(Clt.getOutputStream())), true);
fromClient = new BufferedReader(new
InputStreamReader(Clt.getInputStream()));
fromUser = new BufferedReader(new
InputStreamReader(System.in));
String CltMsg, SrvMsg;
while(true)
{
```

Department of Computer Science and Engineering,

```
CltMsg= fromClient.readLine();
if(CltMsg.equals("end"))
break:
else
{
System.out.println("\nServer <<< " +</pre>
CltMsg);
System.out.print("Message to Client : ");
SrvMsg = fromUser.readLine();
toClient.println(SrvMsg);
}}
System.out.println("\nClient Disconnected");
fromClient.close();
toClient.close();
fromUser.close();
Clt.close();
Srv.close();
} catch (
Exception E)
System.out.println(E.getMessage());
}}
// TCP Chat Client--tcpchatclient.java
import java.io.*;
import java.net.*;
class tcpchatclient
public static void main(String args[])throws Exception
ł
Socket Clt;
PrintWriter toServer;
BufferedReader fromUser, fromServer;
try
{ if (
args.length >
1)
System.out.println("Usage: java hostipaddr");
System.exit(-1);
if (args.length == 0)
Clt = new Socket(InetAddress.getLocalHost(),5555);
else
Clt = new Socket(InetAddress.getByName(args[0]),
5555);
toServer = new PrintWriter(new BufferedWriter(new
```

OutputStreamWriter(Clt.getOutputStream())), true); fromServer = new BufferedReader(new InputStreamReader(Clt.getInputStream())); fromUser = new BufferedReader(new InputStreamReader(System.in)); String CltMsg, SrvMsg; System.out.println("Type \"end\" to Quit"); while (true) System.out.print("\nMessage to Server : "); CltMsg = fromUser.readLine(); toServer.println(CltMsg); if (CltMsg.equals("end")) break; SrvMsg = fromServer.readLine(); System.out.println("Client <<< " + SrvMsg);</pre> } } catch(Exception E) System.out.println(E.getMessage()); }}} Output Server Console \$ javac tcpchatserver.java \$ java tcpchatserver Server started Client connected Server <<< hi Message to Client : hello Server <<< how r u? Message to Client : fine Server <<< me too Message to Client : bye **Client Disconnected** Client Console \$ javac tcpchatclient.java \$ java tcpchatclient Type "end" to Quit Message to Server : hi Client <<< hello Message to Server : how r u? Client <<< fine Message to Server : me too Client <<< bye Message to Server : end Result

Thus both the client and server exchange data using TCP socket programming

# PROGRAM 4 - UDP CHAT SERVER/CLIENT

#### Aim

# To implement a chat server and client in java using UDP sockets. Algorithm

#### <u>Server</u>

- 1. Create two ports, server port and client port
- 2. Create a datagram socket and bind it to client port
- 3. Create a datagram packet to receive client message
- 4. Wait for client's data and accept it.
- 5. Read Client's message
- 6. Get data from user
- 7. Construct a datagram packet and send message through server port
- 8. Repeat steps 3-7 until the client has something to send
- 9. Close the server socket

10. Stop

#### **Client**

- 1. Create two ports, server port and client port
- 2. Create a datagram socket and bind it to server port
- 3. Get data from user
- 4. Create a datagram packet and send data with server ip address and client port
- 5. Create a datagram packet to receive server message
- 6. Read server's response and display it
- 7. Repeat steps 3-6 until there is some text to send
- 8. Close the client socket
- 9. Stop

# Program

# // UDP Chat Server--udpchatserver.java

import java.io.\*;

import java.net.\*;

class udpchatserver

{

public static int clientport = 8040, serverport = 8050;

public static void main(String args[]) throws Exception

{

```
DatagramSocket SrvSoc = new DatagramSocket(clientport);
byte[] SData = new byte[1024];
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
System.out.println("Server Ready");
while (true)
{
byte[] RData = new byte[1024];
DatagramPacket RPack = new DatagramPacket(RData,
RData.length);
SrvSoc.receive(RPack);
```

```
String Text = new String(RPack.getData());
if (\text{Text.trim}().\text{length}() == 0)
break:
System.out.println("\nFrom Client <<< " + Text );</pre>
System.out.print("Msg to Cleint : " );
String srvmsg = br.readLine();
InetAddress IPAddr = RPack.getAddress();
SData = srvmsg.getBytes();
DatagramPacket SPack = new DatagramPacket(SData,
SData.length, IPAddr, serverport);
SrvSoc.send(SPack);
System.out.println("\nClient Quits\n");
SrvSoc.close();
}}
// UDP Chat Client--udpchatclient.java
import java.io.*;
import java.net.*;
class udpchatclient
{
public static int clientport = 8040, serverport = 8050;
public static void main(String args[]) throws Exception
{
BufferedReader br = new BufferedReader(new
InputStreamReader (System.in));
DatagramSocket CliSoc = new DatagramSocket(serverport);
InetAddress IPAddr:
String Text;
if (args.length == 0)
IPAddr = InetAddress.getLocalHost();
else
IPAddr = InetAddress.getByName(args[0]);
byte[] SData = new byte[1024];
System.out.println("Press Enter without text to quit");
while (true)
{
System.out.print("\nEnter text for server : ");
Text = br.readLine();
SData = Text.getBytes();
DatagramPacket SPack = new DatagramPacket(SData,
SData.length, IPAddr, clientport );
CliSoc.send(SPack);
if (\text{Text.trim}().\text{length}() == 0)
break;
byte[] RData = new byte[1024];
DatagramPacket RPack = new DatagramPacket(RData,
```

RData.length); CliSoc.receive(RPack); String Echo = new String(RPack.getData()) ; Echo = Echo.trim(); System.out.println("From Server <<< " + Echo); } CliSoc.close();

}}

# Output

Server Console \$ javac udpchatserver.java \$ java udpchatserver Server Ready From Client <<< are u the SERVER Msg to Cleint : yes From Client <<< what do u have to serve Msg to Cleint : no eatables **Client Quits Client Console** \$ javac udpchatclient.java \$ java udpchatclient Press Enter without text to quit Enter text for server : are u the SERVER From Server <<< yes Enter text for server : what do u have to serve From Server <<< no eatables Enter text for server : Result Thus both the client and server exchange data using UDP sockets.

#### PROGRAM 5 - IMPLEMENTATION OF SLIDING WINDOW PROTOCOL

#### AIM:

To write a java program to perform sliding window protocol **ALGORITHM:** 1.Start the program. 2.Get the frame size from the user 3.To create the frame based on the user request. 4.To send frames to server from the client side. 5. If your frames reach the server it will send ACK signal to client otherwise it will send NACK signal to client. 6.Stop the program **Program**: import java.net.\*; import java.io.\*; importjava.rmi.\*; public class slidsender public static void main(String a[])throws Exception ServerSocket ser=new ServerSocket(10); Socket s=ser.accept(); DataInputStream in=new DataInputStream(System.in); DataInputStream in1=newDataInputStream(s.getInputStream()); String sbuff[]=new String[8]; PrintStream p; int sptr=0,sws=8,nf,ano,i; String ch; do { p=new PrintStream(s.getOutputStream()); System.out.print("Enter theno. of frames : "); nf=Integer.parseInt(in.readLine()); p.println(nf); if(nf<=sws-1) System.out.println("Enter "+nf+" Messages to be send\n"); for(i=1;i<=nf;i++) { sbuff[sptr]=in.readLine(); p.println(sbuff[sptr]); sptr=++sptr%8; } sws-=nf; System.out.print("Acknowledgment received"); ano=Integer.parseInt(in1.readLine()); System.out.println(" for "+ano+" frames"); sws+=nf; } else

```
System.out.println("The no. of frames exceeds window size");
break;
System.out.print("\nDo you wants to send some more frames : ");
ch=in.readLine();
p.println(ch);
while(ch.equals("yes"));
s.close();
}
RECEIVER PROGRAM
import java.net.*;
import java.io.*;
class slidreceiver
public static void main(String a[])throws Exception
Socket s=new Socket(InetAddress.getLocalHost(),10);
DataInputStream in=newDataInputStream(s.getInputStream())
PrintStream p=new PrintStream(s.getOutputStream());
int i=0,rptr=-1,nf,rws=8;
String rbuf[]=new String[8]; String ch;
System.out.println();
do
nf=Integer.parseInt(in.readLine());
if(nf<=rws-1)
for(i=1;i \le nf;i++)
rptr=++rptr%8;
rbuf[rptr]=in.readLine();
System.out.println("The received Frame " +rptr+" is : "+rbuf[rptr]);
} rws-=nf;
System.out.println("\nAcknowledgmentsent\n");
p.println(rptr+1);
rws+=nf;
}
else
break;
ch=in.readLine();
}
while(ch.equals("yes"));
}
```

# **OUTPUT:** //SENDER OUTPUT

Enter the no. of frames : 4 Enter 4 Messages to be send hiii how r u i am fine how is evryone Acknowledgment received for 4 frames Do you wants to send some more frames : no //RECEIVER OUTPUT The received Frame 0 is : hiii The received Frame 1 is : how r u The received Frame 2 is : i am fine The received Frame 3 is : how is everyone

**Result:** Sliding window is implemented.

#### PROGRAM 6 - DISTANCE-VECTOR ROUTING ( DVR ) ALGORITHM

```
import java.io.*;
public class DVR
ł
static int graph[][];
static int via[][];
static int rt[][];
static int v;
static int e;
public static void main(String args[]) throws IOException
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Please enter the number of Vertices: ");
 v = Integer.parseInt(br.readLine());
 System.out.println("Please enter the number of Edges: ");
 e = Integer.parseInt(br.readLine());
 graph = new int[v][v];
 via = new int[v][v];
 rt = new int[v][v];
 for(int i = 0; i < v; i++)
 for(int j = 0; j < v; j++)
  if(i == j)
   graph[i][j] = 0;
  else
   graph[i][j] = 9999;
  }
 for(int i = 0; i < e; i++)
 System.out.println("Please enter data for Edge "+(i + 1) +":");
 System.out.print("Source: ");
 int s = Integer.parseInt(br.readLine());
 s--;
 System.out.print("Destination: ");
 int d = Integer.parseInt(br.readLine());
 d--;
 System.out.print("Cost: ");
 int c = Integer.parseInt(br.readLine());
 graph[s][d] = c;
 graph[d][s] = c;
 }
```

```
dvr_calc_disp("The initial Routing Tables are: ");
```

```
System.out.print("Please enter the Source Node for the edge whose cost has changed: ");
int s = Integer.parseInt(br.readLine());
s--;
System.out.print("Please enter the Destination Node for the edge whose cost has changed: ");
int d = Integer.parseInt(br.readLine());
d--;
System.out.print("Please enter the new cost: ");
int c = Integer.parseInt(br.readLine());
graph[s][d] = c;
graph[d][s] = c;
dvr_calc_disp("The new Routing Tables are: ");
ł
static void dvr_calc_disp(String message)
System.out.println();
init_tables();
update tables();
System.out.println(message);
print_tables();
System.out.println();
}
static void update_table(int source)
for(int i = 0; i < v; i++)
 if(graph[source][i] != 9999)
 int dist = graph[source][i];
 for(int j = 0; j < v; j++)
  int inter_dist = rt[i][j];
  if(via[i][j] == source)
  inter_dist = 9999;
  if(dist + inter dist < rt[source][j])
  {
   rt[source][j] = dist + inter_dist;
   via[source][j] = i;
  }
  }
 }
```

```
}
}
static void update_tables()
{
int k = 0;
for(int i = 0; i < 4*v; i++)
{
 update_table(k);
 k++;
 if(k == v)
 k = 0;
 }
ł
static void init_tables()
{
for(int i = 0; i < v; i++)
 ł
 for(int j = 0; j < v; j++)
 ł
 if(i == j)
  rt[i][j] = 0;
  via[i][j] = i;
  }
  else
  {
  rt[i][j] = 9999;
  via[i][j] = 100;
  }
 }
static void print_tables()
{
for(int i = 0; i < v; i++)
 for(int j = 0; j < v; j++)
 ł
 System.out.print("Dist: " + rt[i][j] + " ");
 System.out.println();
 }
}}
```

```
PROGRAM 7 – SMTP
Aim: Program for Simulating SMTP Client.
Algorithm:
Step1: Create a client socket to the SMTP port(25)
Step2: Extract the first 3 letters of the response code.
Step3: If it is 421 close the connection.
Step4: If it is 250 then send "HELLO Server name" to the server.
Step5: If it is 250 then send Mail from sender mail" to the server.
Step6: If it is250 then send "RCPT to: receipent_e-mail" to the server
Step7: If it is 250 then send "DATA" to server
Step8: If 354 then send Multi line message with header of the mail.
Step9: For completing the message send "." on a single line.
Step10: This sends the mail to the server.
Step11: Close The connection to the server.
/*Program for Simulating SMPT Client*/
import java.io.*;
import java.net.*;
import java.util.*;
import java.lang.*;
public class smtp
Socket s;
DataInputStream din, in;
DataOutputStream out;
PrintWriter pw;
public static void main(String args[]) throws Exception
{ smtp ob = new smtp();
ł
public smtp()
try
{
s = new Socket("localhost", 25);
din = new DataInputStream(System.in);
in = new DataInputStream(s.getInputStream());
out = new DataOutputStream(s.getOutputStream());
pw = new PrintWriter(out);
String data;
checkfor(true, 220);
System.out.println("hello localhost");
System.out.println("Enter from address:");
data = din.readLine();
send("mail from :" + data + "@192.168.1.6");
checkfor(true, 250);
System.out.println("Enter the destination address:");
data = din.readLine();
```

```
send("Rcpt to:" + data + "@192.168.1.6");
checkfor(true, 250);
send("DATA");
checkfor(true, 354);
System.out.println(" Enter content: terminate line with .");
do {
data = din.readLine();
send(data); }
while (!(data.equals(".")));
checkfor(true, 250);
System.out.println("Exiting successfully");
} catch (Exception e) {
System.out.println(e.getMessage()); }
public void checkfor(boolean b, int replycode)
String reply;
try {
reply = in.readLine();
System.out.println(reply);
int r = Integer.parseInt(reply.substring(0, 3));
if (!((b && r == replycode) || (!b && r != replycode)))
{ System.out.println("error occured " + r + "expend" + replycode);
System.exit(0);
} else
{
return;
} }
catch (Exception e)
{ System.out.println("Error is reply"); }
} public void send(String msg)
{ pw.println(msg);
pw.flush();
return; }
}
output:
[cs462@cseoracleserver java]$ vi smtp.java
[cs462@cseoracleserver java]$ javac smtp.java
Note: smtp.java uses or overrides a deprecated API.
Note: Recompile with -deprecation for details.
[cs489@cseoracleserver java]$ java smtp
220 localhost.localdomain ESMTP Sendmail 8.13.1/8.13.1; Thu, 2 Feb 2012 10:56:54
+0530
hello localhost
Enter from address:
Pranaya
```

250 2.1.0 pranaya@192.168.1.6... Sender ok
Enter the destination address:
ramya
250 2.1.5 ramya@192.168.1.6... Recipient ok (will queue)
354 Enter mail, end with "." on a line by itself
Enter content: terminate line with .
hai how are you. .
250 2.0.0 q125QsMU011543 Message accepted for delivery
Exiting successfully

**Result :** SMTP is implemented successfully

#### PROGRAM 8 – FTP

Aim :To write a java program to perform "File Transfer Protocol"

# .ALGORITHM

#### SERVER SIDE

1.Import the java packages and create class fileserver.

2.String of argument is passed to the args[].

3.Create a new server socket and bind it to the port.

4. Accept the client connection at the requested port.

5.Get the filename and stored into the BufferedReader.

6.Create a new object class file and readline.

7.If File is exists then FileReader read the content until EOF is reached.

8.Else Print FileName does't exits.

9.End of main.

10.End of FileServer class.

#### **CLIENT SIDE**

1.Import the java packages and create class fileClient.

2.String of argument is passed to the args[].

3. The connection between the client and server is successfully established.

4. The object of a BufferReader class is used for storing data content which have beenretrieved from socket object s.

5. The content are read and stored in inp until the EOF is reached.

6. The content of file are displayed in displayed in client window and the connection is closed.

7.End of main.

8.End of Fileclient class.

# **IMPLEMENTATION OF FTP SOURCE CODE - SERVER**

import java.io.\*; import java.net.\*; public class fileserver1

{ public static void main(String args[])throws IOException

r •••

ServerSocket s1=null;

try{s1=new ServerSocket(1187);

```
}catch(IOException u1)
```

ł

System.out.println("could not found port 1187");

```
System.exit(1);
```

}

```
Socket c=null;
try
```

```
{c=s1.accept();
```

```
System.out.println("connection frame" +c);
```

```
}catch(IOException e)
```

```
{
```

```
System.out.println("accept failed");
System.exit(1);
ł
PrintWriter out=new PrintWriter(c.getOutputStream(),true);
BufferedReader sin=new BufferedReader(new InputStreamReader(System.in));
System.out.println("enter the text file name");
String s=sin.readLine();
File f=new File(s);
if(f.exists())
{
BufferedReader in=new BufferedReader(new FileReader(s));
String v;while((v=in.readLine())!=null)
out.write(v);
out.flush();
System.out.println("the file send successfully");
in.close();
c.close();
s1.close();
}}
}
CLIENT
import java.io.*;
import java.net.*;
public class fileclient1
public static void main(String args[])throws IOException
ł
Socket s=null;
BufferedReader b=null;
try
{
s=new Socket(InetAddress.getLocalHost(),1187);
b=new BufferedReader(new InputStreamReader(s.getInputStream()));
}catch(Exception u)
{
System.out.println("the file is received");
System.out.println("don't know host");
System.exit(1);
String inp;
while((inp=b.readLine())!=null)
System.out.println("the content of the file is");
System.out.println(inp);
```

System.out.println("the file is received successfully"); b.close(); s.close(); } ł **OUTPUT:SERVER** C:\Documents and Settings\SEENU.R>cd\ C:\>cd C:\Program Files\Java\jdk1.6.0\bin C:\Program Files\Java\jdk1.6.0\bin>javac fileserver1.java C:\Program Files\Java\jdk1.6.0\bin>java fileserver1 connection frameSocket[addr=/127.0.0.1,port=1056,localport=1187] enter the text file name HAI.txt the file send successfully C:\Program Files\Java\jdk1.6.0\bin> C:\Documents and Settings\SEENU.R> C:\>cd C:\Program Files\Java\jdk1.6.0\bin C:\Program Files\Java\jdk1.6.0\bin>javac fileclient1.java C:\Program Files\Java\jdk1.6.0\bin>java fileclient1 the content of the file is GOD LOVE'S EVERY ONE IN THE WORLD. the file is received successfully C:\Program Files\Java\jdk1.6.0\bin>

## **RESULT:**

Thus the output for the java program is executed and verified successfully

# PROGRAM 9 – CONGESTION CONTROL USING A LEAKY BUCKET ALGORITHM

```
import java.util.*;
import java.io.*;
class Leaky
ł
  public static void main (String args[]) {
  int n,sto,bs,ips,ops,sl;
  Scanner sc = new Scanner(System.in);
//Initial total number of packets that are passing into the bucket
  System.out.println("Enter total number of quries entering\n"):
  n=sc.nextInt();
//first estimate total packets in the bucket
  System.out.println("Enter the total number of packets in the bucket n"):
  sto=sc.nextInt();
//the total number of packets that can be filled in the bucket
  System.out.println("total number of packets that can be filled in the bucket n");
  bs=sc.nextInt();
 //number of input packets at a particular time
  System.out.println("total number of packets that are inputed into the bucket n");
  ips=sc.nextInt();
//no. of packets that quit the bucket
  System.out.println("total number of packets that are comming out the bucket n");
  ops=sc.nextInt();
  for(int i=0;i<n;i++)</pre>
     System.out.println("At time:\n''+(i+1));
// total size left
     sl=bs-sto;
     System.out.println("size left "+sl);
     if(ips<=(sl))
     {
          sto+=ips:
          System.out.println("Packets received= "+sto+ " out of bucket size= "+bs);
          System.out.println("Packet loss is 0 ");
     }
     else
          System.out.println("Packet loss = "+(ips-(sl)));
//if the bucket is at filled fully
          sto=bs:
          System.out.println("Buffer size= "+sto+ " out of bucket size= "+bs);
     }
     System.out.println("output sent "+ops);
     sto-=ops;
     System.out.println("After output sent storage is "+sto);
     System.out.println("------"); } }
```

# PROGRAM 10 – WIRESHARK TOOL

What Is Wireshark?

Wireshark is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet. Packet is the name given to a discrete unit of data in a typical Ethernet network.

Wireshark is the most often-used packet sniffer in the world. Like any other packet sniffer, Wireshark does three things:

- 1. **Packet Capture:** Wireshark listens to a network connection in real time and then grabs entire streams of traffic quite possibly tens of thousands of packets at a time.
- 2. **Filtering:** Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
- 3. **Visualization:** Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.

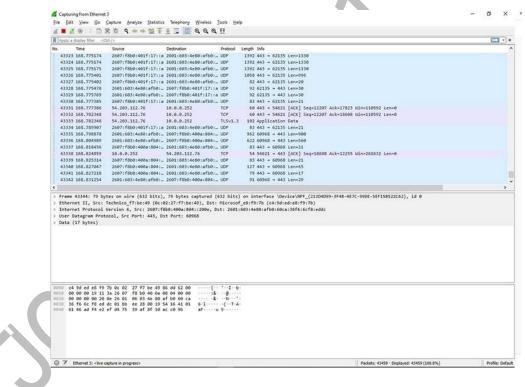


Figure 1: Viewing a packet capture in Wireshark

Packet sniffing can be compared to spelunking – going inside a cave and hiking around. Folks who use Wireshark on a network are kind of like those who use flashlights to see what cool things they can find. After all, when using Wireshark on a network connection (or a flashlight in a cave), you're effectively using a tool to hunt around tunnels and tubes to see what you can see.

What Is Wireshark Used For?

Wireshark has many uses, including <u>troubleshooting networks</u> that have performance issues. Cybersecurity professionals often use Wireshark to trace connections, view the contents of suspect network transactions and identify bursts of network traffic. It's a major part of any IT pro's toolkit – and hopefully, the IT pro has the knowledge to use it.

When Should Wireshark Be Used?

Wireshark is a safe tool used by government agencies, educational institutions, corporations, small businesses and nonprofits alike to troubleshoot network issues. Additionally, Wireshark can be used as a learning tool.

Those new to information security can use Wireshark as a tool to understand network traffic analysis, how communication takes place when particular protocols are involved and where it goes wrong when certain issues occur.

Of course, Wireshark can't do everything.

First of all, it can't help a user who has little understanding of network protocols. No tool, no matter how cool, replaces knowledge very well. In other words, to properly use Wireshark, you need to learn exactly how a network operates. That means, you need to understand things such as the three-way TCP handshake and various protocols, including TCP, UDP, DHCP and ICMP.

Second, Wireshark can't grab traffic from all of the other systems on the network under normal circumstances. On modern networks that use devices called switches, Wireshark (or any other standard packet-capturing tool) can only sniff traffic between your local computer and the remote system it is talking to.

Third, while Wireshark can show malformed packets and apply color coding, it doesn't have actual alerts; Wireshark isn't an intrusion detection system (IDS).

Fourth, Wireshark can't help with decryption with regards to encrypted traffic.

And finally, it is quite easy to spoof IPv4 packets. Wireshark can't really tell you if a particular IP address it finds in a captured packet is a real one or not. That requires a bit more know-how on the part of an IT pro, as well as additional software.

Common Wireshark Use Cases

Here's a common example of how a Wireshark capture can assist in identifying a problem. The figure below shows an issue on a home network, where the internet connection was very slow.

As the figure shows, the router thought a common destination was unreachable. This was discovered by drilling down into the IPv6 Internet Message Control Protocol (ICMP) traffic, which

is marked in black. In Wireshark, any packet marked in black is considered to reflect some sort of issue.

	Mu/le				
Time	Source	Destination	Research	Length 3Hs	
48490 736,620308	13.107.136.9	10.0.0.252	TCP	60 443 + 57115 [ACK] Seg-56920 Ack+71089 Min-1020 Len-0	
48491 736,626839	13,107,136,9	10.0.0.252	TCP	60 443 - 57115 [ACK] Sequ-56920 Ack+71472 kim-1026 Lenv0	
48492 736,697628	13,107,136,9	10.0.0.252		1330 Application Data	
48493 736,737889	10.0.0.252	13,107,136,9	TCP	54 57115 + 443 [ACK] Seq=71472 Ack+58195 Min+1029 Len+0	
48494 737,106440	2601:603:4e80:afb0:	2001:558:feed:11	DNS	96 Standard overv Bx16a4 A woad, comptia.org	
48495 737,186664	2601:603:4e80:afb81.	2001:558:feed::1	DNS.	95 Standard Query Bxbd9a AAAA woad.comptia.org	
48496 737.137846	2601:603:4e88:afb8:	2001:558:feed::2	ONS	96 Standard Query Bxbd9a AAAA upad.comptis.org	
48497 737.137847	2601:603:4e80:afb0:	2001:558:feed::2	ONS	96 Standard guery 0x16a4 A wpad.coeptis.org	
48498 737.142095	2001:558:feed::1	2601:603:4e80:afb0:_	DNS	151 Standard guery response 0x16a4 No such name A upad.comptia.org SOA haven	
48499 737.142447	2601:603:4e88:afb8:.	2001:558:feed::1	DNS	104 Standard guery 0x502f A wpad.hsdl.wa.comcast.net	
48500 737.142618	2601:603:4e80:afb0:	2001:558:feed::1	DNS	104 Standard guery 0x82ba AAAA wpad.hsdl.wa.comcast.net	
48501 737.143251	2001:558:feed::1	2601:603:4e80:afb0:_	DNS	151 Standard query response Bxbd9a No such name AAAA wpad.comptim.org 50A haven	
48502 737.143284	2601:603:4e881afb8:_	20011558:feed:11	109996	199 Destination Unreachable (Port unreachable)	
40503 737.161361	2001:558:feed:11	2601:603:4e80:afb01_	DNS	156 Standard query response 0x502f No such name A wpad.hsdl.wa.comcast.net 50A dhs101.comcast.net	
40504 737.166361	2001:558:feed::1	2601:603:4e80:afb0:_		156 Standard guery response 0x82ba No such name AAAA wpad.hsdl.wa.comcast.met SOA dns101.comcast.met	
48585 737.181369	2001:558:feed::2	2601:603:4e80:afb0:_		151 Standard query response @xbd9a No such name AAAA wpad.comptis.org 50A haven	
48586 737.181456	2601160314e801afb01		109946	199 Destination Unreachable (Port unreachable)	
48587 737.182435	2001:558:feed::2	2601:603:4e80:afb0:_		151 Standard query response 0x16a6 No such name A spad.comptia.org SGA haven	
48588 739,468531	fe80::902e:6cff:fea.		ICHPv6	174 Router Advertisement from 0c:02:27:f7:be:49	
48589 739,917615	Technico_f7:be:49	Microsof_e81f917b	ARP		
				60 Who has 10.0.0.252? Tell 10.0.0.1	
48510 739.917651 Frame 48492: 1330 1	Microsof_e8:f9:7b bytes on wire (10648 b	Technico_f7:be:49 its), 1330 bytes capto	ARP ured (1064	42 10.0.0/252 is at c4:9d:ed:e8:19:75 40 bits) on interface \Device\NPF_(21304099-3F40-4E7C-990E-SEF158522C62), 1d 0	
48510 739.017651 Frame 48402: 1330 f Ethernet II, Src: Internet Protocol U Transpission Contro Transport Layer Ser v TLSV1.2 Record L Content Type: Version: TLS Length: 1271	Microsof_e8:f9:70 bytes on wire (10648 b Technico_f7:be:49 (&c: Version 4, Src: 13.107 ol Protecel, Src Port: curity ayer: Application Dats (23)	Technico_f7:be:49 its), 1330 bytes capt. 02:27:f7:be:40), Dst: 136:9, Ost: 10.0.0.2; 443, Dst Port: 57115 a Protocol: http-over-	ARP ared (1064 Ricrosof, 52 , Seq: 540 tls	42 186.0.352 is at childred/e819920 68 Dits) on interface UbriceWMF_(21304099-3F48-4E7C-990E-5EF158522063), is 0 effettre (childred/e8199276) 920, Ack: 71472, len: 1276	=

Figure 2: Drilling down into a packet to identify a network problem using Wireshark

In this case, Wireshark helped determine that the router wasn't working properly and couldn't find YouTube very easily. The problem was resolved by restarting the cable modem. Of course, while this particular problem didn't necessitate using Wireshark, it's kind of cool to authoritatively finalize the issue.

When you take another look at the bottom of Figure 2, you can see that a specific packet is highlighted. This shows the innards of a TCP packet that is part of a transport layer security (TLS) conversation. This is a great example of how you can drill down into the captured packet.

Using Wireshark doesn't allow you to read the encrypted contents of the packet, but you can identify the version of TLS the browser and YouTube are using to encrypt things. Interestingly enough, the encryption shifted to TLS version 1.2 during the listening.

Wireshark is often used to identify more complex network issues. For example, if a network experiences too many retransmissions, congestion can occur. By using Wireshark, you can identify specific retransmission issues, as shown below in Figure 3.

Time	192.16	3.56.102 8.8.	1.8.8	105.162.147.224 Comme	ent	
.000000 .647439 .809306 .210133 .702100 .708095 .178631 .662885 .804235 .694902 .925304 .804891 .022679	49276 49276 55332 55332 49243 49244 49244 49244 49244 49258 49258 49258 49272 49283	Standard query Oxf6a2 A www.lugerfsodp90ja Standard query response Oxf5a2 Server failure 49243	53 53 53 53 54 51 570] Sequit Wine8192 Lence MSS= 570] Sequit Wine8192 Lence MSS= 45264	DNS: Stand           DN	ard query 0x894 A www.iuqe ard query response 0x894 Se ard query 0x15a2 A www.iuqe ard query response 0x15a2 Se - 445 (SYN) Seq=0 Win=81 - 445 (SYN) Seq=0 Win=81	erver failure A erfsodp9ifjepo sever failure A 192 Len=0 M 192 Len=0 M 192 Len=0 M 192 Len=0 M 192 Len=0 M 192 Len=0 M
<i>sket 14: TCP: 492</i>		q=0 Win=8192 Len=0 MSS=1460 WS=256 SACK Flow t	type: All Flows		Addresses	>
	nay meet		oper rannono		Hadressee	Reset

Figure 3: Viewing packet flow statistics using Wireshark to identify retransmissions

By confirming this type of issue, you can then reconfigure the router or switch to speed up traffic.

How to Use Wireshark

You can download Wireshark for free at <u>www.wireshark.org</u>. It's also freely available, as an open source application under the <u>GNU General Public License</u> version 2.

How to Install Wireshark on Windows

If you're a Windows operating system user, download the version appropriate for your particular version. If you use Windows 10, for example, you'd grab the 64-bit Windows installer and follow the wizard to install. To install, you'll need administrator permissions.

How to Install Wireshark on Linux

If you have a <u>Linux system</u>, you'd install Wireshark using the following sequence (notice that you'll need to have root permissions):

\$ sudo apt-get install wireshark

\$ sudo dpkg-reconfigure wireshark-common

\$ sudo usermod -a -G wireshark \$USER

Department of Computer Science and Engineering,

\$ newgrp wireshark

Once you have completed the above steps, you then log out and log back in, and then start Wireshark:

\$ wireshark &

How to Capture Packets Using Wireshark

Once you've installed Wireshark, you can start grabbing network traffic. But remember: To capture any packets, you need to have proper permissions on your computer to put Wireshark into promiscuous mode.

- In a Windows system, this usually means you have administrator access.
- In a Linux system, it usually means that you have root access.

As long as you have the right permissions, you have several options to actually start the capture. Perhaps the best is to select Capture >> Options from the main window. This will bring up the Capture Interfaces window, as shown below in Figure 4.

Interface	Traffic	Link-layer Header	Promis	Snaplen	Buffer (N	Monite	Capture Filter		
Local Area Connection* 9		Ethernet		default	2				
> Bluetooth Network Connection		Ethernet		default	2				
> Local Area Connection* 1		Ethernet		default	2				
> Wi-Fi		Ethernet		default	2	_			
> VirtualBox Host-Only Network		Ethernet		default	2				
> Local Area Connection* 2		Ethernet		default	2				
> Ethernet 3		Ethernet		default	2				
Local Area Connection* 11		Ethernet		default	2	-			
Local Area Connection* 8		Ethernet		default	2				
Adapter for loopback traffic ca	pture	BSD loopback		default	2				
USBPcap1		USBPcap							
2 Enable promiscuous mode on all in								 Manage Inte	

Figure 4: The Capture Interfaces dialog in Wireshark

This window will list all available interfaces. In this case, Wireshark provides several to choose from.

For this example, we'll select the Ethernet 3 interface, which is the most active interface. Wireshark visualizes the traffic by showing a moving line, which represents the packets on the network.

Once the network interface is selected, you simply click the Start button to begin your capture. As the capture begins, it's possible to view the packets that appear on the screen, as shown in Figure 5, below.

len h	a display litter +0	265-25			C
	Time	Source	Destination	Protocol	Lenoth Info
	19 6,987999	10.0.0.252	52, 10, 0, 129	TCP	54 51676 • 443 [ACX] Seq+2 Ack+2 Win+1025 Len+0
	20 8,087128	fe80:1902e:6cff:fea.		109946	174 Router Advertisement from 0c102127:F7:be:49
	21 8,220134	18.0.0.252	192.204.15.5	TL5v1.2	139 Application Data
	22 8.310244	192.204.15.5	10.0.0.252	TLSv1.2	139 Application Data
	23 8.351287	10.0.0.252	192.204.15.5	TCP	54 57105 + 5091 [ACK] Seg-86 Ack-86 Min=62866 Len=0
	24 9.017538	10.0.0.252	54.149.94.178	TLSv1.2	112 Application Data
	25 9.067201	10.0.0.252	54.149.94.178	TL5v1.2	178 Application Data
	26 9.079258	54.149.94.178	10.0.0.252	TCP	60-443 + 57110 [ACK] Seq=1067 Ack=115 Win=25 Len=0
	27 9.091205	54.149.94.178	10.0.0.252	TCP	60-443 + 57119 [ACX] Seq=1067 Acx=239 Win=25 Len=0
	28 9.091205	54.149.94.178	10.0.0.252	TLSv1.2	198 Application Data
	29 9.132428	10.0.0.252	54,149,94,178	TCP	54 57119 + 443 (ACX) Seq=239 Ack=1211 Win=1025 Len=0
	30 9,140705	2601:603:4e80:afb0:_	2001:558:feed::1	DNS	97 Standard query 0xbf9d A edgeapi.slack.com
	31 9.140934	2601:603:4e80:afb0:	2001:558:feed:11	DNS	97 Standard query 0x4038 AAAA edgeapi.slack.com
	32 9.158384	2001:558:feed:11	2601160314e801afb01	DNS	194 Standard query response 0xbf9d A edgespi.slack.com CNAME prod-haproxy-edge-nlb-eip-la731160ba803b29.elb.us-west-2.amazonaw.
	33 9.164440	2001:558:feed::1	2601:603:4e80:afb0:	DNS	262 Standard query response 0x4038 AAAA edgeapi.slack.com CNAME prod-haproxy-edge-nlb-eip-la731160ba803b29.elb.us-west-2.anazo.
	34 9.172942	10.0.0.252	54.214.87.230	TCP.	66 51654 + 443 (51N) Sequél kinvé4240 Lenvél MSS-1460 kS-256 SACK PERM-1
	35 9,197224	54.214.87.230	18.8.8.252	TCP	66 443 - 53694 [STN, ACK] Seque Ackv1 Hinv26883 Lenve MSS+1460 SACK_PERM+1 HS+4696
	36 9.197340	10.0.0.252	54.214.87.230	TCP	54 51694 + 443 [ACX] Seq=1 Ack=1 Win=262656 Len=0
	37 9,197682	10.0.0.252	54.214.87.230	TL5v1.3	571 Client Hello
	38 9.229860	54.214.87.230	10.0.0.252	TLSv1.3	1514 Server Hello, Change Cipher Spec, Application Data
Fran Ethe Inte	ornet II, Src: ornet Protocol	Micro-St_b2:41:9e (d8: Version 4, Src: 10.0.0	cb:84:b2:41:9e), Ost: .62, Ost: 10.0.0.255		1514 443 - 51694 [ACC] Seq=1461 Ack+518 Him-28672 Lem-1460 [TCP segment of a reassembled POU] interface (Device/NOF_(21306009-3F48-4E7C-900E-5EF158522662), id 0 t (ff:ff:ff:ff:ff:ff:ff:ff)
inter Inter	re 1: 92 bytes rnnet 11, Src: rnnet Protocol EDS Name Servi ff ff ff ff ff ff ff 00 4e 6d 00 00 00 6f 00 80 00	on wire (736 bits), 92 Micro-St_b2:41:9e (d8: Version 4, Src: 10.0.0 tocol, Src Port: 137, D	bytes captured (736   cb:Baib2:45:00), Out: cb:Baib2:45:00), Out: cb:Baib2:45:00), Out: cb:Baib2:45:00), Out: cb:Baib2:45:00, O	bits) on	interface (Device(WBF_(12)00009-3F48-4E7C-990E-SEFISBS22EE3), id 0 t. (ff:ff:ff:ff:ff:ff:ff:ff).

Figure 5: Wireshark capturing packets

Once you have captured all the packets that you want, simply click the red, square button at the top. Now you have a static packet capture to investigate.

What the Color Coding Means in Wireshark

Now that you have some packets, it's time to figure out what they mean. Wireshark tries to help you identify packet types by applying common-sense color coding. The table below describes the default colors given to major packet types.

Color in Wireshark	Packet Type
Light purple	ТСР
Light blue	UDP
Black	Packets with errors
Light green	HTTP traffic

Color in Wireshark	Packet Type
Light yellow	Windows-specific traffic, including Server Message Blocks (SMB) and NetBIOS
Dark yellow	Routing
Dark gray	TCP SYN, FIN and ACK traffic

The default coloring scheme is shown below in Figure 6. You can view this by going to View >> Coloring Rules.

Name	Filter
Bad TCP	tcp.analysis.flags && itcp.analysis.window_update
HSRP State Change	hsrp.state != 8 && hsrp.state != 16
Spanning Tree Topology Change	
OSPF State Change	ospf.msg != 1
ICMP errors	icmp.type eq 3    icmp.type eq 4    icmp.type eq 5    icmp.type eq 11    icmpv6.type eq 1    icmpv6.type eq 2    icmpv6.type eq 3    icmpv6.type
ARP ARP	arp
ICMP	icmp    icmpv6
TCP RST	tcp.flags.reset eq 1
SCTP ABORT	sctp.chunk_type eq ABORT
TTL low or unexpected	( ! ip.dst == 224.0.0.0/4 && ip.ttl < 5 && lpim && lospf)    (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && !(vrrp    carp)
Checksum Errors	eth.fcs.status=="Bad"    ip.checksum.status=="Bad"    tcp.checksum.status=="Bad"    udp.checksum.status=="Bad"    sctp.checksum.status=="Bad"    tcp.checksum.status=="Bad"    tcp.checksu
SMB	smb    nbss    nbns    netbios
HTTP	http    tcp.port == 80    http2
DCERPC	dcerpc
Routing	hsrp    eigrp    ospf    bgp    cdp    vrrp    carp    gvrp    igmp    ismp
TCP SYN/FIN	tcp.flags & 0x02    tcp.flags.fin == 1
TCP	tcp
UDP UDP	udp
🗹 Broadcast	eth[0] & 1
System Event	systemd_journal    sysdig
< Couble click to edit. Drag to more. Rules are processed + - Ra Ra	in cryfer well e metch is found.

Figure 6: Default coloring rules

You can even change the defaults or apply a custom rule. If you don't want any coloring at all, go to View, then click Colorize Packet List. It's a toggle, so if you want the coloring back, simply go back and click Colorize Packet List again. It's possible, even, to colorize specific conversations between computers.

In Figure 7 below, you can see standard UDP (light blue), TCP (light purple), TCP handshake (dark gray) and routing traffic (yellow).

867 87.748037       3.1.3.6.778       18.8.6.323       11.54.7.8       85 forsysted Alert         867 87.748037       3.1.3.6.778       18.8.6.323       1.3.1.67.78       18.8.6.323       1.3.1.67.78       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       18.8.6.323       1.3.1.6.778       10.8.6.323       1.3.1.6.778       <	6673 177,16968       2661;6632;4681;4761;       184,4,672       TUS       69< 5244 - 441 Lon-28         6673 177,16968       313,6778       184,4,623       TUS       64       457 177,4978       184,4,123       113,6778       184,4,123       113,6778       184,4,123       113,6778       184,4,123       113,6778       100       45 3724 + 440 (KC) Seq.3254 (KA42) Stan.0358 (Lon-0       64       64       457 174,4444       160,23       116,0338	coly a display filter <0	35/2×		-	
8474 377.499357       3.13.6.7.78       18.8.8.122       115.4.7.8       85 Forzysted Alert         8675 327.48847       13.0.6.7.78       18.8.8.122       1.13.6.7.78       18.8.8.122       1.13.6.7.78       18.8.8.122       1.13.6.7.78       19.4 44.7 5321 [150, 452] [150,	8474 377.40037       3.1.3.477.8       10.4.0.32       TLVAL.2       15 forcepted Alert         8475 377.40037       3.1.3.477.8       10.4.0.32       3.1.3.477.8       10.4.0.32       3.1.3.477.8         8475 377.40047       10.4.0.32       3.1.3.677.8       10.4.0.32       3.1.3.677.8       10.4.0.425       3.1.3.677.8       10.4.0.32       3.1.3.677.8       10.4.0.32       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       3.1.3.677.8       10.4.0.252       10.7.0       64.3.757.9       10.4.0.252       10.3.677.8       10.4.0.252       10.7.0       64.3.757.9       10.4.0.252       10.7.0       64.3.757.9       10.4.0.252       10.7.0       64.3.757.9       10.4.0.252       10.7.0       64.3.757.9       10.4.0.25       10.7.0       10.4.0.25       10.7.0       10.4.0.25       10.7.0       10.4.0.25       10.7.0       64.3.757.9       10.4.0.252       10.1.0.7.7       10.4.0.25       10.7.0       64.3.97.7       10.4.0.25       10.7.0       64.3.97.7       10.4.0.25       10.7.0       64.3.97.7       10.4.0.25       10.7.0       64.3.97.7       10.4.0.252       10.7.0       64.3.97.7       10.4.0.252	Time	Source	Destination	Protocol	Length Ends
BYS 227-048012       1.13.4778       10.8.8.723       10.9       49.40.7.5734 [150, ACC] (50.4542 ACC) (50.5554 [150, ACC] (50.5544 [150, A	BYS 127-248412       3.13.67.78       10.8.6.22       107       64.91.7.5212 [CDL, ACL] 5 seq-46.8.45.324 [cm.0.4 Cm.0         BYS 127-248159       10.8.6.123       3.13.67.78       107       54.577.4 = 4.41 [CDL, KCL] 5 seq-35.4 KA:64.27 Xin-65084 [cm.0         BYS 127-248159       10.8.6.23       3.13.67.78       107       54.577.4 = 4.41 [CDL, KCL] 5 seq-35.4 KA:64.27 Xin-65084 [cm.0         BYS 127-248159       10.8.6.23       3.13.67.78       107       64.9 177 [LCL] 5 seq-35.4 KA:64.25 Xin-35.08       10.9.6         BYS 127-84813       10.8.6.23       3.13.67.78       107       64.9 177 [LCL] 5 seq-35.4 KA:64.25 Xin-35.08       10.9.6         BYS 127-84813       10.8.6.25       3.13.67.78       107       64.9 177 [LCL] 5 seq-35.4 KA:15.5 Xin-35.08       10.9.6         BYS 127-88433       10.8.6.25       3.13.67.78       107       64.9 177 [LCL] 5 seq-35.4 KA:15.5 Xin-35.08       10.9.6.25         BYS 127-88433       10.8.6.25       3.13.67.78       107       54.577.9 TYR       10.8.6.25       3.13.67.78       107       54.577.9 TYR       10.8.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.67.78       10.9.6.25       3.13.6.7.78		2601:603:4e88:afb8:			
60/6 27/2 44849       19.8.4.623       3.13.67.78       107       54 9714 - 441 (262) Seq234 Act6427 Minc4984 Lene#         60/6 27/2 172-6869       19.8.4.623       3.13.67.78       10.8.4.623       3.13.67.78       10.8.4.623       3.13.67.78         60/6 27/2 172-6869       3.13.67.78       10.8.4.625       170       64 43 - 9371 (263) Seq4627 Act6235 Minc494 Lene#         60/6 27/2 172-6869       3.13.67.78       10.8.4.625       170       64 43 - 9371 (263) Seq4627 Act6235 Minc494 Lene#         60/6 27/2 18127       10.8.4.625       3.13.67.78       10.6.4.625       170       64 43 - 9371 (263) Seq4628 Lene#         60/6 27/2 18127       10.8.4.625       3.13.67.78       10.6.4.825       170       64 43 - 9378 (263) Seq4628 Lene# Sex348 Min238 Seq268 Min4 Sex38 Sex328 Sex27#Min1         60/6 27/2 18127       10.8.4.6.125       10.1.6.77       107       54 9718 - 443 (261) Seq48 Min4 Sex38 Sex38 Min238 Min2488       10.0.6.255         60/6 27/2 98536       10.8.4.6.12       10.1.6.78       109       100.4.6.252 Sex38 Min4 Min448       10.0.6.255         60/6 172.969539       10.8.4.6.12       10.1.6.4.8.1.1 Se939 Seq48 Min44 Min448       10.0.6.255       10.0.6.255       10.0.6.252       10.0.6.252       10.0.6.252       10.0.6.252       10.0.6.252       10.0.6.252       10.0.6.252       10.0.6.753       10.0.6.252	00/12/7,010607       10/0,01/12       10/0,7/18       TOP       54 57/14 - 440       (CO) Seq22/4 40-6627 140-0628 (Lend         00/12/7,204060       31.0,6.7.78       10.0,0.78       TOP       64 57/14 - 440       (CO) Seq22/4 40-6627 140-2568 (Lend       0         00/12/7,204060       31.0,6.7.78       10.0,0.73       TOP       64 41 - 57/14       (CO) Seq22/4 40-6627 140-2568 (Lend       0         00/12/7,204060       31.0,6.7.78       10.0,0.73       TOP       64 91 - 57/14       (CO) Seq24/2 40-1627 140-256 (Lend       0         00/12/7,204060       31.0,6.7.78       10.0,0.73       TOP       64 57/17 - 401 (DN) Seq0 Hand-MSO Hand       0	8674 177,409157	3.13.67.78	18.8.8.252	TL5v1.2	85 Encrypted Alert
65/6 177,144847       10.4.6.732       3.1.3.67.78       10°       54 5717 + 443 (AC) Specify A Acodd27 Minch048 (mm8         66/7 177,144847       10.4.6.732       3.1.3.67.78       10.8.6.752       3.1.3.67.78       10.8.6.752         66/7 177,544840       3.3.3.67.78       10.8.6.752       11.76       64 43 + 5372 (AC) Specify A Acodd27 Minch048 (mm8         66/7 177,544840       3.3.3.67.78       10.8.6.752       11.76       64 43 + 5372 (AC) Specify Acod355 Minch049 (mm8         66/7 177,14277       10.8.6.722       3.1.3.67.78       10.8.6.8521       10°       64 43 + 5372 (AC) Specify Acod355 Minch049 (mm8       Specify Acod355 Minch049 (mm8         66/7 177,14277       10.8.6.722       3.1.3.67.78       10.8.6.8521       10°       64 43 + 10°       10°       Specify Acod355 Minch049 (mm8       Specify Acod355 Minch049 (Specify Acod355 Minch049 (Specify Acod355 Minch	67:0       120.4.0.122       3.1.3.0.7.78       TOP       54 5774 + 440 (EXC) Seq.354 Act.6427 MicroSoPe (Encl         67:0       77.27.40807       108.4.0.223       3.1.3.0.7.78       TOP       64 5774 + 440 (EXC) Seq.354 Act.6427 MicroSoPe (Encl         67:0       77.27.40807       3.8.3.0.7.78       108.4.0.232       TOP       64 41 + 57174 (EXC) Seq.354 Act.6427 MicroSoPe (Encl       56.0.000         67:0       77.74       108.4.0.232       1.3.1.0.7.78       TOP       64 41 + 57174 (EXC) Seq.354 Act.6427 MicroSoPe (Exc) TSMC1         68:0       77.71 (EXC)       108.4.0.232       1.3.1.0.7.78       TOP       64 41 + 57176 (EXC) MicroSoPe (Exc) TSMC1       56.0.000         68:0       77.71 (EXC)       108.4.0.232       100.4.010       64 41 + 57178 (EXC) MicroSoPe (Exc) TSMC1       64.000         69:0       77.00706       108.4.0.122       1.3.1.0.7.78       TOP       64 42 + 57178 (EXC) MicroSoPe (Exc) TSMC2       56.0.000         64:0       77.00706       108.4.0.122       1.3.1.0.7.78       TOP       64 42 + 57178 (EXC) MicroSoPe (Exc) TSMC2       56.0.000         64:0       77.00706       108.4.0.122       10.0.0.000       56.0.000       56.0.000       56.0.000         64:0       7.00707       108.4.0.122       10.0.0.0.000       50.0000       50.0.000 <td< td=""><td>8675 177,410417</td><td>3,13,67,78</td><td>10,0,0,252</td><td>TCP</td><td></td></td<>	8675 177,410417	3,13,67,78	10,0,0,252	TCP	
678 177,54440       3.13, 67.78       10.8, 0.523       170       64 43 + 57374 (ACI Sequel Acta 2555 March 2554 (ACI 2555 March 2564 March 256 March	8/72 172,04440       3.13, 67.72       10.0.0.22       170       64 41 + 5724 [ACG] Septid 27 Acc232 Acc2323 Acc232 Acc232 Acc232 Acc232 Acc232 Acc232 Acc232 A	8676 177.410467	10.0.0.252	3.13.67.78	TCP	
8479 2773-84413       19.8.6.232       1.33.67.78       109       64 9724 - 443 [SM] Seque Min-6426 (Lens)	B473 172.084313       10.8 8.220       1.3.3 0.7.78       TOP       64 372.16 4.41       Status 4420 (some MSS1464 Usc): SSC/1984.1         B481 177.085744       3.3.6 7.78       B8.8.8.252       TOP       64 43 + 5737. [SN, KK] Seek Acks (soci48 Usc): SSC/1984.1         B481 177.085744       3.3.6 7.78       B8.8.8.252       TOP       64 43 + 5737. [SN, KK] Seek Acks (soci48 Usc): SSC/1984.1         B481 177.085745       10.8.6.252       3.1.3.6 7.78       B8.8.8.252       TOP       64 44 + 5737. [SN, KK] Seek Acks (soci48 Usc): SSC/1984.1         B481 177.085765       10.8.6.252       3.1.3.6 7.78       B8.8.8.252       TOP       64 44 + 5737. [SN, KK] Seek Acks (soci48 Usc): SSC/1984.1       SSC/1984.1         B481 177.085765       10.8.6.252       3.1.3.6 7.78       B8.8.8.252       TOP       64 41 + 73818 [SN, KK] Seek Acks (soci48 Usc): SSC/1984.1       SSC/1984.1         B481 177.08576       10.8.6.322       3.1.3.6 7.78       TOP       55 3778 + 461 [KC] Seek Acks (soci48 Usc): SSC/1984.1       SSC/1984.1         B481 177.08583       10.8.6.322       3.1.3.6 7.78       TOP       55 3778 + 461 [KC] Seek Acks (soci48 Usc): SSC/1984.1       <	8677 177.410559	10.0.0.252	3.13.67.78	TCP.	54 57174 + 443 [FIN, ACK] Sequ2354 Acke6627 Wine63094 Lenv0
Bits         Display         Display <thdisplay< th=""> <thdisplay< th=""> <thdisp< th=""><td>0000         177,18227         10.8.0.22         11.8.0.7.78         TCP         06.8.7127.44217         10.8.0.22         11.8.0.7.78         TCP         06.8.7127.81278         10.8.0.22         10.9.0.728</td><td>8678 177.504460</td><td>3.13.67.78</td><td>10.0.0.252</td><td>TCP</td><td>60 443 + 57174 [ACK] Seq+6627 Ack+2355 Win+33580 Len+0</td></thdisp<></thdisplay<></thdisplay<>	0000         177,18227         10.8.0.22         11.8.0.7.78         TCP         06.8.7127.44217         10.8.0.22         11.8.0.7.78         TCP         06.8.7127.81278         10.8.0.22         10.9.0.728	8678 177.504460	3.13.67.78	10.0.0.252	TCP	60 443 + 57174 [ACK] Seq+6627 Ack+2355 Win+33580 Len+0
Inst. 177,095544         3.1.3.6.7.78         18.8.0.22         107         64.8.4         - 977% [SW, AC] Seek Act, 18.0.4288         Inst. RS5-6460           8481 177,095545         10.8.0.52         3.1.3.6.7.78         175         54.577.5         75.457.5         54.577.5         75.457.5         75.457.5         75.457.7         75.457.5         75.257.5         75.457.5         75.457.5         75.257.5         75.457.5         75.257.5         75.257.5         75.257.5         75.257.5         75.257.5         75.257.5         75.257.5         75.257.5         75.257.5         75.277.5         75.277.5         75.277.5         75.277.5         75.277.5	Image: Internet	8679 177.803413	10.0.0.252	3.13.67.78	TOP	66 57176 + 641 (51%) Sequel kins64200 Lense PSSs1468 kSs256 SACK PER61
0482         77.93756         10.8.0.732         3.1.0.67.78         107         54.9726 + 440 [ACK] Sequid Abck Minick/Del Lene#           0481         17.7.99058         3.0.8.0.73         10.8.0.732         10.8.0.73         10.9.0.73           0481         17.7.99058         3.0.8.0.77         10.8.0.732         10.7         64.40         +57718 [SFN, ACQ] Sequid Abck Minick/Del Lene#           0484         17.7.90558         3.0.8.0.77         10.8.0.77         107         64.40         +57718 [SFN, ACQ] Sequid Abck Minick/Del Lene#           0456         377.7.90529         10.8.0.6.12         23.1.0.0.7.78         105.9         49 Add (Abck Minick/Del Lene#           0456         377.7.90529         10.8.0.6.12         23.1.0.0.7.78         105.9.1         571 Classical Minick/Del Lene#           0458         377.7.90528         10.8.0.6.12         105.9.7         105.9.1         571 Classical Minick/Del Lene#         30.01 group 224.0.8.231 for any sources / Join group	Hest 277,49726       10.8.0.252       1.1.3.0.7.76       TCP       55 577,69724       Likel Strinds200 Lend         Hest 377,69925       10.8.0.252       1.1.3.0.7.76       TCP       55 577,6974       Likel Strinds200 Lend         Hest 377,69925       10.8.0.252       1.1.3.0.7.76       TCP       66 43 - 5738       Constant         Hest 377,69925       10.8.0.252       1.1.3.0.7.76       TCP       66 43 - 5738       Constant       Constant         Hest 577,69927       10.8.0.252       1.1.3.0.7.76       TCP       66 443 - 5738       Constant       Constant       Constant         Hest 577,69927       10.8.0.252       1.1.3.0.7.76       TCP       66 443 - 5738       Constant       Con	8688 177.816278	10.0.0.252	3.13.67.78	TCP	66 57178 - 443 (5YN) Sequé kimié4240 Lenvé PSS-1466 MS-256 SACK_PERM-1
100.177.09005         10.4.0.252         3.3.0.7.7         TUSH.2         257 Cleart Melle           400.177.09005         10.4.0.252         3.3.0.7.7         10.6.0.252         3.7.0           400.177.09005         10.4.0.252         3.3.0.7.7         10.6.0.252         3.7.0           400.177.09005         10.4.0.252         3.3.0.7.7         10.6.0.252         3.7.0           400.177.09005         10.4.0.252         3.3.0.7.7         10.6.0.252         3.3.0.7.7           400.177.09005         10.4.0.4.252         3.3.0.7.7         10.6.0.22         10.00           400.177.09005         10.4.0.4.252         3.3.0.7.7         10.0.0.22         10.00           400.177.09105         10.4.0.4.22         10.00         70.00000         10.00.0.25         10.01           400.177.09105         10.4.0.4.22         10.00         70.00000         10.00.0000 <td>1001 377.00005         10.0.0.22         1.3.0.7.78         10.0.0.228         10.0.0.228         10.0.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.0.228         10.0.0.0.228         <t< td=""><td>8681 177.893644</td><td>3.13.67.78</td><td>18.8.8.252</td><td>TCP</td><td>68 443 - 57176 [SYN, ACK] Seque Ackal Win-2688) Lenve PSS-1460</td></t<></td>	1001 377.00005         10.0.0.22         1.3.0.7.78         10.0.0.228         10.0.0.228         10.0.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.228         10.0.0.0.228         10.0.0.0.228 <t< td=""><td>8681 177.893644</td><td>3.13.67.78</td><td>18.8.8.252</td><td>TCP</td><td>68 443 - 57176 [SYN, ACK] Seque Ackal Win-2688) Lenve PSS-1460</td></t<>	8681 177.893644	3.13.67.78	18.8.8.252	TCP	68 443 - 57176 [SYN, ACK] Seque Ackal Win-2688) Lenve PSS-1460
Image: Description         Display State         Display State <thdisplay state<="" th="">         Display State         Disp</thdisplay>	EMA:         177.98033         3.33.97.78         10.8.0.792         10.9.0.77         10.9.0.792         10.9.0.78         10.9.0.77         10.9.0.78	8682 177.893786	10.0.0.252	3.13.67.78	TCP	54 57176 + 443 [ACK] Seq=1 Ack+1 Win+64240 Len+0
665 177,09570         310.40,778         100.40,522         3.13.40,778         107         54 5727.443 [AK] Seq.1 AGAL Min-ASIGS Lenvel           666 177,09570         310.40,738         128.40,4.1         1079 (JA)         64 Percensing Digwrs, general           666 177,09570         310.40,778         128.40,4.1         1079 (JA)         64 Percensing Baptor J         501 prop. 224.40,232 for any sources J         301 no ray sources J         301 no ra	665 177,09570         10.8.0.22         1.3.0.7.78         TCP         55 3727 - 440 [dx(3] Sep1 4ck-1 Str-22056 tened           665 177,09570         10.8.0.22         1.3.0.7.78         120.6.0.1         100% 60 Potements Query, general           665 177,09570         10.8.0.22         1.3.0.7.78         115.0.2.2         100 Potements Report / Join group 224.0.0.251 for any sources		10.0.0.252		TL5v1.2	571 Client Hello
6661 (77,0592)         10.8.0.1         224.0.6.1         109*0         40 Restorating Query, general           6661 (77,05920)         10.8.0.52         3.1.0.6.7         155.0.2         571 Clement Line           6661 (77,05920)         10.8.0.52         234.0.6.2         109*0         70 Restorating Response Display Clement Line           6661 (77,05920)         10.8.0.52         100*0         70 Restorating Response Display Clement Line         100*0           6660 177,059108         3.1.3.6.7.7         10.8.0.22         170         64 41 - 5716 (A25 5ec) Line         100*0           6660 177,059108         10.8.0.52         1.1.3.6.7.7         10.8.0.22         170         64 41 - 5716 (A25 5ec) Line         100*0           6660 177,059108         10.8.0.52         3.1.3.6.7.7         15.0.1.2         195 Server Hello, Change Cleber Sec, Incryotel Madhake Ressage           6901 177,059108         10.8.0.52         3.1.3.6.7.7         10.8.0.52         105         441 - 5712 (A25 5ec) Line         10.8.0.12         10.9 Restorating Clement Line         10.8.0.12           6901 177,059108         10.8.0.52         10.9.0.72         10.9 A01-6712 (A25 5ec) Line         10.9.0.12         10.9 Restorating Clement Line         10.9.0.12           6901 177,059108         10.8.0.52         10.9.0.72         10.9.0.12         10.9	666: 172-05920         10.8.0.1         202.0.0.1         100%->>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>					
6467 27.510243         10.4.0.252         1.13.47.78         11.54.2         10.74           6481 277.510259         10.4.0.252         224.4.0.252         10.94         75.12           6481 277.510259         10.4.0.252         224.4.0.252         10.94         75.12           6481 277.510259         10.4.0.252         10.4.0.252         10.4.0.252         10.4.0.252           6481 277.510259         10.4.0.252         11.54.2         15.94.2         19.94         Ack:518 intro-0.0516 intro           6481 277.510251         3.11.0.77         11.0.0.7.25         11.54.2         19.95 Server, 10.00756 (Entro) 50.6, Entropted Hundhale Message           6491 177.598151         3.0.4.77         11.54.2         11.54.2         11.55.2         11.54.2           6491 177.598151         3.0.4.77         11.54.2         11.55.2         11.54.7         11.54.2           6491 177.598151         3.0.4.77         11.54.2         12.55 Charge Claber Spee, Increpted Hundhale Message         10.97 Charge Claber Spee, Increpted Hundhale Message           6491 177.598151         3.0.4.77         11.54.2         3.01 Charge Claber Spee, Increpted Hundhale Message         10.97 Charge Claber Spee, Increpted Hundhale Message           6491 175.20150         3.1.6.7.78         11.54.2.852         10.31.67.78         10.54.2.852	686 277.93024         1.0.8.0.22         1.3.0.7.78         11.5.0.2         15.0.1.2         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.1         15.0.2.2         15.0.2.1         15.0.2.2         15.0.2.1         15.0.2.2         15.0.2.1         15.0.2.2         15.0.2.1         15.0.2.2         15.0.2.1         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2         15.0.2.2 <td></td> <td></td> <td></td> <td></td> <td></td>					
6468         177, 93090         16.0.6,252         224.0.0.2         100% //         100 Percent hole speet / 30in group 224.0.0.251 for any sources / 30in group 224.0.0.251 for any sources / 30in group 224.0.0.251           1668         177, 93071         3.13.0.7, 78         18.0.0.252         100 //         64 43 - 37376 (AXI) Seq.0.4.252         AxXii Si In-20051 (en-0)           1669         177, 930818         3.13.0.7, 78         18.0.0.252         T.5716 (AXI) Seq.0.4.AxXii Si In-20051 (en-0)           1669         177, 930818         3.0.0.7, 78         15.0.2.2         195 Server Hello, Charge Claber Seq. (Incrysted Hundhake Besage           1691         177, 950815         10.0.0.252         1.05 Ar.2         195 Charge Claber Seq. (Incrysted Hundhake Besage           1691         177, 950815         10.0.0.252         1.05 Ar.2         195 Charge Claber Seq. (Incrysted Hundhake Besage           1691         177, 950815         10.0.0.252         105 Grad         105 Ar.2         105 Charge Claber Seq. (Incrysted Hundhake Besage           1691         177, 950815         10.0.0.252         105 Grad         105 Ar.2         105 Grad           1691         17.0.0.0.251         10.0.0.77         10.0.0.252         105 Grad         105 Grad           170         10.0.0.278         0.0.0.0.252         10.0.0.0.252         105 Grad <td< th=""><td>4688 177,930900             10.6.0.252             226.0.0.2             226.0.0.2             1009-&gt;</td><td></td><td></td><td></td><td></td><td></td></td<>	4688 177,930900             10.6.0.252             226.0.0.2             226.0.0.2             1009->>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>					
6609 177,987571       3.13.67.78       10.8.4.523       TCP       64 43 - 57276 [KX] Sepi-1 Ack/538 (MAD10456 (em-0)         6609 177,987678       3.13.67.78       10.8.4.523       TL57-1.2       195 Server Hello, Change (Cherber Spec, Incrysteel Minodahak Reisage         6601 177,987688       10.8.4.252       3.13.67.78       TL57-1.2       195 Server Hello, Change (Cherber Spec, Incrysteel Minodahak Reisage         6601 177,987688       10.8.4.252       3.13.67.78       TL57-1.2       195 Server Hello, Change (Cherber Spec, Incrysteel Minodahak Reisage         6601 177,987681       10.8.4.252       3.13.67.78       TL57-1.2       195 Server Hello, Change (Cherber Spec, Incrysteel Minodahak Reisage         6601 177,987681       10.8.4.252       3.13.67.78       TL57-1.2       105 Server Hello, Change (Cherber Spec, Incrysteel Minodahak Reisage         6601 177,987681       10.8.4.252       TL57       60 443 - 57178 [ACK] Sepi-1 Ack/518 Min-28160 Lem-0         Frame S7081 (Stytes on wire (648 Dits), 85 Systes captured (648 Dits) on Interface Uperice(HW	669 177, 98771         3.13.6.7.78         19.4.0.523         17.9         19.4.3.23         17.9         19.4.0.232         15.1.2.77         19.4.0.232         15.1.2         19.5.2         15.2					
6601 277,09818         3.3.3.67.78         18.6.6.22         15.94.2         19 Server Hello, Change Clober See, Encryptel HandShaler Ressage           6601 277,09818         3.3.3.67.78         15.94.2         195 Server Hello, Change Clober See, Encryptel HandShaler Ressage           6601 277,09818         10.8.6.32         3.3.3.67.78         15.94.2         195 Omega           6601 277,09818         10.8.6.32         3.3.3.67.78         15.94.2         195 Omega           6901 277,09818         10.8.6.32         3.3.3.67.78         15.94.2         195 Omega           6901 277,09818         10.8.6.32         10.8.6.32         10         64 40.5         17.92164 (amed           7ease SPAD: B5 bytes on using (SBB 515), B5 bytes captured (SBB	1609 177, 901818       3.13, 67, 78       10.0, 0, 32       11.5, 42, 20       11.5, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42					
6001         177.095048         10.6.8.252         3.1.3.67.78         TLSvJ.2.         105 Onage Clove Specific Mondbalk Message           6001         77.095048         10.8.6.252         3.1.3.67.78         TLSvJ.2.         105 Onage Clove Specific Mondbalk Message           6001         77.095048         10.8.6.252         3.1.3.67.78         TLSvJ.2.         2017 Onage Clove Specific Mondbalk Message           6001         77.095048         10.8.6.252         3.1.3.67.78         TLSvJ.2.2017 Optic Mondbalk Message           6001         77.095048         10.8.6.252         TD         60.43.5         57.218 (AK) Seg.1.6.0.04518 Min-2010 Mess           6001         77.095048         Mondbalk Message         10.8.6.252         TD         60.43.5         57.218 (AK) Seg.1.6.0.0518 Min-2010 Message           6.0001         Mondbalk Message         Mondbalk Message         Mondbalk Message         57.218 (AK) Seg.1.18.0000-541758052063), 10.8         0           Transistion Control Protocol, Seg. Nevel: 44.8, 0.0.8.252         Transistion Control Protocol, Seg. Nevel: 44.8, 0.0.8.252         Heriti Strategies Message         10.8.6.252           Transistion Control Protocol, Seg. Nevel: 44.9, 0.01 Rett: 57276, 584: 454. AKK: 2852, Len: 31         Transistion Control Protocol, Seg. Nevel: 44.9, 0.01 Rett: 57276, 584: 454. AKK: 2852, Len: 31         Transistion Control Protocol, Seg.10.00 Min Wersion TLS J.2. (04.000)	6001 177:990404       10.0.0.252       3.13.07.78       TUSV1.2       100 Charge Cipers Sect, Encrypted Machina Wessage         6001 177:990404       10.0.0.252       3.13.07.78       TUSV1.2       207 Application Data         6001 177:990404       10.0.0.252       1.13.07.78       TUSV1.2       207 Application Data         6001 177:990404       10.0.0.0.522       TCP       0.0.0.43.5       207 Application Data         6001 177:990404       10.0.0.0.77       TUSV1.2       207 Application Data       Application Data         6001 177:00104       10.0.0.0.77       TUSV1.2       207 Application Data       Application Data         7000 February       TUSV1.2       207 Application Data       Application Data       Application Data         7000 February       TUSV1.2       200 February       Data       Application Data       Application Data         7000 February       TUSV1.2       200 February       Data       Application Data       Application Data         7000 February       TUSV1.2       200 February       Data       Application Data       Data         7000 February       TUSV1.2       200 February       Data       Data       Data       Data         7000 February       Application Data       200 February       Data       Data					
0002 177,999815         10:0.6.252         3.3.0.67.78         T15/3.2         237 Poplication Data, Application Da	0002 177.998315         10.8.0.252         1.3.0.778         TLSv1.2. 237 AppLication Data           0003 1778.00153         3.13.0.778         10.0.0.728         TDF         64.43.57178         LOCALSCH		3.13.67.78			
1093 1278.0012/0       3.13.47.78       10.0.0.22       10       60.43.5 ST278 [ACK] Sep-1 Activ518 kin=28160 [an=0         Frame STNE IS bytes on vire (680 bits), 85 bytes captured (680 bits) on interface (bruice(640 [cl300])-1640-427C-900-561558522063), 10.0         Werner II, Size, Technica, 71:0:0:0 [cl00212:71:70:00, 05:11 0.0.0.252         Internet Protocol Version 4, Sec: 3.13.67.78, ost: 10.0.0.252         Transitision Control Protocol, 55:0 Potert 443, 014 Pett: 53256, Seg: 454, A4K: 2852, Len: 31         Transitision Control Protocol, 55:0 Potert 443, 014 Pett: 53256, Seg: 454, A4K: 2852, Len: 31         Transitision Control Protocol, 55:0 Potert 443, 014 Pett: 53256, Seg: 454, A4K: 2852, Len: 31         Visivi2, Record Users: In orypote Alert         Content Type: Alert (12)         Wersion Ti, 5.2, (04000)	1093 128.00120       3.13.67.78       10.0.0.202       10*       60.43.9.5712% [ACX] Sepi-1 Adx-518 Min-28160 Lenned         Frame SNNE ISS bytes on wine (688 bits), as bytes captured (688 bits) on interface (benice(WF) (2130099-348-417C-9906-54715852263), 10 0         Externet T1, Scr. Technico, 71:xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
Frame B708: 85 Systes on wire (680 bits), 85 Systes captured (660 bits) on interface \Device\WF[21304099-3F48-417C-9906-56F138522063), 14 0 Ethernet TL, Src: Technic, g7:Det-40 (Bc:00127:07:Det-40), Dot: Microsof_ed:19:75 (64:Det-19:76) Transport Layer Security FrameWire AD, Det Provide, Security FrameWire AD, Det	Frame B708: 85 bytes on wire (688 bits), 85 bytes captured (688 bits) on interface lbevice/U#/_(2130409-3/48-4/37C-9006-56/158522063), 16 0         Ethernet TV. Soc: Technic, 07:0x-40 (0c:0212:07:0x-40), 0c:01 x1(crosof_d4:02:0)         Internet TV. Soc: Technic, 07:0x-40 (0c:0212:07:0x-40), 0c: X1(crosof_d4:02:0)         Internet TV. Soc: Technic, 07:0x-40 (0c:0212:07:0x-40), 0c: X1(crosof_d4:02:0x-00:0x-40)         Internet TV. Soc: Technic, 07:0x-40 (0c:0212:07:0x-40, 0c: X1(crosof_d4:02:0x-00:0x-40)         Internet TV. Soc: Technic, 04, 0c: X1(crosof_d4:02:0x-00:0x-40)         Internet TV. Soc: Technic, 04, 0c: X1(crosof_d4:04:0x-00:0x-40)         Internet TV. Soc: Technic, 04, 0c: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: Assoc: Technic, 04, 0c: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: Assoc: Technic, 04, 0c: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: Technic, 04, 0c: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: Assoc: Technic, 04, 0c: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: Assoc: Technic, 04, 0c: X1(crosof_d4:0x-40)         Internet TV. Soc: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: X1(crosof_d4:0x-00:0x-40)         Internet TV. Soc: X1(crosof_d4:0x-40:0x-40)         Internet TV. Soc: X1(crosof_d4:0x-40:0x-40)	8691 177.995048				
<pre>Uberner 11, Sec: Technico, 27:Dec40 (0c:02:27:750=00), Dirt Microsof_e8:19:70 (c4:5dired:e8:19:70) Intermer Protocol Version 4, Sec: 31.3.3.70, Dirt Dir No.0.352 Transdission Control Protocol, Sec Part: 443, Dirt Port: 53276, Seq: 454, Ack: 2852, Len: 31 Transport Layer Security v Liviz: 2 Record Layer: Loreysted Alert Content Type: Alert (21) Version: Tis 2.2 (0e00)</pre>	Etherner 11, Srci Technico (7:be:40) (bit: 22: 7:2:be:40), Dit: Hicrosof_e8: 92:70 (d:dd:edi:e8: 92:70)           Interner Protocol Wriston 4, Src 13.13.6.72, Dit: 10.8.0.22           Translation Control Protocol, Src Pert: 43, Dit Port: 57176, Seq: 454, Ack: 2852, Lee: 31           Translow Lippe: Exerciting           * LSV:12. Record Lippe: Intropiced Alert           Genter Protocol, Ser Exerciting           * LSV:12. Record Lippe: Intropiced Alert           Genter Protocol           Series (0:0000)           Intropice Intropiced Alert           Series (0:00000)           Series (0:00000)           Series (0:00000)           Series (0:000000)           Series (0:00000000000000000000000000000000000	8691 177.995848 8692 177.995815	10.0.252	3.13.67.78	TLSv1.2	2337 Application Data, Application Data
	010 00 47 dt 15 40 00 46 66 72 44 dt 30 43 44 64 80 0 dt - μ <sup>2</sup> - τ <sup>2</sup> 0 - (N+- 000 00 47 dt 15 40 75 46 15 28 to 76 46 35 45 21 tt	8091 177.995048 8092 177.995815 8093 178.001769 Frame 8708: 85 byt Ethernet II, Src:	10.0.0.252 3.13.67.78 es on wire (680 bits), Technico_f7:be:49 (0c:	3.13.67.78 10.0.0.252 85 bytes captured (6 02:27:f7:be:49), Dst:	TLSv1.2 TCP 80 bits) Microsof	2377 deplication Data, Aeplication Data (e) 449 - 57278 [ACC] Sept. Ackolsk kin-Blakk (m-Bl on interface: Mexice/MP(21300099-3/48-417C-900-56/158522063), 1d 0

Figure 7: Viewing colorized packets in Wireshark

However, you're not limited to just interpreting by color. It's possible to view the input/output (I/O) statistics of an entire packet capture. In Wireshark, just go to Statistics >> I/O Graph, and you'll see a graph similar to the one shown in Figure 8.

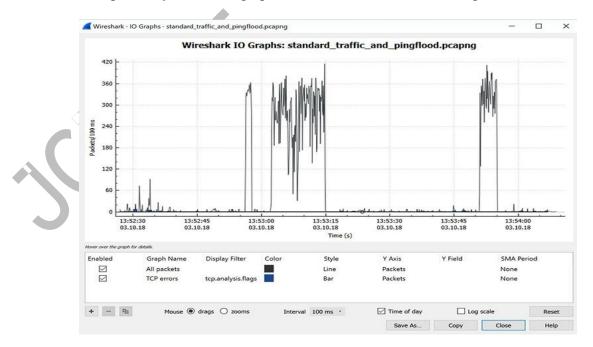


Figure 8: Viewing the input/output traffic graph in Wireshark

This particular graph is showing typical traffic generated by a home office. The spikes in the graph are bursts of traffic that were caused by generating a <u>Distributed Denial of</u> <u>Service (DDoS) attack</u> using a few Linux systems.

In this case, three major traffic bursts were generated. Many times, cybersecurity pros use Wireshark as a quick and dirty way to identify traffic bursts during attacks.

It's also possible to capture the amount of traffic generated between one system and another. If you go to Statistics and then select Conversations, you will see a summary of conversations between end points, as shown below in Figure 9.

Tx Packets         Tx Bytes           33         15           9         272           11         116           1         7           1         11           1         10           1         10           1         11           1         10           1         11           1         10           1         11           14         184           1         7           23         71	k 44 1 9 6 12 0 0 0 0 0 0 0 2 0 2 0 0 0 0 0 0	Rx Bytes         AS Number           35 k         AS8075 Microsoft Corporation           1611         AS16509 Amazon.com, Inc.           1716         AS16509 Amazon.com, Inc.           0         AS7922 Comcast Cable Communications, LLC           0         AS7922 Comcast Cable Communications, LLC
9 272 11 116 1 7 1 11 1 11 1 11 1 10 1 10 1 11 14 184 1 7	1 9 6 12 0 0 0 0 0 0 2 0 2 0 0 0 0 0	1611 AS16509 Amazon.com, Inc. 1716 AS16509 Amazon.com, Inc. 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
11 116 1 7 1 11 1 11 1 11 1 10 1 10 1 11 14 184 1 7	6 12 0 0 0 0 2 0 2 0 0 0 0 0	1716 AS16509 Amazon.com, Inc. 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
1 7 1 11 1 11 1 10 1 10 1 10 1 11 14 184 1 7	0 0 0 0 2 0 2 0 0 0	0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
1 11 1 11 1 10 1 10 1 11 14 184 1 7	0 0 0 0 2 0 2 0 0 0	0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
1 11 1 10 1 10 1 11 14 184 1 7	0 0 2 0 2 0 0 0	0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
1 10 1 10 1 11 14 184 1 7	2 0 2 0 0 0	0 AS7922 Comcast Cable Communications, LLC 0 AS7922 Comcast Cable Communications, LLC
1 10 1 11 14 184 1 7	2 0 2 0 0 0	0 AS7922 Comcast Cable Communications, LLC
1 11 14 184 1 7	2 0 0 0	
14 184 1 7	0 0	
1 7	6 14	
1 7		1006 AS7922 Comcast Cable Communications, LLC
22 74	0 0	0 AS7922 Comcast Cable Communications, LLC
	k 32	3011 AS13335 Cloudflare Inc
6 55		557 AS15169 Google LLC
22 30		14 k AS8075 Microsoft Corporation
1,873 19 1		177 k AS33011 Box.com
8 418		1635 AS33011 Box.com
207 276		95 k AS33011 Box.com
5 107		1126 AS33011 Box.com
25 270	1 23	5923 AS21581 M5 Computer Security
8 120		824 AS8075 Microsoft Corporation
11 380		2310 AS23467 New Relic
23 221	9 38	4467 AS16509 Amazon.com, Inc.
	0 70	10 k —
1,166 129	k 649	171 k AS393324 CompTIA, Inc.
51 41		20 k AS36007 Kamatera, Inc.
10 395		9642 AS36007 Kamatera, Inc.
1 18		0 AS2828 MCI Communications Services, Inc. d/b/a Ve
		0 AS2828 MCI Communications Services, Inc. d/b/a Ve
		0 AS2828 MCI Communications Services, Inc. d/b/a Ve
		0 AS2828 MCI Communications Services, Inc. d/b/a Ve
		· · · · · · · · · · · · · · · · · · ·
	1 18 1 18 1 18	1 182 0 1 182 0 1 182 0 1 182 0

Figure 9: Viewing endpoint conversations in Wireshark

In the above case, Wireshark was used to see if an old piece of equipment from MCI communications that was running on a client's network could be traced.

It turned out that the client didn't know this device was even on the network. Thus, it was removed, helping to <u>make the network a bit more secure</u>. Notice, also, that this network connection is experiencing a lot of traffic to Amazon (administering a server in AWS at the time) and Box.com (using Box for system backup at the time).

In some cases, it is even possible to use Wireshark to identify the geographic location of source and destination traffic. If you click on the Map button at the bottom of the screen (shown in Figure 9 above), Wireshark will show you a map (Figure 10), providing its best guess of the location of the IP addresses you've identified.

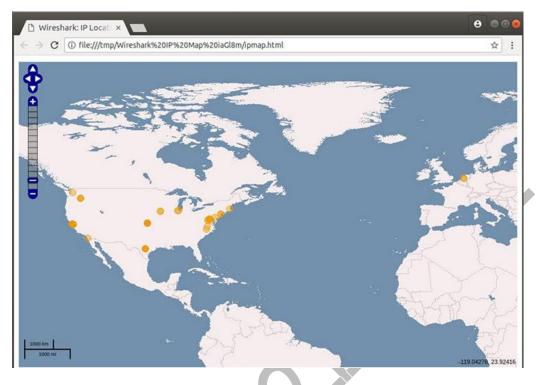


Figure 10: Viewing geographic estimations in Wireshark

Because IPv4 addresses can be easily spoofed, you can't rely completely on this geographical information. But it can be fairly accurate.

How to Filter and Inspect Packets in Wireshark

You can apply Wireshark filters in two ways:

In the Display Filter window, at the top of the screen

By highlighting a packet (or a portion of a packet) and right-clicking on the packet

Wireshark filters use key phrases, such as the following:

ip.addr Specifies an IPv4 address

ipv6.addr Specifies an IPv6 address

src Source - where the packet came from

dst Destination - where the packet is going

You can also use the following values:

- && Means "and," as in, "Choose the IP address of 192.168.2.1 and 192.168.2.2"
- == Means "equals," as in "Choose only IP address 192.168.2.1"
- ! Means "not," as in, do not show a particular IP address or source port

Valid filter rules are always colored green. If you make a mistake on a filter rule, the box will turn a vivid pink.

Let's start with a couple of basic rules. For example, let's say you want to see packets that have only the IP address of 18.224.161.65 somewhere inside. You would create the following command line, and put it into the Filter window:

Figure 11 shows the results of adding that filter:

R io.add/ == 18.224.161.63			¢Π		80
No. True	Source	Destruction	The second	length linfo	w
1776 2.207232	18,224,161,65	10.0.0.252	TCP	60 443 + 57147 [ACK] Sequ146 Ack+1685 kin+32364 Len+0	
1778 2.377633	18.224.161.65	10.0.0.252		50 443 4 5/14/ [ACA] SEQUING ACKIDES HIN-JING LENKE	
1779 2,418909	10.0.0.252	18,224,161.65	TCP	54 57147 - 443 [ACK] Seq=1685 Ack=603 Win=63638 Len=0	
1780 3.375947	18.224.161.65	10.0.0.252		85 Encrypted Alert	
1781 3.377601	18,224,161,65	10.0.0.252	TCP	68 443 + 57147 [FIN, ACK] Sequ634 Ack+1685 kin+32364 Lenv8	
1782 3.377830	10.0.0.252	18.224.161.65	TCP	54 57147 + 443 [ACK] Seg+1685 Ack+635 K1n+63607 Len+0	
1783 3, 377592	10,0;0,252	18,224,161.65	TCP	54 57147 - 443 [FIN, ACK] Seg-1685 Ack+635 Win+63607 Len+8	
1784 3.474021	18,224,161,65	10.0.0.252	TCP	60 443 = 57147 [ACK] Seg+635 Ack+1686 Win+32364 Len+0	
2272 14,796445	10.0.0.252	18,224,161,65	TCP	66 57151 + 443 [5YN] Seque Win+64248 Len+8 MSS+1468 WS+256 SACK_PER%1	
2273 14,890707	18.224.161.65	10.0.0.252	TCP	60 443 - 57151 [SYN, ACK] Seg-0 Ack+1 Min+26883 Lenvel MS5+1460	
2274 14.890811	10.0.0.252	18.224.161.65	TCP	54 57151 + 443 [ACK] Seg-1 Ack+1 Min+64240 Len+0	
2275 14.893873	10.0.0.252	18.224.161.65	TLSv1.2	571 Client Hello	
2278 14.988869	18,224,161.65	10.0.0.252	TCP	60 443 - 57151 [ACK] Seg-1 Ack-518 Win-30016 Len-0	
2279 14.989828	18.224.161.65	10.0.0.252	TLSv1.2	199 Server Hello, Change Cipher Spec, Encrypted Handshake Message	
2280 14.992127	10.0.0.252	18.224.161.65	TL5v1.2	105 Change Cipher Spec, Encrypted Handshake Message	
2286 15.122402	18.224.161.65	10.0.0.252	TCP	60 443 + 57151 [ACX] Seq=146 Ack+569 kin=30016 Len=0	
2287 15.122443	10.0.0.252	18.224.161.65	TLSv1.2	1203 Application Data	
2294 15.207489	18.224.161.65	10.0.0.252	TCP	60 443 - 57151 [ACK] Seq=146 Ack=1718 Win=32172 Len=0	
2295 15.208166	18.224.161.65	10.0.0.252		362 Application Data	
2297 15.249339	10.0.0.252	18.224.161.65	TCP	54 57151 = 443 [ACK] Seq=1718 Ack=454 Win=63787 Len=0	
2315 16.205590 > Frame 2315: 85 by > (thernet II, Src: > Internet Protocol > Transmission Conto	18.224,161.65 tes on wire (680 bits Technico_f7:be:49 (6 Version 4, Src: 18.1 rol Protocol, Src Par	10.0.0.252 ), 85 bytes captured hc:02:27:67:be:49), 0 h24.161.65, 0st: 10.0	TLSv1.2 (680 bits) o st: Microsof, 0.252	54 5755 - 445 (Adr.) 5449718 Accesson Alime5787 (ene 55 Tocopied Adret 16 Tocopied Adret 16 Folzo (cc:stociete(197)) 44 - 417C-9906 -5(F158522652), 18 0 46 Folzo (cc:stociete(197)) 48 - 417C-9906 -5(F158522652), 18 0 48 - 417C-9906 - 417C-	
2315 16.205500 > Frame 2315: 85 by S (Bhernet II, Sec: > Internet Protocol > Transistic Cat > Transistic Cat > Transport Layer S 0000 c4 65 ed e8 f6 0010 e0 47 b8 ed 40 0000 e0 46 b6 ed b6 d7	18.224,161.65 tes on wire (680 bits Technico_f7:be:49 (6 Version 4, Src: 18.1 rol Protocol, Src Par	10.0.0.252 ), 85 bytes (aptured (002)27(7)0+64), 0 224.161.65, 0+15 10.0 10 08 00 45 00 0 41 41 06 00 - 6 0 4 60 50 0 0 4 60 50 0 0 4 60 50 0 10 08 00 45 00	TLSv1.2 (680 bits) o st: Microsof, 0.252	85 Encrypted Alert n Interface \Device\UBF_(2104099-3F48-4EXC-9906-5EF158522C62), 18 0 (81:9270 (64:96:ed:e81:99:70)	

Figure 11: Applying a filter to a capture in Wireshark

Alternatively, you can highlight the IP address of a packet and then create a filter for it. Once you select the IP address, right-click, and then select the Apply As Filter option.

You'll then see a menu of additional options. One of those is called Selected. If you choose Selected, then Wireshark will create a filter that shows only packets with that IP address in it.

You can also decide to filter out a specific IP address using the following filter, also shown in Figure 12:

a55'++18.224.161.65	1.				80++
Time	Source	Destination	Research	Length Solo	
2291 15.203874	209,205.52.111	10.0.0.252	TCP	1514 993 = 57152 [ACK] Segul461 Acku142 Winu30720 Lenu1460 [TCP segment of a reassembled POU]	
2292 15.203875	209.206.52.111	10.0.0.252		910 Certificate, Server Key Exchange, Certificate Request, Server Hello Done	
293 15.204476	10.0.0.252	209,205,52,111	TCP	54 57152 + 993 [ACK] Segu142 Ack+3777 Min=262656 Lenie@	
296 15.233910	10.0.0.252	209,206,52,111		224 Certificate, Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message	
298 15.293895	289,286,52,111	10.0.0.252		280 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message	
299 15, 335242	10.0.0.252	209.205.52.111	TCP	54 57152 + 993 [ACK] Seq+312 Ack+4003 Win+262400 Len+0	
300 15,393011	209,206.52.111	10.0.0.252	TLSv1.2	188 Application Data	
301 15,433257	10.0.0.252	209,205,52,111	TCP	54 57152 + 993 [ACK] Sequ312 Ack+4057 Win+262400 Leni0	
302 15,436791	10.0.0.252	209.205.52.111	TLSv1.2	177 Application Data	
383 15,499777	209,205.52.111	10.0.0.252	TLSv1.2	131 Application Data	
384 15,536497	10.0.0.252	209,205,52,111	TLSv1.2	NO Application Data	
385 15.639074	209.206.52.111	10.0.0.252	TCP	60 993 - 57152 [ACK] Sequ4134 Ack+641 Nin+32768 Len+0	
306 15.649625	10.0.0.252	13.107.136.9	TLSv1.2	1626 Application Data	
307 15.649768	10.0.0.252	13.107.136.9	TL5v1.2	305 Application Data	
388 15.671785	13.107.136.9	10.0.0.252	TCP	60 443 - 57115 [ACK] Seq=1277 Ack=3264 Win=1026 Len=0	
309 15.676841	13.107.136.9	10.0.0.252	TCP	60 443 - 57115 [ACK] Seq=1277 Ack=3647 Win=1025 Len=0	
310 15.759876	13.107.136.9	10.0.0.252	TLSv1.2	1331 Application Data	
311 15.801159	10.0.0.252	13,107.136.9	TCP	54 57115 = 443 [ACK] Sequ3647 Ackx2554 Winx1024 Lenv0	
312 16.076426	10.0.0.252	54.149.94.178	TLSv1.2	112 Application Data	
313 16.114264	fe80:1902e:6cff:fea.	6602111	1010-6	174 Router Advertisement from 0c:02:27:47:be:49	
name 2314: 60 by thernet II, Sec: sternet Protocol		10.0.0.252 . 60 bytes captured .02:27:f7:be:49), Dr 0.94.178, Dst: 10.0.	TCP (480 bits) ( it: Microsof, 0.252	60 443 + 57119 (ACC) Seq.2007 AcXo115 kine0 (ene0 on interface \Device\VMF_212100009-3F48-427C-990E-5FF138522C63), 10 0 y#1997b (c4:90rd:e8:1997b)	-
thernet II, Src: sternet Protocol	tes on wire (480 bits), Technico_f7:be:49 (0c: Version 4, Src: 54.149	10.0.0.252 . 60 bytes captured .02:27:f7:be:49), Dr 0.94.178, Dst: 10.0.	TCP (480 bits) ( it: Microsof, 0.252	60 443 + 57119 (ACC) Seq.2007 AcXo115 kine0 (ene0 on interface \Device\VMF_212100009-3F48-427C-990E-5FF138522C63), 10 0 y#1997b (c4:90rd:e8:1997b)	
ame 2314: 68 by hernet II, Src: ternet Protocol ansmission Cont 04 9d ed e8 f9 00 28 70 77 60 fc 81 56 df	tes on wire (480 bits), Technico_f7:be:49 (0c: Version 4, Src: 54.149	10.0.0.252 (40.bytes captured 2022)7(72bec40), Ob 104.178, Dat 100.0 (443, Dat Poet: 577 (443, Dat Poet: 577	TCP (480 bits) ( tt: Microsof, 0.252 119, Seq: 10 { I - E. 4 . 9 6 ~	40:40 - 57139 (ACK 5942407 AKKa15 Min-0 Len-0 on interface (Device(NUF_C1106009-3F48-4E7C-900C-5EF35832C63), 1d 0 fff770 (ACHNORE4, F9270) 67, ACk: 315, Len: 0	
ame 2314: 68 by hernet II, Src: ternet Protocol ansmission Cont 04 9d ed e8 f9 00 28 70 77 60 fc 81 56 df	tes on wire (488 bits), Technico, F7:bet69 (80: Version 4, Srec 54.148 nol Protocol, Src Port: Protocol, Src Port: 70 8c 62 27 f7 be 46 e0 26 66 39 16 36 55	10.0.0.252 (40.bytes captured 2022)7(72bec40), Ob 104.178, Dat 100.0 (443, Dat Poet: 577 (443, Dat Poet: 577	TCP (480 bits) t: Hicrosof, 0.252 119, Seq: 10 (	40:40 - 57139 (ACK 5942407 AKKa15 Min-0 Len-0 on interface (Device(NUF_C1106009-3F48-4E7C-900C-5EF35832C63), 1d 0 fff770 (ACHNORE4, F9270) 67, ACk: 315, Len: 0	

!ip.addr==18.224.161.65

Figure 12: Filtering out a specific IP address in Wireshark

You're not limited to just IPv4 addresses. For example, if you want to see if a particular computer is active and using an IPv6 address on your network, you can open up a copy of Wireshark and apply the following rule:

ipv6.dst == 2607:f8b0:400a:15::b. This same rule is shown in Figure 13.

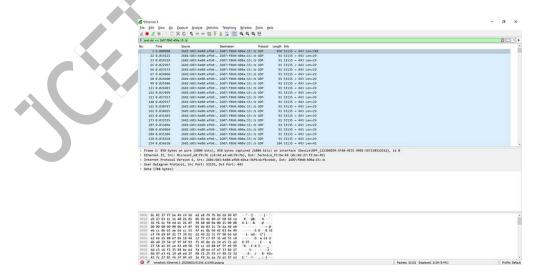


Figure 13: Applying an IPv6 filter in Wireshark

Clearly, this system is alive and well, talking on the network. There are so many possibilities.

Additional filters include:

tcp.port==8080	Filters packets to show a port of your own choosing – in this case, port 8080
!(ip.src ==	Shows all packets except those originating from
162.248.16.53)	162.248.16.53
!(ipv6.dst ==	Shows all packets except those going to the IPv6 address of
2607:f8b0:400a:15::b)	2607:f8b0:400a:15::b
ip.addr == 192.168.4.1	
&&	Shows both 192.168.4.1 and 192.168.4.2
ip.addr == 192.168.4.2	
http.request	Shows only http requests – useful when troubleshooting or visualizing web traffic
	0

# **PROGRAM 11 – NETWORK CONFIGURATION PROCEDURES**

Network software installation takes place along with the installation of the operating system software. At that time, certain IP configuration parameters must be stored in appropriate files so they can be read at boot time.

The procedure is a matter of creating or editing the network configuration files. How configuration information is made available to a machine's kernel depends on whether these files are stored locally (**local files** mode) or acquired from the network configuration server (**network client** mode).

Parameters supplied during network configuration are:

- IP address of each network interface on every machine
- Host names of each machine on the network. You can type the host name in a local file or a name service database.
- NIS, NIS+, or DNS domain name in which the machine resides, if applicable
- Default router addresses. You supply this only if you have a simple network topology with only one router attached to each network, or your routers don't run routing protocols such as the Router Discovery Server Protocol (RDISC) or the Router Information Protocol (RIP). (See <u>"Routing Protocols"</u> for more information about these protocols.)
- Subnet mask (required only for networks with subnets)

This chapter contains information on creating and editing local configuration files. See the <u>Solaris</u> <u>Naming Administration Guide</u> for information on working with name service databases.

Network Configuration Task Map Table 6-3 Network Configuration Task Map

Task	Description
Configure a host for local files mode	Involves editing the nodename, hostname, hosts, defaultdomain, defaultrouter, and netmasks files.
Set up a network configuration server	Involves turning on the in.tftp daemon, and editing the intetd.conf, hosts, ethers, and bootparams files.
Configure a host for network client mode	Involves creating hostname file, editing the hosts file, and deleting the nodename and defaultdomain files, if they exist,

Task	Description
Specify a rou the network cl	Involves editing the defaultrouter and hosts files.

How to Configure a Host for Local Files Mode

Use this procedure for configuring TCP/IP on a machine that runs in local files mode.

- 1. Become superuser and change to the /etc directory.
- 2. Type the host name of the machine in the file /etc/nodename.

For example, if the name of the host is tenere, type tenere in the file.

3. Create a file named /etc/hostname.*interface* for each network interface.

(The Solaris installation program automatically creates this file for the primary network interface.) Refer to <u>"/etc/hostname.*interface* File"</u> for details. If you are using IPv6, see <u>"IPv6 Network Interface Configuration File"</u>.

4. Type either the interface IP address or the interface name in each /etc/hostname.*interface* file.

For example, create a file named hostname.ie1, and type either the IP address of the host's interface or the host's name.

- 5. Edit the /etc/inet/hosts file to add:
  - a. IP addresses that you have assigned to any additional network interfaces in the local machine, along with the corresponding host name for each interface.

The Solaris installation program has already created entries for the primary network interface and loopback address.

b. IP address or addresses of the file server, if the /usr file system is NFS mounted.

## Note -

The Solaris installation program creates the default /etc/inet/hosts for the local machine. If the file does not exist, create it as shown in <u>"hosts Database"</u>. Also, if you are using IPv6, see <u>"/etc/inet/ipnodes File"</u>.

6. Type the host's fully qualified domain name in the /etc/defaultdomain file.

For example, suppose host tenere was part of the domain deserts.worldwide.com. Therefore, you would type: deserts.worldwide.com in /etc/defaultdomain. See "/etc/defaultdomain File" for more information.

7. Type the router's name in /etc/defaultrouter.

See <u>"/etc/defaultrouter File"</u> for information about this file.

8. Type the name of the default router and its IP addresses in /etc/inet/hosts.

Additional routing options are available. Refer to the discussion on routing options in <u>"How</u> to <u>Configure Hosts for Network Client Mode"</u>. You can apply these options to a local files mode configuration.

9. If your network is subnetted, type the network number and the netmask in the file /etc/inet/netmasks.

If you have set up a NIS or NIS+ server, you can type netmask information in the appropriate database on the server as long as server and clients are on the same network.

10. Reboot each machine on the network.

How to Set Up a Network Configuration Server

- 1. Become superuser and change to the root directory of the prospective network configuration server.
- 2. Turn on the in.tftpd daemon by creating the directory /tftpboot:

# mkdir /tftpboot

- 3. This configures the machine as a TFTP, bootparams, and RARP server.
- 4. Create a symbolic link to the directory.

## # ln -s /tftpboot/. /tftpboot/tftpboot

5. Enable the tftp line in intetd.conf.

Check that the /etc/inetd.conf entry reads:

tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd -s /tftpboot

This prevents **inettftpd**() from retrieving any file other than one located in /tftpboot.

- 6. Edit the hosts database, and add the host names and IP addresses for every client on the network.
- 7. Edit the ethers database, and create entries for every host on the network to run in network client mode.
- 8. Edit the bootparams database.

See <u>"bootparams Database"</u>. Use the wildcard entry or create an entry for every host that run in network client mode.

9. Reboot the server.

Information for setting up install servers and boot servers can be found in <u>Solaris Advanced</u> <u>Installation Guide</u>.

Configuring Network Clients

Network clients receive their configuration information from network configuration servers. Therefore, before you configure a host as a network client you must ensure that at least one network configuration server is set up for the network.

How to Configure Hosts for Network Client Mode

Do the following on each host to be configured in network client mode:

- 1. Become superuser.
- 2. Check the directory for the existence of an /etc/nodename file. If one exists, delete it.

Eliminating /etc/nodename causes the system to use the hostconfig program to obtain the host name, domain name, and router addresses from the network configuration server. See "Network Configuration Procedures".

3. Create the file /etc/hostname.*interface*, if it does not exist.

Make sure that the file is empty. An empty /etc/hostname.*interface* file causes the system to acquire the IP address from the network configuration server. If you are using IPv6, see "IPv6 Network Interface Configuration File".

4. Ensure that the /etc/inet/hosts file contains only the host name and IP address of the loopback network interface.

(See <u>"Loopback Address"</u>.) The file should not contain the IP address and host name for the local machine (primary network interface). If you are using IPv6, see <u>"/etc/inet/ipnodes File"</u>.

5. Check for the existence of an /etc/defaultdomain file. If one exists, delete it.

The hostconfig program sets the domain name automatically. If you want to override the domain name set by hostconfig, type the substitute domain name in the file /etc/defaultdomain.

6. Ensure that the search paths in the client's /etc/nsswitch.conf reflects the name service requirements for your network.

How to Specify a Router for the Network Client

- 1. If you have only one router on the network and you want the network configuration server to specify its name automatically, ensure that the network client does not have a /etc/defaultrouter file.
- 2. To override the name of the default router provided by the network configuration server:
  - a. Create /etc/defaultrouter on the network client.
  - b. Type the host name and IP address of the machine you have designated as the default router.
  - c. Add the host name and IP address of the designated default router to the network client's /etc/inet/hosts.
- 3. If you have multiple routers on the network, create /etc/defaultrouter on the network client, but leave it empty.

Creating /etc/defaultrouter and leaving it empty causes one of the two dynamic routing protocols to run: ICMP Router Discovery protocol (RDISC), or Routing Information Protocol (RIP). The system first runs the program in.rdisc, which looks for routers that are running the router discovery protocol. If it finds one such router, in.rdisc continues to run and keeps track of the routers that are running the RDISC protocol.

If the system discovers that routers are not responding to the RDISC protocol, it uses RIP and runs the daemon in.routed to keep track of them.

## PROGRAM 12 – NETWORK SIMULATOR 2 (NS2) : FEATURES & BASIC ARCHITECTURE OF NS2

## 1. What is NS2

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

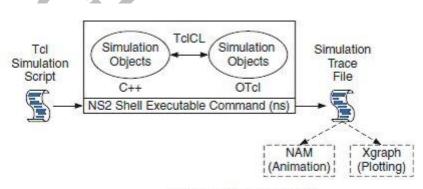
2. Features of NS2

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
- 3. 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.

4. Why two language? (TCL and C++)

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

NS2 uses OTcl to create and configure a network, and uses C++ to run simulation. All C++ codes need to be compiled and linked to create an executable file.

Use OTcl

- For configuration, setup, or one time simulation, or
- To run simulation with existing NS2 modules.

This option is preferable for most beginners, since it does not involve complicated internal mechanism of NS2. Unfortunately, existing NS2 modules are fairly limited. This option is perhaps not sufficient for most researchers.

Use C++

• When you are dealing with a packet, or - when you need to modify existing NS2 modules.

This option perhaps discourages most of the beginners from using NS2. This book particularly aims at helping the readers understand the structure of NS2 and feel more comfortable in modifying NS2 modules.

5. Installing NS2 on windows 7

NS2 builds and runs under windows using Cygwin. Cygwin provides Linux like environment under windows.

**System Requirements:** A computer with C++ compiler. Building full NS2 package requires large memory space approximately 250MB

# I. Steps to install NS 2 on windows 7 are given below

- 1. Download Cygwin from following link https://www.cygwin.com/setup.exe
- 2. Run the downloaded setup.exe and you will see screen shown below click next.

Cygwin Setup		
	Cygwin Net Release Setup Program	
	This setup program is used for the initial installation of the Cygwin environment as well as all subsequent updates. Make sure to remember where you saved it.	
6	The pages that follow will guide you through the installation. Please note that Cygwin consists of a large number of packages spanning a wide variety of purposes. We only install a base set of packages by default. You can always run this program at any time in the future to add, remove, or upgrade packages as necessary.	
	upprove pochoges as necessary.	
	Setup.exe version 2.774	
	Copyright 2000-2012	
	http://www.cygwin.com/	
	< Back Next > Ca	ncel

3. Select option "Install From Internet". If you have already downloaded the package select "Install from local directory" and click next

Cygwin Setup - Choose Installation Type	
Choose A Download Source Choose whether to install or download from the internet, or install from files in a local directory.	E
<ul> <li>Install from Internet (downloaded files will be kept for future re-use)</li> <li>Download Without Installing</li> </ul>	
Install from Local Directory	
< <u>B</u> ack	kt > Cancel

4. Keep the default installation directory as "C:\cygwin" and click next

Select Root Install Directory Select the directory where you want t	to install Cygwin. Also choose a few	E
installation parameters.		
Root Directory		
C:\cygwin		Browse
	in the second	
Install For		
Al Users (RECOMMENDED)		
Cygwin will be available to all users	of the system.	
🗇 Just Me		
Cygwin will still be available to all us	ers, but Desktop Icons, Cygwin Menu En nly available to the current user. Only sele if you have specific needs.	

5. Keep default local package directory as your download folder and click next.

Select Local Package Directory Select a directory where you want Se downloads. The directory will be created	etup to store the install <mark>a</mark> tion f ated if it does not already ex	iles it ist.
Local Package Directory C:\Users\sys27\Downloads		Browse

Department of Computer Science and Engineering,

6. Next screen will ask for your Internet connection type keep it as "Direct connection" and click next and in next screen choose one site to download the packages and click next.

	wnload Site ite from this list, or add your own sites to the list		E
	Available Download Sites:		
	http://mirrors.163.com http://box-soft.com http://cygwin.mirrors.hoobly.com	×	
	ftp://cygwin.mimorcatalogs.com http://cygwin.mimorcatalogs.com http://cygwin.mimors.pair.com http://cygwin.mimors.pair.com http://cygwin.mimors.pair.com http://cygwin.skazkaforyou.com http://cygwin.skazkaforyou.com http://mimors.ymmds.com http://tweedo.com ftp://mimors.mission.com	~	S.
User URL:		Ad	dd

7. In next screen Cygwin will allow to select the packages you want to install

Search		lear	O Keep	⊚ <u>C</u> urr	🔘 Exp	View Cate	egory
Category	Current	New			В	S Size	
🖂 All 🚯 De	fault						:
⊞ Access	sibility 😯 Default						1
🕀 Admin	Default						
🗄 Archive	e 😯 Default						
E Audio	🕑 Default						
🗄 Base 🌢	🕽 Default						
🗉 Databa	ise 😯 Default						
E Debug	Default						
< [	III						

Department of Computer Science and Engineering,

8. Uncheck the option "Hide obsolete packages" then click on "view" button till the word "category" changes to "Full"

Search	Glear	O Keep	<u>و</u>	π	Ego View	Full	
Current	New		B	S	Categories	Size	
	Skip		nja	n/a	Libs		
	Skip		nja	n/a	Devel, Libs, System	1, 1	
3.2.5-1	Keep		n/a		Gnome		
	Skip		nja	n/a	_obsolete		
	<li>Skip</li>		nja	n/a	_obsolete		
	Skip		nía	n/a	Graphics		
	Skip		n/a	n/a	Graphics		
	Skip		n/a	n/a	Graphics		
4			- *-	- **			
* Hide obsolet						,	

To install NS2 you need to select and install following packages:

- gcc
- gcc-g++
- gnuplot
- make
- patch
- perl
- tar
- X-startup-scripts
- xorg-x11-base
- xorg-x11-bin
- xorg-x11-devel
- xorg-x11-bin-dlls
- xorg-x11-bin-lndir
- xorg-x11-etc
- xorg-x11-fenc
- xorg-x11-fnts
- xorg-x11-libs-data
- xorg-x11-xwin
- libxt-devel
- libXmu-devel

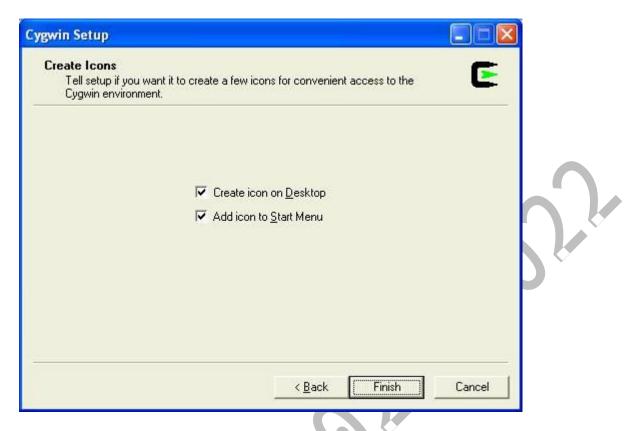
Department of Computer Science and Engineering,

To select a package search the package name and click on word "skip" this will change it to version number of the package. Do this for all above packages and click next to start download and installation

E
Cancel

9. Once installation is complete create desktop icons if you need.





NOTE: If you missed any package while installing Cygwin first time you can install it by running the setup.exe again and selecting the package in step 8.

10. Cygwin installation is complete now you can run Cygwin from desktop and see its interface.

## **II.** Steps is to install NS2

1. Download NS2 from following link: https://www.isi.edu/nsnam/dist/ns-allinone-2.28.tar.gz

2. Decompress the file use winrar. Copy the decompressed folder the Cygwin installation directory under the subdirectory home. It will be C:\cygwin\home\system\_name : where system\_name is name of your system in above Cygwin installation this path will be C:\Cygwin\home\sys27

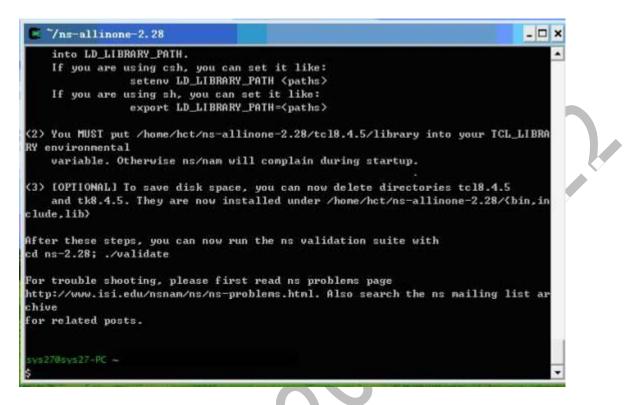
3. Run Cygwin from desktop and change the directory to folder you copied just now in step 2 command to change directory:cd /home/sys27/ns-allinone-2.28

NOTE: please change sys27 to name of your system

4. To start installation type following command:"./install"(WITHOUT qoutes)

This will began the installation process if any Cygwin package is missing it will be reported to you if so the run Cygwin setu.exe and install the missing package and start again from step 2.

Installation is a long process and take quite some time once it is finished you will get a screen as shown below:



#### 5. Add following lines to the .bashrc

export NS\_HOME=/home/sys27/ns-allinone-2.28 export PATH=\$NS\_HOME/nam-1.11:\$NS\_HOME/tcl8.4.5/unix:\$NS\_HOME/tk8.4.5/unix:\$NS\_HOME/bin:\$PATH export LD\_LIBRARY\_PATH=\$NS\_HOME/tcl8.4.5/unix:\$NS\_HOME/tk8.4.5/unix:\$NS\_HOME/otcl-1.9:\$NS\_HOME/lib:\$LD\_LIBRARY\_PATH export TCL\_LIBRARY=\$NS\_HOME/tcl8.4.5/library

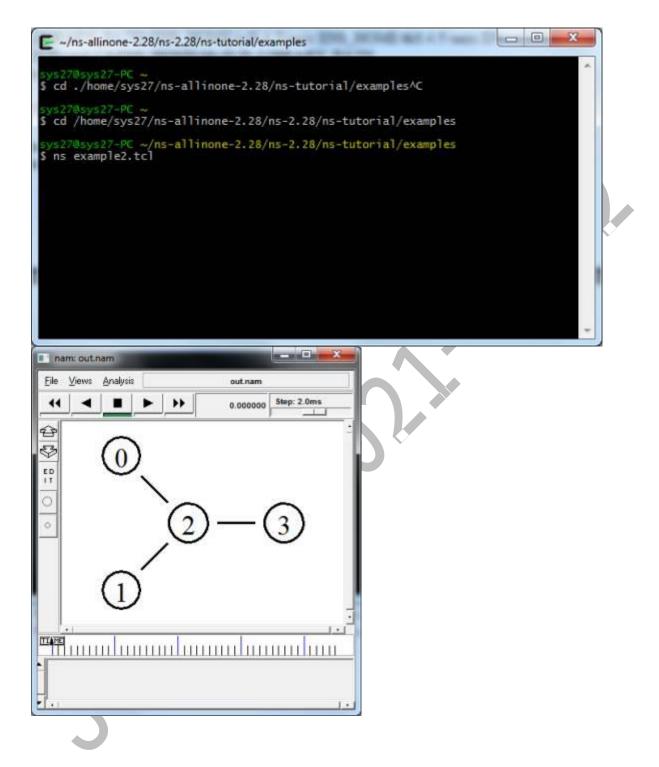
## NOTE: replace sys27 with your system name

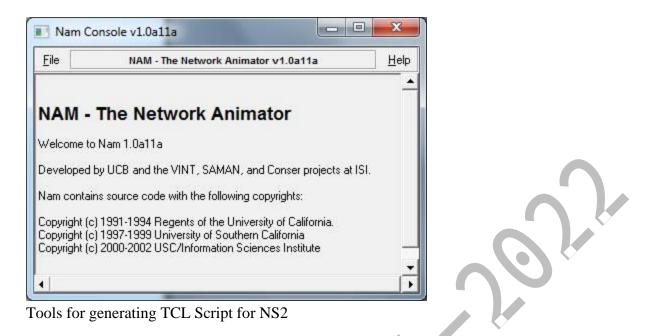
6. To check if NS2 is installed correctly you can run one sample example given in ns-tutorials folder

To run the example change the directory to examples folder: cd ./home/sys27/ns-allinone-2.28/ns-tutorial/examples

Then type following command:

ns example2.tcl





NS2 a very common and widely used tool to simulate small and large area networks. Tcl scripts are widely used in NS-2 simulation tool. Tcl scripts are used to set up a wired or wireless communication network, and then run these scripts via the NS-2 for getting the simulation results. Several tools are available to design networks and generate TCL scripts some of them are discussed below

# I. NS2 scenario Generator (NSG):

Its a java based tool that can run on any platform and can generate TCL scripts for wired and Wireless scenarios for NS2.Main features of NSG are:

- Creating Wired and wireless nodes by drag and drop.
- Creating Simplex and Duplex links for wired network
- Creating Grid, Random and Chain topologies.
- Creating TCP and UDP agents. Also supports TCP
- Tahoe, TCP Reno, TCP New-Reno and TCP Vegas.
- Supports Ad Hoc routing protocols such as DSDV,
- AODV, DSR and TORA.
- Supports FTP and CBR applications.
- Supports node mobility.
- Setting the packet size, start time of simulation, end
- Time of simulation, transmission range and interference
- Range in case of wireless networks, etc.
- Setting other network parameters such as bandwidth, etc for wireless scenarios

# II. Visual Network Simulator (VNS):

This tool is centered on capabilities of NSG. It also provides support to Differentiated Services (DiffServ) scenarios and simple and intuitive set of icons to represent the components of a network. Some features of VNS are given below:

- Adding and configuration of links, agents and traffic sources.
- Modeling network scenarios with support to multicast.
- Selection of a dynamic routing protocol.
- Definition of the simulation output as an animation and/or graphics.
- Edition of the Tcl script generated.
- Saving the defined simulation scenario

# III. NS 2 Workbench

Ns Bench makes NS-2 simulation development and analysis faster and easier for students and researchers without losing the flexibility or expressiveness gained by writing a script. Some features are:

- Nodes, simplex/duplex links and LANs
- Agents: TCP,UDP, TCPSink, TCP/Fack,TCP/FullTcp, TCP/Newreno, TCP/Reno,TCP/Sack1, TCPSink, TCPSink/Sack1, TCPSink/DelAck,
- TCPSink/Sack1/DelAck,TCP/Vegas, Null Agent.
- Applications/Traffic: FTP, Telent, https/Server,https/Client, https/Cache, webtraf, Traffic/CBR,Traffic/Pareto, Traffic/Exponential.
- Services: Multicast, Packet Scheduling, RED, Diff-Serv.
- Creating "Groups" concept to compensate for "loops"
- Scenario generator.
- Link Monitors.
- Loss Models.
- Routing Protocols

# IV. Network Simulation by Mouse (NSBM)

NSBM, developed in java, is a graphical tool that is used to generate TCL script using a mouse. Nodes and links can be created with a single mouse click. You can draw a network topology with multiple nodes with only a few mouse clicks. Afterwards you click on a button and there is the TCL code, almost ready for use with the ns.

NSBM used in order to process the XML configuration data. It must provide many functions, which are specified only in the configuration data at run time. Because the classes are implementation-specific, classes generated by the binding compiler in one JAXB implementation will probably not work with another JAXB implementation. So if you change to another JAXB implementation, you should rebind the schema with the binding compiler provided by that implementation

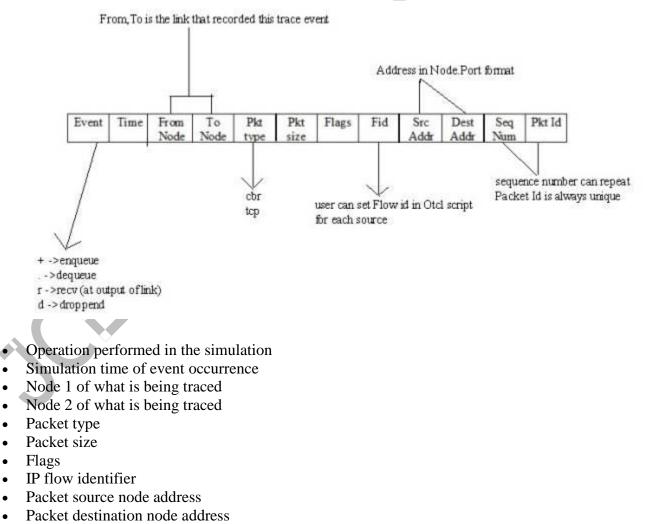
Trace Files Generated in NS2

NS2 currently supports a number of different types of trace files. In addition to its own format, NS2 also has the Nam trace format, which contains the necessary information from the simulation to drive the Nam visualizer. Both of these trace formats are very specific when it comes to giving details about the events that occur during an NS2 simulation.

Traces and monitors represent the only support for data collection in ns-2. Traces record events related to the generation, enqueueing, forwarding, and dropping of packets. Each event corresponds to a line of ASCII characters, which contains information on the event type and the information stored into the packet

NS-2 provides three kinds of formats for wired networks: Tracing, Monitoring and NAM trace file.

I. Tracing: Trace file format is given below:



Sequence number

•

•

•

•

•

Department of Computer Science and Engineering,

• Unique packet identifier

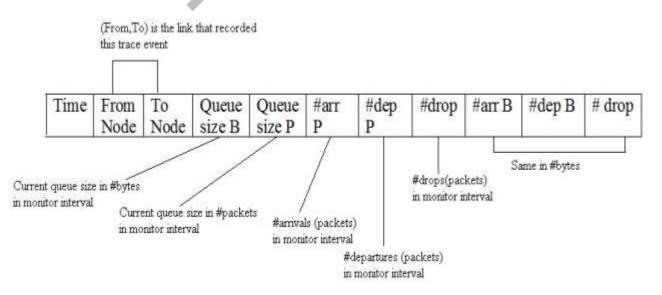
Example of Trace file:

```
+ 4.315145 0 1 ack 40 ----- 0 3.0 6.0 0 1
- 4.315145 0 1 ack 40 ----- 0 3.0 6.0 0 1
r 4.325305 0 1 ack 40 ----- 0 3.0 6.0 0 1
+ 4.325305 1 6 ack 40 ----- 0 3.0 6.0 0 1
- 4.325305 1 6 ack 40 ----- 0 3.0 6.0 0 1
r 4.335465 1 6 ack 40 ----- 0 3.0 6.0 0 1
+ 4.335465 6 1 tcp 1040 ----- 0 6.0 3.0 1 2
- 4.335465 6 1 top 1040 ----- 0 6.0 3.0 1 2
r 4.349625 6 1 top 1040 ----- 0 6.0 3.0 1 2
+ 4.349625 1 0 top 1040 ----- 0 6.0 3.0 1 2
4.349625 1 0 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.363785 1 0 tcp 1040 ----- 0 6.0 3.0 1
                                          2
+ 4.363785 0 4 tcp 1040 ----- 0 6.0 3.0 1
                                          2
- 4.363785 0 4 top 1040 ----- 0 6.0 3.0 1
                                          2
r 4.377945 0 4 tcp 1040 ----- 0 6.0 3.0 1
                                          2
+ 4.377945 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
4.377945 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.392105 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
+ 4.392105 3 4 ack 40 ----- 0 3.0 6.0 1 3
- 4.392105 3 4 ack 40 ----- 0 3.0 6.0 1 3
r 4.402265 3 4 ack 40 ----- 0 3.0 6.0 1 3
```

#### **II.** Monitoring

Queue monitoring refers to the capability of tracking the dynamics of packets at a queue (or other object). A queue monitor tracks packet arrival/departure/drop statistics, and may optionally compute averages of these values. Monitoring was useful tools to find detail information about queue.

Flow monitor trace format is given below:



III. NAM trace files which are used by NAM for visualization of ns simulations. The NAM trace file should contain topology information like nodes, links, queues, node connectivity etc as well as packet trace information. A NAM trace file has a basic format to it. Each line is a NAM event. The first character on the line defines the type of event and is followed by several flags to set options on that event. There are 2 sections in that file, static initial configuration events and animation events. All events with -t \* in them are configuration events and should be at the beginning of the file.

Example of NAM file is:

n -t \* -a 4 -s 4 -S UP -v circle -c black -i black n -t \* -a 0 -s 0 -S UP -v circle -c black -i black n -t \* -a 5 -s 5 -S UP -v circle -c black -i black n -t \* -a 1 -s 1 -S UP -v circle -c black -i black n -t \* -a 6 -s 6 -S UP -v circle -c black -i black n -t \* -a 2 -s 2 -S UP -v circle -c black -i black n -t \* -a 3 -s 3 -S UP -v circle -c black -i black 1 -t \* -s 2 -d 4 -S UP -r 2000000 -D 0.01 -c black 1 -t \* -s 3 -d 4 -S UP -r 2000000 -D 0.01 -c black 1 -t \* -s 4 -d 0 -S UP -r 2000000 -D 0.01 -c black 1 -t \* -s 0 -d 1 -S UP -r 2000000 -D 0.01 -c black 1 -t \* -s 6 -d 1 -s UP -r 2000000 -D 0.01 -c black 1 -t \* -s 1 -d 5 -S UP -r 2000000 -D 0.01 -c black + -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x (6.0 3.0 0 ----- null) - -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x (6.0 3.0 0 ----- null) h -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x (6.0 3.0 -1 ----- null) r -t 4.26434515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x (6.0 3.0 0 ----- null) + -t 4.26434515323951 -s 1 -d 0 -p tcp -e 40 -c 0 -i 0 -a 0 -x (6.0 3.0 0 ----- null) - -t 4.26434515323951 -s 1 -d 0 -p tcp -e 40 -c 0 -i 0 -a 0 -x {6.0 3.0 0 ----- null}

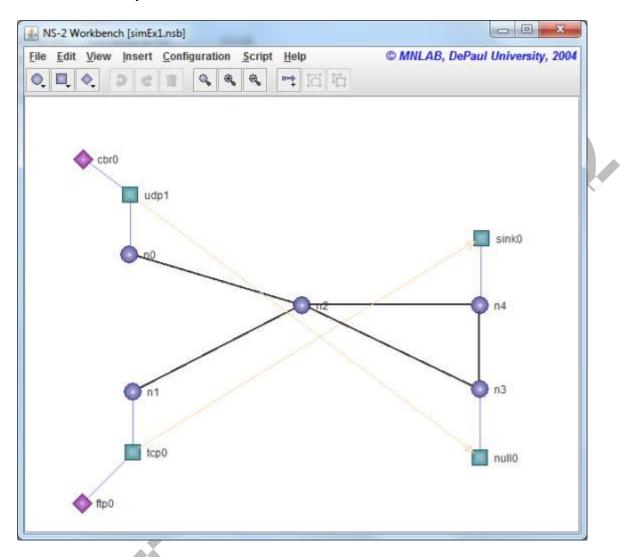
6. Example on NS2

NS workbench is used in following examples to create scenarios and generate TCL scripts, which are then run in NS2 to generate trace file and NAM file

To use NS workbench you will need Java Sdk installed on your system then download ns-bench jar file and execute it to start ns workbench

# Example 1:

The network consists of five nodes n0 to n4. In this scenario, node n0 sends constant bit-rate (CBR) traffic to node n3, and node n1 transfers data to node n4 using a file transfer protocol (FTP). These two carried traffic sources are carried by transport layer protocols User Datagram Protocol (UDP) and Transmission Control Protocol (TCP), respectively. In NS2, the transmitting object of these two protocols are a UDP agent and a TCP agent, while the receivers are a Null agent and a TCP sink agent, respectively.



Network created by ns workbench is shown below

Then Generate the TCL script and save it. Script generated is given below:

# Creating New Simulator set ns [new Simulator]

# Setting up the traces
set f [open outEx1.tr w]
set nf [open outEx1.nam w]
\$ns namtrace-all \$nf
\$ns trace-all \$f

proc finish { } {

global ns nf f \$ns flush-trace

Department of Computer Science and Engineering,

```
puts "Simulation completed."
  close $nf
  close $f
  exit 0
#
#Create Nodes
#
```

```
set n0 [$ns node]
   puts "n0: [$n0 id]"
set n1 [$ns node]
    puts "n1: [$n1 id]"
set n2 [$ns node]
   puts "n2: [$n2 id]"
set n3 [$ns node]
   puts "n3: [$n3 id]"
set n4 [$ns node]
   puts "n4: [$n4 id]"
```

```
#
```

}

```
#Setup Connections
#
```

\$ns duplex-link \$n0 \$n2 100Mb 5ms DropTail \$ns duplex-link \$n2 \$n4 54Mb 10ms DropTail \$ns duplex-link \$n1 \$n2 100Mb 5ms DropTail \$ns duplex-link \$n2 \$n3 54Mb 10ms DropTail \$ns queue-limit \$n2 \$n3 40 \$ns simplex-link \$n3 \$n4 10Mb 15ms DropTail \$ns simplex-link \$n4 \$n3 10Mb 15ms DropTail

#### #

#Set up Transportation Level Connections #

set tcp0 [new Agent/TCP] \$ns attach-agent \$n1 \$tcp0

set udp1 [new Agent/UDP] \$udp1 set dst\_addr\_ Unicast \$udp1 set fid\_1 \$ns attach-agent \$n0 \$udp1

Department of Computer Science and Engineering,

set null0 [new Agent/Null]
\$ns attach-agent \$n3 \$null0

set sink0 [new Agent/TCPSink]
\$ns attach-agent \$n4 \$sink0

# #Setup traffic sources #

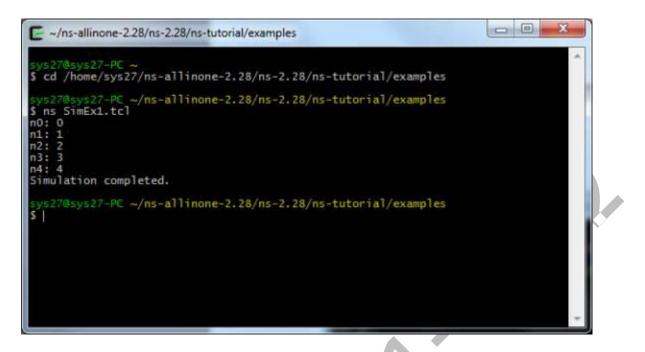
set ftp0 [new Application/FTP]
\$ftp0 attach-agent \$tcp0

set cbr0 [new Application/Traffic/CBR] \$cbr0 set rate\_ 2Mb \$cbr0 set packetSize\_ 1000 \$cbr0 attach-agent \$udp1 \$ns connect \$udp1 \$null0 \$udp1 set fid\_ 0 \$ns connect \$tcp0 \$sink0 \$tcp0 set fid\_ 1

# #Start up the sources #

\$ns at 0.05 "\$ftp0 start" \$ns at 0.1 "\$cbr0 start" \$ns at 60.0 "\$ftp0 stop" \$ns at 60.5 "\$cbr0 stop" \$ns at 61.0 "finish" \$ns run

Then to run the script in NS2 start Cygwin and change directory path to folder you saved your TCL script in and then use ns command to run the script.



Once the simulation is completed two files OutEx1.tr and OutEx1.nam will be generated in the same folder.

```
+ 4.315145 0 1 ack 40 ----- 0
                               3.0 6.0
                                      0
- 4.315145 0 1 ack 40 ----- 0 3.0 6.0 0 1
r 4.325305 0 1 ack 40 ----- 0 3.0 6.0 0 1
+ 4.325305 1 6 ack 40 ----- 0 3.0 6.0 0 1
- 4.325305 1 6 ack 40 ----- 0 3.0 6.0 0 1
r 4.335465 1 6 ack 40 ----- 0 3.0 6.0 0 1
+ 4.335465 6 1 tcp 1040 ----- 0 6.0 3.0 1 2
- 4.335465 6 1 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.349625 6 1 tcp 1040 ----- 0 6.0 3.0 1 2
+ 4.349625 1 0 tep 1040 ----- 0 6.0 3.0 1 2
- 4.349625 1 0 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.363785 1 0 tcp 1040 ----- 0 6.0 3.0 1 2
+ 4.363785 0 4 tcp 1040 ----- 0 6.0 3.0 1 2
- 4.363785 0 4 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.377945 0 4 tcp 1040 ----- 0 6.0 3.0 1 2
+ 4.377945 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
- 4.377945 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
r 4.392105 4 3 tcp 1040 ----- 0 6.0 3.0 1 2
+ 4.392105 3 4 ack 40 ----- 0 3.0 6.0 1 3
- 4.392105 3 4 ack 40 ----- 0 3.0 6.0 1 3
r 4.402265 3 4 ack 40 ----- 0 3.0 6.0 1 3
```

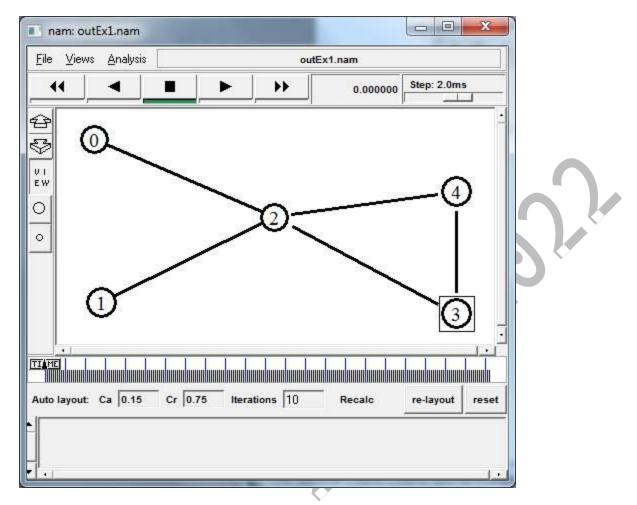
Content of Nam file

```
n -t * -a 4 -s 4 -S UP -v circle -c black -i black
n -t * -a 0 -s 0 -S UP -v circle -c black -i black
n -t * -a 5 -s 5 -S UP -v circle -c black -i black
n -t * -a 1 -s 1 -S UP -v circle -c black -i black
n -t * -a 6 -s 6 -S UP -v circle -c black -i black
n -t * -a 2 -s 2 -S UP -v circle -c black -i black
n -t * -a 3 -s 3 -S UP -v circle -c black -i black
1 -t * -s 2 -d 4 -S UP -r 2000000 -D 0.01 -c black
1 -t * -s 3 -d 4 -S UP -r 2000000 -D 0.01 -c black
1 -t * -s 4 -d 0 -S UP -r 2000000 -D 0.01 -c black
1 -t * -s 0 -d 1 -S UP -r 2000000 -D 0.01 -c black
1 -t * -s 6 -d 1 -S UP -r 2000000 -D 0.01 -c black
1 -t * -s 1 -d 5 -S UP -r 2000000 -D 0.01 -c black
+ -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x
{6.0 3.0 0 ----- null}
- -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x
{6.0 3.0 0 ----- null}
h -t 4.25418515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x
(6.0 3.0 -1 ----- null)
r -t 4.26434515323951 -s 6 -d 1 -p tcp -e 40 -c 0 -i 0 -a 0 -x
(6.0 3.0 0 ----- null)
+ -t 4.26434515323951 -s 1 -d 0 -p tcp -e 40 -c 0 -i 0 -a 0 -x
(6.0 3.0 0 ----- null)
- -t 4.26434515323951 -s 1 -d 0 -p tcp -e 40 -c 0 -i 0 -a 0 -x
{6.0 3.0 0 ----- null}
```

To run the Network Animator in the finish procedure of the script call nam.exe with name file name as

proc finish {} { global ns nf f \$ns flush-trace close \$nf close \$f exec nam outEx1.nam & exit 0

}



7. Advantages and Disadvantages of NS2

# Advantages

- Cheap- Does not require costly equipment
- Complex scenarios can be easily tested.
- Results can be quickly obtained-more ideas can be tested in a smaller time frame.
- Supported protocols
- Supported platforms
- Modularity
- Popular

# Disadvantages

- Real system too complex to model. i.e. complicated structure.
- Bugs are unreliable