NETWORK TRAFFIC ANALYZER

By

T.PRADEEPRAJ

Reg. No 71202702009

of

Kumaraguru College of Technology

A PROJECT REPORT

Submitted to the

FACULTY OF SCIENCE AND HUMANITIES

In partial fulfillment of the requirements

for the award of the degree

of

MASTER OF SCIENCE

IN

APPLIED SCIENCE-COMPUTER TECHNOLOGY

June, 2004

BONAFIDE CERTIFICATE

Certified that this project report titled

NETWORK TRAFFIC ANALYZER

IS A BONAFIDE WORK OF

Mr. T.PRADEEPRAJ (Reg. No: 71202702009)

Who carried out the research under my supervision.

Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Project Guide

Head of the Department

The Candidate with University Register No. 71202702009 was examined by us in the project Viva-Voce examination held on 17.06.04

Internal Examiner

External Examiner

DCL SOFTWARE LTD.

(Formarly DC ELCOT SOFTWARE LTD.)
(A JOINT VENTURE OF ELCOT, A GOVT. OF TAMIL NADU ENTERPRISE)

TO WHOMSOVER IT MAY CONCERN

This is to certify that Mr.T.Pradeep Raj, Final year M.Sc(Applied Science-Computer Technology) Roll No.01 MC T 09, student of Kumaraguru College of Technology, Coimbatore has been involved in the Project titled "NETWORK TRAFFIC ANALYZER" from January 2004 to April 2004 at our concern.

His performance and conduct during the project was appreciable.

For DCL SOFTWARE LTD.

/DIRECTOR.

Place:Chennai - 20
Date:31.05.2004

"Padmini Complex", New No. 85, (Old No. 37), First Main Road, Gandhi Nagar, Adyar,
Chennai - 600 020, India. ✆ : 2490 8505, 2441 5825 1 - 44 - 2445 3479 E-mail : dcl@vsnl.com

ABSTRACT

The Network Traffic Analyzer allows the user to examine data from a live network or from a captured file on disk. The user can interactively browse the captured data, viewing summary and detail information such as source address, destination address, time, Traffic name and the encrypted information of each packet. This analyzer can be implemented in LAN and WAN.

A Traffic analyzer is the only tool that shows you exactly what is happening on the LAN. Once a problem is isolated and recorded, there can be no denying which vendor, or which system is the cause. For example, if the TCP/IP sessions are "hanging", a Traffic analyzer can show which system sent the last packet, and which system failed to respond. If you are experiencing slow screen updates, a Traffic analyzer can display delta time stamps and show which system is waiting for packets, and which system is slow to respond.

This Proposed System contains a GUI (Graphical user interface) based interface. So that, the user can 'see' where packets are going, and where they're not. This proposed software system, captures conversation between two or more systems or devices. The Traffic analyzer not only captures the traffic, it also decodes (interprets) the traffic. Also provide statistics and trend information on the captured traffic.

Whenever the user has run this software "NETWORK TRAFFIC ANAYLSER IN LINUX", a browser window will be opened with the following details such as Source and the Destination IP address of each packet, and the Traffics used for transferring and the encrypted Information.

The browser also has other option in which the user can choose a particular Traffic or a group Traffic that flows among the network. It is also have coloring option for manipulation of a particular Traffic in the network.

The Traffic analyzer can show runaway traffic (broadcast or multicast storms) and its origin, system errors and retries, and whether a station is sending, trying to send, or only seeming to communicate. We will get information that is otherwise unavailable, which results in more efficient troubleshooting and better LAN/WAN health.

This project is extremely useful for the Linux users in the network and makes them to use network efficiently.

ACKNOWLEDGEMENT

At this pleasing moment of having successfully completed the project work, I wish to acknowledge my sincere gratitude and heartfelt thanks to our beloved Principal **Dr.K.K.Padmanabhan B.Sc (Engg), M.Tech, Ph.D.** for having given me the adequate support and opportunity for completing this project work successfully.

I express my sincere thanks to **Dr. S.Thangasamy, Ph.D.** the ever active and sympathetic, Head of the Department of Computer science & Engineering, who with his careful supervision has ensured me in attaining perfection of work.

I extend my sincere thanks to **Mr. R. Ranganathan Dinesh M.S,** Project Coordinator for rendering us all the timely helps through out the project.

I regard my heartfelt thanks and everlasting gratitude to my Project Guide **Mrs. L.S.Jayashree M.E;** Senior Lecturer, Department of Computer Science & Engineering for her uplifting ideas, inspiring guidance and valuable suggestions which have been very helpful in refining upon the project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

ARP	Address Resolution Protocol
ARIN	American Registry for Internet Numbers
ARPANET	Advanced Research Projects Agency Network
ASCII	American Standard Code for Information Interchange
ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
BSD	Berkeley Software Development
CCITT	International Telegraph and Telephone Consultative Committee
CIX	Commercial Internet Exchange
CDPD	Cellular Digital Packet Data protocol
CSLIP	Compressed Serial Line Internet Protocol
DARPA	Defense Advanced Research Projects Agency
DDP	Datagram Delivery Protocol
DDS	Digital data service
DNS	Domain Name System
DOCSIS	Data Over Cable System Interface Specification
DoD	U.S. Department of Defense
DWDM	Dense Wave Division Multiplexing
FAQ	Frequently Asked Questions lists
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
FYI	For Your Information series of RFCs
GOSIP	U.S. Government Open Systems Interconnection Profile
HDLC	High-level Data Link Control
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IAB	Internet Activities Board

IANA	Internet Assigned Numbers Authority
ICANN	Internet Corporation for Assigned Names and Numbers
ICMP	Internet Control Message Protocol
IESG	Internet Engineering Steering Group
IETF	Internet Engineering Task Force
IMAP	Internet Message Access Protocol
InterNIC	Internet Network Information Center
IP	Internet Protocol
IPX	Internet work Packet Exchange
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISOC	Internet Society
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
MAC	Medium (or media) access control
Mbps	Megabits (millions of bits) per second
NICNAME	Network Information Center name service
NSF	National Science Foundation
NSFNET	National Science Foundation Network
NTP	Network Time Protocol
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PING	Packet Internet Groper
POP3	Post Office Protocol v3
PPP	Point-to-Point Protocol
RADIUS	Remote Authentication Dial-In User Service
RARP	Reverse Address Resolution Protocol
RIP	Routing Information Protocol
RFC	Request for Comments
SDH	Synchronous Digital Hierarchy

SLIP	Serial Line Internet Protocol
SMDS	Switched Multimegabit Data Service
SMTP	Simple Mail Transfer Protocol
SNAP	Sub network Access Protocol
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Network
SSL	Secure Sockets Layer
STD	Internet Standards series of RFCs
TACACS+	Terminal Access Controller Access Control System plus
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TLD	Top-level domain
UDP	User Datagram Protocol
WAP	Wireless Application Protocol
DSL	Digital Subscriber Line family of technologies

CHAPTER 1

INTRODUCTION

1. 1 What is Network Traffic Analyzer?

The main objective of this proposed project "NETWORK TRAFFIC ANAYLZER IN LINUX" is to analyze the network such that to provide efficient data transmission rate from one machine to another machine in any type of the network in the easy and reportable format using the GUI.

Normally the Data transmission can be done with any type of protocol in the network, but the efficiency of the protocol in data transfer cannot be checked with present Linux Operating System's Utilities. Moreover the Linux Operating system does not have any software or the GUI based applications to interactively analyze the data flow rates and the data transmission in the network.

In the Network, there are many hundreds of protocols for the data transmission. Each protocol has its own standard and syntax. The data transmissions are done based on the rules in the protocol standard. Based on the factors like time, day and traffic of the data transmission flow rate of each protocol is analyzed. Such that we are able to see the live network traffic, hence it is very helpful for the network developers to view the exact contents of a network conversation and manage the traffic.

The Proposed Software system **Network Traffic Analyzer in Linux** has details and functions manipulation over the present exiting protocols.

In this project **Network Traffic Analyzer in Linux** the data transfer rate analysis is done in LAN, MAN and WAN. The analysis is mainly done based on the flow rate of different protocols in the entire network. The standards and syntax of the protocols are maintained with out any disturbance.

This proposed project also has many other functions such as the verification of the Flow rate of all the protocols in the network or of the single or a more than one protocols present in the Network. Whenever the user runs this software **Network Traffic Analyzer in Linux**, a browser window will be opened with the following details of the Source Machine and the Destination Machine IP address, Protocol using for the transferring and it's encrypted Information. The Browser also gives the option like File, Edit, Saving and Filter option. When the user clicks start in the capture menu, capturing of packet starts that flow over the entire network. The packets are captured based on the protocol and are listed.

The browser also has other option in which the user can choose a particular protocol or a group protocols flows among the network. It also has coloring option to highlight the particular protocol that we need.

The main browser can have different options for finding the protocol or frame that flow in the networks. Its frame details and the packets flow across the network. The browser also has other details such as manipulation over the

particular protocol details and its flow rate. It also consists of the timing details about the protocols used.

Thus, this software helps the user to have very easy interaction for finding the details for the transferred data with its encrypted data contains in it along with the Packet , Socket , Frame and its extreme bits in it as the Binary Form. It also tells the user about the total packets flows in the network. The Software also tells to the user about total packet loss and its ways in which it make to loss.

This software provides facilities like saving the recorded manipulations in the text file with all the details as that are see in the network. This software **Network Traffic Analyzer in Linux** plays a big role in terms of user friendliness and data transmission with efficient flow rates handling of various transactions over the different network.

CHAPTER 2

SYSTEM ANALYSIS

2.1 Need for the Project

A Network analyzer is an invaluable piece of software that takes the strain out of managing a network. Using an easy to use interface we can see where our packets are going... and where they're not. Using features such as top-talker, a protocol analyzer will tell we who or what is using up valuable network resources, when, where and why.

The Network Analyzer captures conversations between two or more systems or devices. A Network analyzer not only captures the traffic, it also decodes (interprets) the traffic. The decoding allows us to view the conversation in English, as opposed to binary language. A sophisticated protocol analyzer will also provide statistics and trend information on the captured traffic.

Network Analyzer provides information about the traffic flow on our LAN/WAN, from which we can view device-specific information. Unlike SNMP-based management consoles, protocol analyzers are device independent.

A Network Analyzer is the only tool that shows us exactly what is happening on your LAN/WAN. Once a problem is isolated and recorded, there can be no denying which vendor, or which system is the cause.

The Network Analyzer should provide three main sources of information about your LAN traffic. The Network Statistics about traffic flow, network or station line errors. This information helps to identify trends and general conditions that may signal an unexpected network problem condition or a load issue that is causing slowdowns.

Additionally if we are considering adding a switch, the statistical traffic breakdown can show how best to implement the new switch. If we currently have a switch installed, statistical analysis can show if the switch is configured correctly and the ports are supporting a balanced load of LAN traffic.

Packet Capture and Decode displays LAN traffic (packets) decoded into specific function and sub-function for LAN or protocol problem isolation. Being able to view the specific packet-by-packet conversion can show exactly what is happening during a system-to-system communication, when both things are functioning correctly and when things are not.

Trending Information displays historical usage data over days, weeks, months or even years. This information provides a historical perspective on any new problem, and can show trends that may indicate a potential problem before it happens.

2.2 SNMP Implementation:

The SNMP Protocol is not need implement in the Protocol Analyzer due to the SNMP products provide device specific information, where protocol analyzers obtain all their information by examining the traffic on the LAN. For example, an SNMP collection utility could not provide session delta time stamps for a UNIX telnet session, nor can SNMP provide bandwidth utilization statistics directly.

Example SNMP statistics would include how many packets came in or went out of a router, a print server's IP address, or a predefined trap generated by a network printer for "out of paper". SNMP products are a good complement to any protocol analyzer.

The Current Traffic Analyzers does not able to see all the segments of the Entire Network and the current protocol analyzer work over a WAN. The Current Traffic Analyzers can only view and collect traffic from the segment where the analyzer is located.

To capture and analyze traffic from another segment (local multi-segment LAN or remote WAN), a distributed or multi-segment analyzer is required. Distributed analyzers offer similar functionality to a standard (non-distributed) analyzer, displaying multiple diagnostic windows, each representing a segment on your LAN - all from a single management station.

Typically, distributed analyzers consist of a software based management station and either software or hardware based probes allowing an administrator to "view" any segment that hosts a probe. The Current protocols are very much interacting with intranet. If our switch supports port mirroring, we can turn on mirroring of one or more of the switched ports to the span port which the analyzer is plugged in to.

Using a Traffic analyzer in a switched environment is common, and can provide both global port balancing information (using station statistics) and specific conversation troubleshooting information (using packet capture and decode). In most switched environments using an analyzer is as simple as placing the tool on a server to collect access and conversational data to and from that server.

Placing the analyzer on a "downstream" hub can show if the hub's users are correctly placed to maximize the aggregate throughput of the switch. Most switches allow for port tapping directing any port's traffic to the port where the protocol analyzer is installed.

Internet Packet Exchange (IPX) is the networking protocol used by Novell in their NetWare product. Technically, IPX is roughly equivalent to IP in TCP/IP. Additional protocols, such as sequenced packet exchange (SPX equivalent to TCP) round out the Novell network stack. The **NetWare Core Protocol (NCP)** is NetWare's file sharing mechanism. It is one of the primary tools in the equivalent of the application layer of the Novell networking stack.

IBM designs its own networking protocol, which it called the **Network Basic Input/Output System (NetBIOS).** Microsoft eventually licensed NetBIOS from IBM, and now NetBIOS is called **NetBIOS Extended User Interface (NetBEUI),** and serves to encapsulate NetBIOS data on a network.

A NetBIOS network is organized quite differently from a TCP/IP network. Rather than TCP/IP's 32-bit numeric IP addresses, subnets, and DNS system, NetBIOS and NetBEUI organize a network in two ways.

First, the network is broken into one or more workgroups, each of which is given a unique name. Within each workgroup, each computer has its own name. Computer talks to each other using their names, and they know each other's names because they periodically announce them to the entire workgroup.The naming and limited organizational features of NetBIOS/NetBEUI also limit its expandability. In addition, with the advent of the Internet, Microsoft found it desirable to shift its networking emphasis to TCP/IP. Completely abandoning NetBIOS was undesirable for legacy reasons, however. One approach to this competing set of needs is to transmit NetBIOS over TCP/IP, and Microsoft and IBM have both chosen to support this approach. Linux doesn't support "raw" NetBIOS, but it does not support it when it's encapsulated in TCP/IP. This is the basis for samba, Linux's approach to file sharing with Windows.

CHAPTER 3

PROJECT DEFINITION

3. 1 Hardware and Software Requirements

The software and hardware requirements for this proposed project are,

3. 1. 1 Hardware Requirements

This is been a Network based Project for which the Hardware Requirements also considered to be more important. The efficient check and the analysis of the flow rate of the various protocols that flows over different systems in the Network are analyzed. The Hardware requirements that are needed for the proposed system networking project as follows:

Network	: Any Network such as Novel, Linux, Windows Based Star or linear Network Structure at least of 25 Clients and a Server.

Client Machine

Operating System	: Any Operating System with Networking enable And Supportable.
Memory	: Minimum of 16 MB RAM.
Bus Architecture	: Minimum of 16 BIT Architecture
Display card	: VGA Display Card.

Server Machine

Operating System	: Linux operating system with network enabled
Kernel Supports	: Linux Kernel 2.2.14 and above.
Memory	: Minimum of 16 MB RAM.
Bus Architecture	: Minimum of 32 BIT Architecture.
Display card	: VGA Display Card.
Hard Disk	: Minimum of 10 GB.
Network Card	: 3COM Network Card.
Network Interface	: Switch, Hub or Router for the Networking Interface for Networking between Clients.
Software	: GLADE and C Compiler with GCC Libraries.

3. 1. 2 Software Requirements:

Operating System	:	Linux Operating System.
Programming Language	:	C, Shell Programming and Glade Programming GTK (GIMP Tool Kit).
Libraries	:	GCC Version 2, BASH Version 3, GTK Programming Libraries Version 4
Software	:	Glade in Linux Operating System, to design GUI.
Protocols Libraries	:	Linux Operating System's all Supportable Protocols and its Libraries.

3. 2. Introduction To Linux

3. 2. 1. Linux's History

Linux was developed as a freely distributable version of UNIX. UNIX is the most widely used operating system in the world and has long been the standard for high-performance workstations and larger servers. UNIX, first developed in 1969, has a strong programmer-oriented user group that supports the operating system. Because UNIX is a commercial product, it must be bought for each platform it runs on. Licensing fees for UNIX versions for PC machines range from a few hundred dollars to several thousand. In an attempt to make UNIX widely available for no cost to those who want to experiment with it, a number of public domain UNIX systems have been developed over the years.

One of the early UNIX workalikes was Minix, written by Andy Tanenbaum. Although Minix didn't have a full range of features, it provided a small operating system that could be used on PC machines. To expand on Minix, a number of users started developing an enhanced operating system that would take advantage of the 80386 CPU's architecture. One of the primary developers of this system, which became known as Linux, was Linus Torvalds of the University of Helsinki. He released an early version of Linux in 1991. A first commercial, almost bug-free release was unleashed to the programming community in March 1992.

Soon, many programmers were working on Linux, and as the challenge and excitement of producing a growing UNIX work alike caught on; Linux grew at a remarkable rate. As the number of developers working on Linux grew, the entire

In March 1991 Linus Benedict Torvalds bought the multitasking system Minix for his AT 386. He used it to develop his own multitasking system which he called Linux. In September 1991 he released the first prototype by e-mail to some other Minix users on the internet, thus beginning the Linux project. Many programmers from that point on have supported Linux. They have added device drivers, developed applications, and aimed for POSIX compliance. Today Linux is very powerful, but what is best is that it's free.

Linux is a freely distributed, multitasking, multi-user operating system that behaves like UNIX. Designed specifically for the PC, Linux takes advantage of the PC's architecture to give you performance similar to UNIX workstations of a couple of years ago. Linux isn't a small, simple operating system like DOS (even in its latest incarnations). The development of UNIX has resulted in a mish-mash of files and directories, all of which are carried over to Linux for compatibility and programming reasons. Linux includes a bunch of files for the operating system itself (called the kernel), a ton of utility programs, documentation files, add-on emulators for other operating systems, and much more.

Linux's Kernel

Linux is a complete multitasking, multi-user operating system that behaves like the UNIX operating system in terms of kernel behavior and peripheral support. Linux has all the features of UNIX, plus several recent extensions that add new versatility to Linux. All source code for Linux and its utilities is freely available.

Memory management is especially strong with the 80386 (compared to earlier CPUs). A floating-point emulation routine allows Linux to function on machines that do not have math coprocessors (such as the SX series of Intel CPUs).

Linux allows shared executables so that if more than one copy of a particular application is loaded (either by one user running several identical tasks, or several users running the same task), all the tasks can share the same memory. This process, called copy-on-write pages, makes for much more efficient use of RAM.

The Linux kernel also supports demand paging, which means that only sections of a program that are necessary are read into RAM. To further optimize memory usage, Linux uses a unified memory pool. This pool enables all free memory on the system to be used as disk cache, effectively speeding up access to frequently used programs and data. As memory usage increases, the amount of cache is automatically adjusted.

Linux uses dynamically shared libraries extensively. Dynamically shared libraries use a common library section for many different applications, effectively reducing the size of each application. Linux does allow full library linking (called statically linked libraries) for portability to machines that may not have the dynamic libraries.

To make Linux widely acceptable, it supports a number of different file systems, including those compatible with DOS and OS/2. Linux's own primary file system, called ext2fs, is designed for optimal use of the disk.

Linux is ideally suited for application development and experimentation with new languages. Several different compilers, including C, C++, Fortran, Pascal, Modula-2, LISP, Ada, Basic, and Smalltalk, come with the distribution software. Many of the Linux compilers, tools, debuggers, and editors are from the Free Software Foundation's GNU project.

(i) GNU software

GNU (a recursive acronym for Gnu's Not UNIX) was developed by the Free Software Foundation (FSF) to provide royalty-free software to programmers and developers. Since it was created, many programmer packages and toolkits have been developed and assigned to FSF for distribution. Most of the GNU software mirrors (and often improves upon) commercially available software.

Linux includes many GNU utilities, including the languages mentioned earlier, debuggers, and compiler tools. Text processors, print utilities, and other GNU tools are also included with most Linux distributions. As more software becomes available from FSF, it can be ported and compiled under Linux because Linux behaves as a standard UNIX operating system.

(ii) DOS Interface

Because Linux is designed for PCs, some compatibility with Microsoft MS-DOS is naturally part of the operating system. Linux provides a DOS emulator, which allows many DOS applications to be executed directly from within Linux, as part of the distribution system. Don't expect complete portability of DOS applications, though, as some applications are written to access peripherals or disk drives in a manner that Linux can't handle. The WINE (WINdows Emulator) project has developed a Microsoft Windows emulator for Linux, which enables Windows applications to be run from within Linux.

Although Linux can emulate DOS and Windows, the emulation feature is not intended to support full DOS usage. Instead, it provides the occasional DOS user with the ability to run an application under Linux. For heavy DOS use, your

system should be set up with both DOS and Linux in separate partitions, enabling you to enter either one at boot time.

Linux does allow you to transfer files seamlessly between the Linux file system and DOS by accessing the DOS partitions on a hard disk directly, if so on figured. This capability makes it easy to move files and applications back and forth between the two operating systems.

(iii) TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) is the primary networking system used by UNIX and Linux. TCP/IP is a full family of protocols that were developed for the Internet, and you must use TCP/IP when you venture out onto the Internet. If you want to connect with other UNIX machines, you will probably have to use TCP/IP as well. The Linux TCP/IP implementation provides all the networking software and drivers usually associated with a commercial UNIX TCP/IP package. With this implementation, you can create your own local area network (LAN), attach to existing Ethernet LANs, or connect to the Internet.

Linux Operating System:

Linux is a most efficiently working Operating System concerned with many servers along with many different protocols and free from cost. It is an operating system of dazzling complexity with source code. Linux is a Multi-user, Multi-Tasking operating system from the ground up which has flexibility and responsibility, far beyond of any of other operating system.

The main reason for selecting this Linux Operating System is due to the reason that there is no specified GUI based interaction and the maintenance software for the connecting the particular systems with the network or in the subnets or in the workgroups. The important reason for Linux Operating System became famous is due that it is easy to use and compact in size.

At present, In the Linux Operating System, the connection establishments, flow of connectivity, maintenance over the different systems and even the manipulation of the files along with data transfer controls is done through command prompt only. Basically Linux is the most powerful Operating System. The Linux Operating System's Core is the Kernel is compact in size and portable to any type of machine and the hardware.

The Linux Operating System has many features. The Following are the important features of Linux:

1. Full Multitasking:

Multiple tasks can be performed at the same time. Linux is operating systems that can able to maintenance over many tasks effectively with handling of many threads in it.

2. Virtual Memory:

One of the important features of the Linux is Virtual Memory. Linux is safely uses a partition of the hard disk as the virtual memory, which increases the efficiency of the system by keeping the active part in RAM and placing less frequently used or inactive process in memory on the disk (swap memory).

3. Hardware Support:

The Best feature of the Linux, especially Intel based versions supports nearly all hardware architectures and devices, with best support for legacy hardware. If there is any new hardware is added, it will seek and maintained over the plug and play software.

4. Linux Kernel:

The Best feature of the Linux, especially its Kernel. The Kernel is core and compact is the size and portable in the nature. The Linux Kernel is developed by C and C++ Programs.

5. The X-Window System:

The x-window system is the graphical system for LINUX. It is powerful interface which support many applications that are based on the GUI. There are four types of desktops there in Linux.

They are as follows:

1. Gnome
2. KDE
3. Window Maker
4. fvwm

6. Build In Networking Support:

The foremost feature of Linux is its Networking Support. The Linux Operating systems has the default networking interfacing and the administering over the Networking capacities.Linux Operating System uses all the maximum standard protocols, including

1. TCP/IP protocols,
2. Network File System (NFS),
3. Network Information Services (NIS),
4. Session Message Block (SMB)
5. Net BIOS Protocols and others.

7. Shared Libraries:

In Linux, each function, each command and the system calls has common library of subroutine it can at run time.

8. Compatibility with the IEEE Posix.1 Standard:

Because of this compatibility of multitasking, multiple accessing along with the shared libraries, Linux supports many of the standards set forth for all UNIX system. This make Linux so compatibility with the IEEE POSIX.1 standards.

9. Open Source Code:

Linux is Internet operating system. It is developed by a group of interested programming people all over the world. Any one who have interested can develop the application and contribute it for the Linux Operating System. Thus it makes the operating system development as open source for all the people.

10. Lower Cost:

The greatest feature of Linux operating system is that it is free of Cost. It can be downloaded from the Internet as it provided freely. The books about Linux will provide the free copy of Linux.

11. Gnu Software Support:

Linux can run a wide range of free software available through the GNU project. This software includes every thing from programming tools, such as compilers, assemblers, linkers and loaders, to system administration utilities, such as stream editors, the venerable emacs editors and games.

3.2.2 GLADE:

GLADE is the GUI based software used in this project for front end design. GLADE is a C/C++ Integrated Developing Environment. This software is highly intensive for the developments of GUI based applications.

Using this GLADE software, the designing of the window frame, this is used for the GUI representation of the network representations. This GLADE Software is developed with the GTK (GIMP- Graphics Image Manipulation Processing Tool Kit) programming language and GDK (GIMP- Graphics Image Manipulation Processing Developing Kit) programming languages.

The GLADE Software in Linux works as that of a software in the Windows product called VC++ of Visual Studio. The GLADE environment is used to implement our widgets and the other main windows widgets. The GUI interface with C/C++ programs is the software itself.

This GLADE software has provided a facility of developing and deploy the coding along the programming done by itself. The user can also develop or manipulate over the coding for the programming widgets to the creation of the GUI based applications and software.

This GLADE software is does not need not of separate compilation of the C/C++ programs and other compilations.

We can develop all the required source code in the GLADE environment itself. So this GLADE software is used in this project for making easy GUI interfaces and it much easy to implement a project or program in Linux Operating System and Environment. This software is also very much easier to use.

Another important advantage of using this software is that it provides good and efficient network support. The GLADE Software is come along with in the Linux Operating System Installation CD and the Source code for this GLADE Software is downloaded from Internet.

The GLADE is need not to separately install in the Linux Operating System. Thus makes the job simpler for implementation and developing the GUI based applications and projects. in Linux OS It is also very easy to develop and deploy a small and a complex project in the C/C++ IDE Environment using the GLADE, is the fore factor for using this software.

One of the important features of this GLADE Software is that it also provided a utility for interfacing with the applications related with databases connectivity and other interactions such as designing tools as that of QT support developments and deployments.

Another one of the important features of this GLADE Software is that it also provided a utility for interfacing with the applications related with C, C++ , Effile , ADA Languages.

3.2.3 Shell Programming

The Shell Programming is the greater sophisticated programming in the Linux Operating System. This Shell Programming is the programming interface over with Linux Operating Systems and the users of the Operating System. The Shell is a place where to run the program for interfacing with Linux Operating System and the user interfacing done over by the system calls.

The Linux Operating Systems has four different Shells. They are as follows:

1. ASH Shell
2. Broun shell (BASH Shell)
3. C Shell (CSH Shell)
4. Z Shell (ZSH Shell)

The Shell Programming is also very high interacted towards the Linux Operating Systems and Linux System Calls. Thus it is played a way towards the System interfacing, Networking and also used for the client and server communication.

This also provides the immense supporting interface and interactions over the various protocols in Linux and other common Operating systems. It also supports for the flow of TCP/IP sockets Application Programming Interface (API), and the net BIOS interfacings along interactions over the various common networks, which is mainly used for the communication between the systems and also help in connecting the different systems over the network.

The Shell Programming can be very easily complied in Linux or other UNIX operating systems. The Linux Operating System is fully managed by the Shell Programming only because all the functions of Linux are based on the System Calls interfacing only.

This is the advantage of using the Shell Programming language. The important reason for using the shell programming is that, it has a concept called pointers which is used to link the files and data's that are stored in any part of the memory.

In this project, the shell programming is effectively used for the concept of pointers for combining the packets before transferring and after receiving in the network transferring over one machine to another machine never mind the operating systems present over that system.

Hence the Shell Programming can be used in this project for interfacing of many systems and networks machines using the IP address or the MAC address of the particular systems present over the Intranet.

Thus totally Linux provides itself as a good environment for developing and deploying the projects in shell programming for the networking and interfacings.

3.2.4 'C' LANGUAGE

Another language used in this project is 'C'. The 'C' programming language is one of the oldest languages in the world for easy interactions of the networking and systems call. The 'C' programming language is used for the effective client and server communication and transferring of data from the one computer to another computer in the Network.

The 'C' language provides the TCP/IP sockets Application Programming Interface (API) along with TCP/IP Protocols, which is mainly used for the communication between the systems that are connected in the network.

Another important reason for using the language 'C' is, it has a concept called pointers, which are used to link the files, and data's that are stored in any

part of the memory. In our project we effectively used the concept of pointers for combining the packets before transferring and after receiving. This pointer is also used over the software for the maintenance over the various log creations.

The Pointers concept is provided in the language C++ also, but the C++ is not so efficient as that of the Programming in C. In the case of networking and implementation over the system transferring calls "C" is most effectively used. Linux supports the compilation of the C as well as C++ efficiently.

The Structures and the unions concepts is provided in the language C++ also, but the C++ is not so efficient as that of the Programming in C. In the case of class and the objects in the C Programming in the form of structures and the unions makes the C programming so efficient than C++ programming languages. Being king of the Programming Languages – C has many implementations over the system transferring calls "C" is most effectively used along with the Linux System programming processing calls.

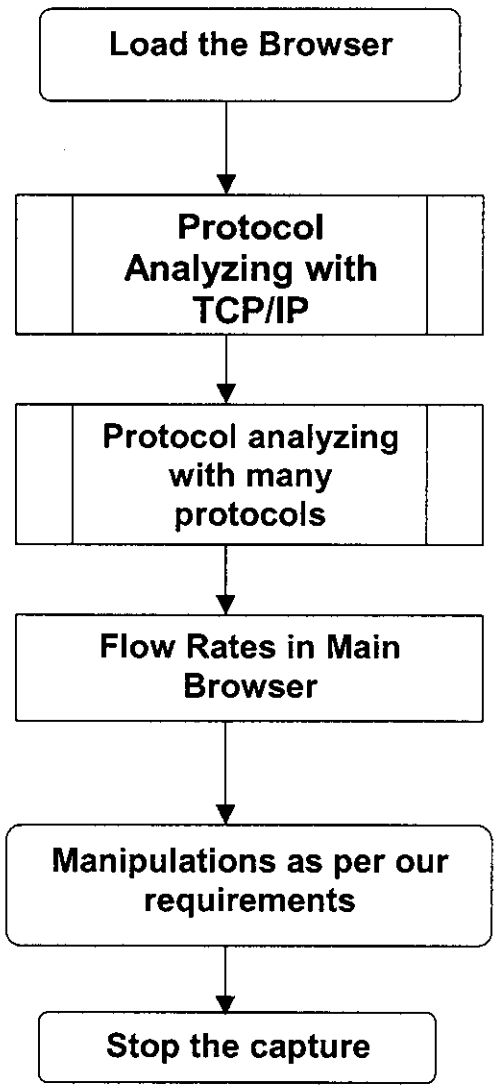
CHAPTER 4

SYSTEM DESIGN

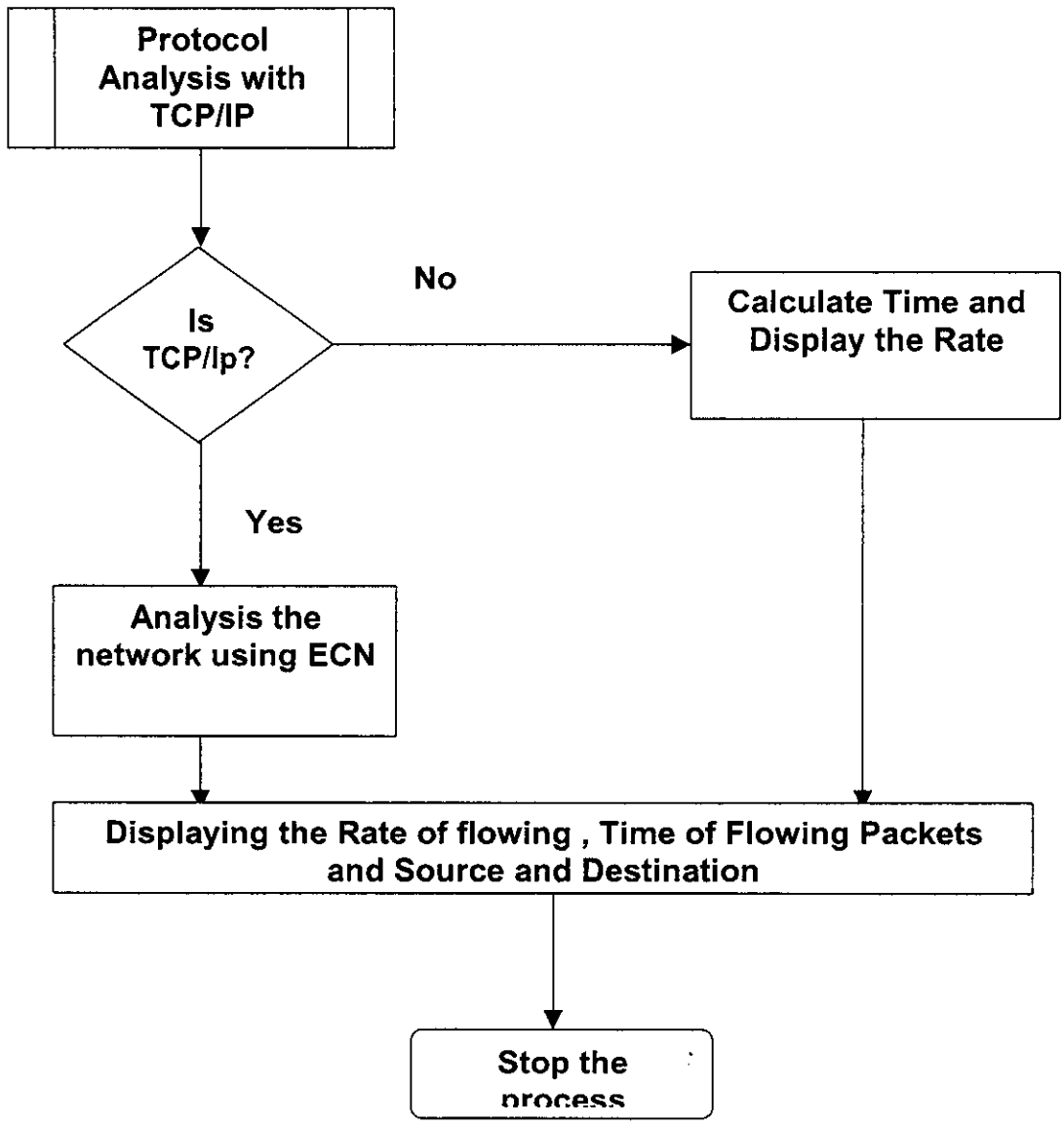
The System Design is also considered important for the software development. The followings are the logical design and physical design of this Project:

4. 1. Logical Design

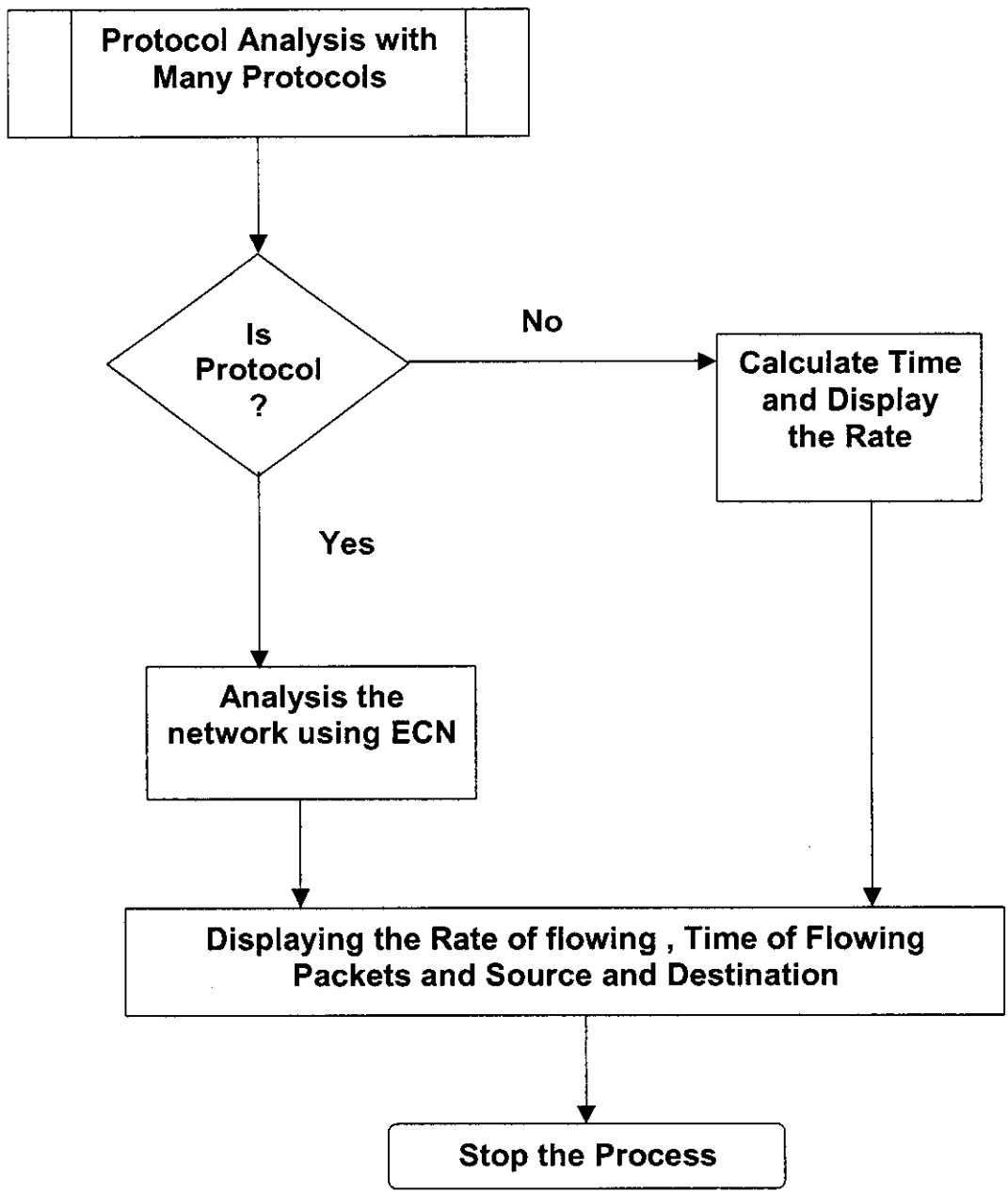
The following figure 4.1 shows the design diagram for protocol analyzing



(Fig 4.1) Flowchart *for Protocol Analyzing*



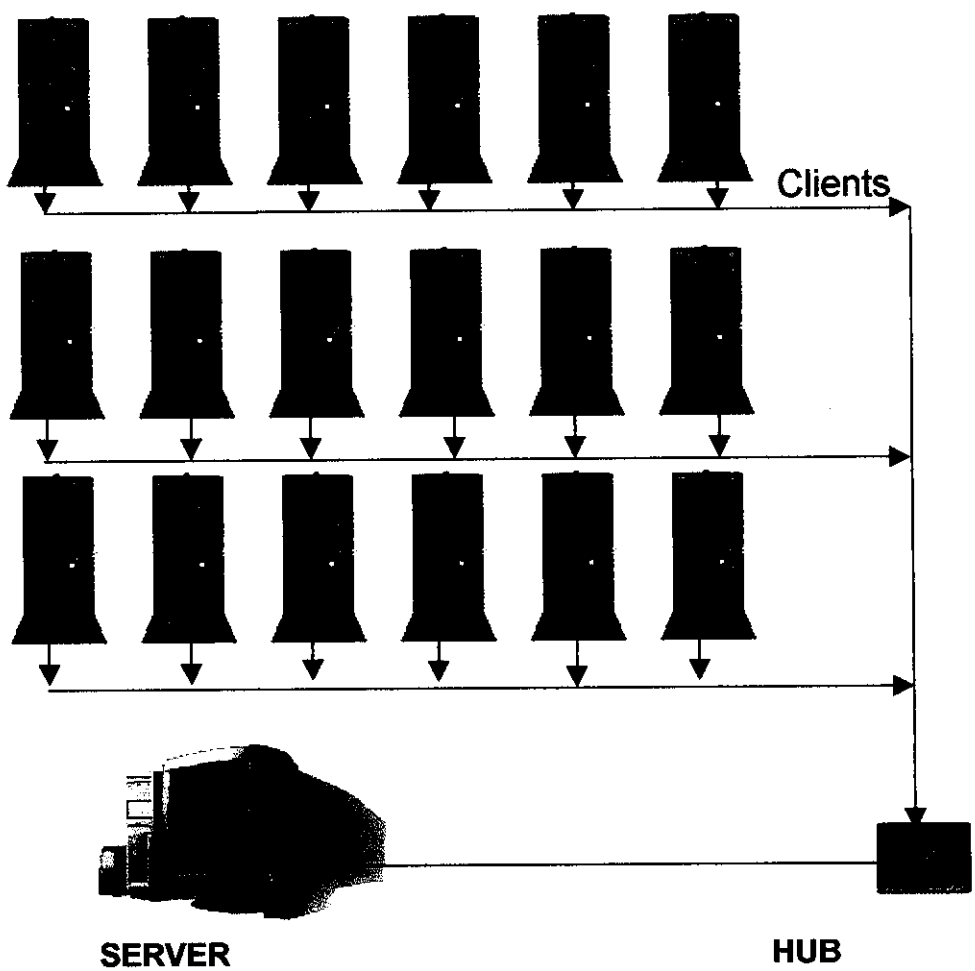
(Fig 4.2) Flowchart *for Protocol Analyzing With TCP*



(Fig 4.3) Flow chart *for Protocol Analyzing With Many Protocols*

4. 2. Physical Design

The figure 4.4 shows the Physical Network Setup for this proposed project:



(Fig 4.4) Network Structure Design

The above diagram gives the clear setup for the entire representation of the proposed project. Al the systems are arranged in the manner of the linear network structure. 15 nodes of the clients and a Server with the Hub are also arranged for the effective networking and flow rate of each protocol in the Network.

CHAPTER 5

SYSTEM IMPLEMENTATION AND TESTING

The System Implementation is also important part of the each project. Our Project is implemented with the following Protocol analysis:

5.1. The TCP/IP Protocol Based Analysis

The TCP/IP is most commonly associated with the Unix operating system. While developed separately, they have been historically tied, as mentioned above, since 4.2BSD Unix started bundling TCP/IP protocols with the operating system. Nevertheless, TCP/IP protocols are available for all widely-used operating systems today and native TCP/IP support is provided in OS/2, OS/400, and Windows 9x/NT/2000, as well as most Unix variants.

The TCP/IP protocol architecture; this diagram is by no means exhaustive, but shows the major protocol and application components common to most commercial TCP/IP software packages and their relationship.

The Network Interface Layer

The TCP/IP protocols have been designed to operate over nearly any underlying local or wide area network technology. Although certain accommodations may need to be made, IP messages can be transported over all of the technologies shown in the figure, as well as numerous others.

It is beyond the scope of this paper to describe most of these underlying protocols and technologies.

7	TCP	echo	80	TCP	http
9	TCP	discard	110	TCP	pop3
13	TCP	daytime	111	TCP	sunrpc
19	TCP	chargen	119	TCP	nntp
20	TCP	ftp-control	123	UDP	ntp
21	TCP	ftp-data	137	UDP	NetBIOS-ns
23	TCP	telnet	138	UDP	NetBIOS-dgm
25	TCP	smtp	139	TCP	NetBIOS-ssn
37	UDP	time	143	TCP	imap
43	TCP	whois	161	UDP	snmp
53	TCP/UDP	dns	162	UDP	snmp-trap
67	UDP	bootps	179	TCP	bgp
68	UDP	bootpc	443	TCP	https (http/ssl)
69	UDP	tftp	520	UDP	rip
70	TCP	gopher	1080	TCP	socks

(Table 5.1) IP Message Transportation

The Transport Layer Protocols

The TCP/IP protocol suite comprises two protocols that correspond roughly to the OSI Transport and Session Layers; these protocols are called the Transmission Control Protocol and the User Datagram Protocol (UDP). One can argue that it is a misnomer to refer to "TCP/IP applications," as most such applications actually run over TCP or UDP. TCP includes rules for formatting messages, establishing and terminating virtual circuits, sequencing, flow control, and error correction. Most of the applications in the TCP/IP suite operate over the *reliable* transport service provided by TCP.

The TCP data unit is called a *segment*; the name is due to the fact that TCP does not recognize messages, per se, but merely sends a block of bytes from the byte stream between sender and receiver.

UDP:

The UDP provides an end-to-end datagram (connectionless) service. Some applications, such as those that involve a simple query and response, are better suited to the datagram service of UDP because there is no time lost to virtual circuit establishment and termination. UDP's primary function is to add a port number to the IP address to provide a socket for the application.

ICMP :

The Internet Control Message Protocol is an adjunct to IP that notifies the sender of IP datagram's about abnormal events. This collateral protocol is particularly important in the connectionless environment of IP. ICMP is not a classic host-to-host protocol like TCP or UDP, but is host-to-host in the sense that

one device (e.g., a router or computer) is sending a message to another device (e.g., another router or computer).

TCP Logical Connections and ICMP:

It is imperative to understand how a TCP connection is established to get a good feel for how TCP operates. TCP connections have three main parts: connection establishment, data exchange, and connection termination. The example below shows a POP3 server (listening on TCP port 110) being contacted by a client (using TCP port 1967). The *connection establishment* phase comprises a three-way handshake during which time the client and server exchange their initial sequence number (ISN) and acknowledge the other host's ISN.

In this example, the client starts by sending the server a TCP segment with the syn-bit set and a Sequence Number of 800. The syn-bit tells the receiver (i.e., the server) that the sender (i.e., the client) is in "ISN initialization" mode and that the ISN hasn't yet been confirmed.The segment's Acknowledgement Number isn't shown because its value is, at this point, invalid. The server responds with a segment with the syn- and ack-bits set, a Sequence Number of 1567, and an Acknowledgement Number of 801. The syn-bit and ISN of 1567 have the same meaning as above.

The ack-bit indicates the value of the Acknowledgement Number field is valid and the ACK value of 801 is the way in which the server confirms the client's ISN. The final part of the three-way handshake is when the client sends a segment

with just the ack-bit set. Note that the Acknowledgement Number field (1568) is one greater than the server's ISN.

This three-way handshake is sometimes referred to as an exchange of "syn, syn/ack, and ack" segments. It is important for a number of reasons. For individuals looking at packet traces, recognition of the three-way handshake is how to find the start of a connection. For firewalls, proxy severs, intrusion detectors, and other systems, it provides a way of knowing the direction of a TCP connection setup since rules may differ for outbound and inbound connections.

The second part of the TCP connection is *data exchange*. The information here is more or less made up for example purposes only; it shows a POP server sending a banner message to the client system, the user sending the "quit" command, and the server signing off. (Note that the "\n" indicates an "end-of-line" indicator.) These segments show the changing of, and relationship between, the client's and server's sequence and acknowledgement numbers.

The final phase is *connection termination*. Although TCP connections are full-duplex (even if a given application does not allow two-way simultaneous communication). The TCP protocol views the logical connection as a pair of simplex links. Therefore, connection termination requires four segments or, more properly, two pair of segments. In this case, the client sends the server a segment with the fin- and ack-bits set; the server responds with a segment with just the ack-bit set and the Acknowledgment Number is incremented. The server then sends a fin/ack segment to the client.

The paragraphs above describe a normal scenario setting up a TCP connection between a client and server. Two UDP hosts communicate in a similar fashion; one host sends a UDP datagram to the other which is presumably listening on the port indicated in the datagram.

The TCP/IP Application Layer

The TCP/IP Application Layer protocols support the applications and utilities that are the Internet. This section will list a number of these applications and show a sample packet decodes of all protocol layers.

The Following is the example for the Family of the protocol interface with the network. The Case study is considered to be the Apple Computer Interactions Protocols. The Apple Talk Protocol is very much interacted towards other networks of Systems. They play important role in the Interactions of the entire network with any numbers of the protocols or systems in the Intranet.

Apple Talk Protocol Suite:

Apple Computer developed the AppleTalk protocol suite to implement file transfer, printer sharing, and mail service among Apple systems using the Local Talk interface built into Apple hardware. AppleTalk ports to other network media such as Ethernet by the use of LocalTalk to Ethernet bridges or by Ethernet add-in boards for Apple machines.

The AppleTalk protocol suite includes the following protocols:

1. AARP - AppleTalk Address Resolution Protocol.
2. DDP - Datagram Delivery Protocol.
3. RTMP - Routing Table Maintenance Protocol.
4. AEP - AppleTalk Echo Protocol.
5. ATP - AppleTalk Transaction Protocol.
6. NBP - Name-Binding Protocol.
7. ZIP - Zone Information Protocol.
8. ASP - AppleTalk Session Protocol.
9. PAP - Printer Access Protocol.
10. ADSP - AppleTalk Data Stream Protocol.
11. AFP - AppleTalk Filing Protocol.

5. 2. *Protocols Used in this Project*

The following are different protocols used in this proposed Project:

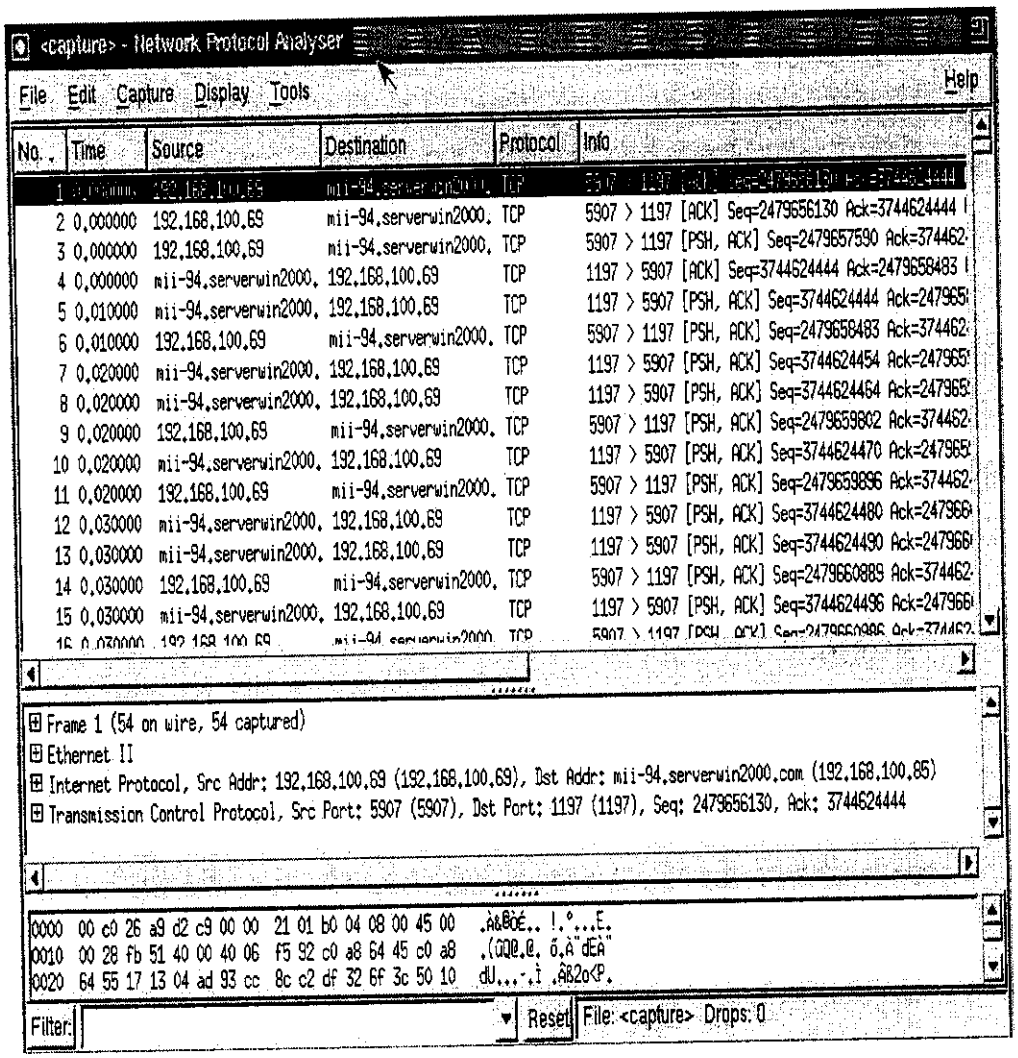
1. ATM (atm)
2. Address Resolution Protocol (arp)
3. Appletalk Address Resolution Protocol (aarp)
4. Authentication Header (ah)
5. Bootstrap Protocol (bootp)
6. Border Gateway Protocol (bgp)
7. Data (data)
8. Datagram Delivery Protocol (ddp)
9. Domain Name Service (dns)
10. Dynamic DNS Tools Protocol (ddtp)
11. Enhanced Interior Gateway Routing Protocol (eigrp)
12. Ethernet (eth)

13. Extended X.25 (modulo 128) (ex25)
14. FTP Data (ftp-data)
15. ame)
16. General Inter-ORB Protocol (giop)
17. Generic Routing Encapsulation (gre)
18. Hypertext Transfer Protocol (http)
19. internet Control Message Protocol (icmp)
20. Internet Control Message Protocol v6 (icmpv6)
21. Internet Group Management Protocol (igmp)
22. Internet Message Access Protocol (imap)
23. Internet Protocol Version 6 (ipv6)
24. Internet Relay Chat (irc)
25. internetwork Packet eXchange (ipx)
26. light Directory Access Protocol (ldap)
27. Microsoft Windows Logon Protocol (netlogon)
28. NetBIOS (NetBIOS)
29. Network File System (nfs)
30. Network Time Protocol (ntp)
31. Open Shortest Path First (ospf)
32. PPP Multilink Protocol (mp)
33. Point-to-Point Protocol (ppp)
34. Post Office Protocol (pop)
35. Protocol Independent Multicast (pim)
36. RIPng (ripng)
37. RX Protocol (rx)
38. Radius Protocol (radius)
39. Real-Time Transport Protocol (rtp)
40. Real-time Transport Control Protocol (rtcp)
41. Remote Procedure Call (rpc)
42. Remote Shell (rsh)

43. login Protocol (rlogin)
44. Routing Information Protocol (rip)
45. Routing Table (rtmp)
46. Simple Mail Transfer Protocol (smtp)
47. Simple Network Management Protocol (snmp)
48. Systems Network Architecture (sna)
49. Time Protocol (time)
50. Token-Ring (tr)
51. Token-Ring Media Access Control (trmac)
52. Transmission Control Protocol (tcp)
53. Transparent Network Substrate Protocol (tns)
54. Trivial File Transfer Protocol (tftp)
55. User Datagram Protocol (udp)
56. Wireless Transaction Protocol (wap-wsp-wtp)
57. X.25 (x25)
58. X11 (x11)

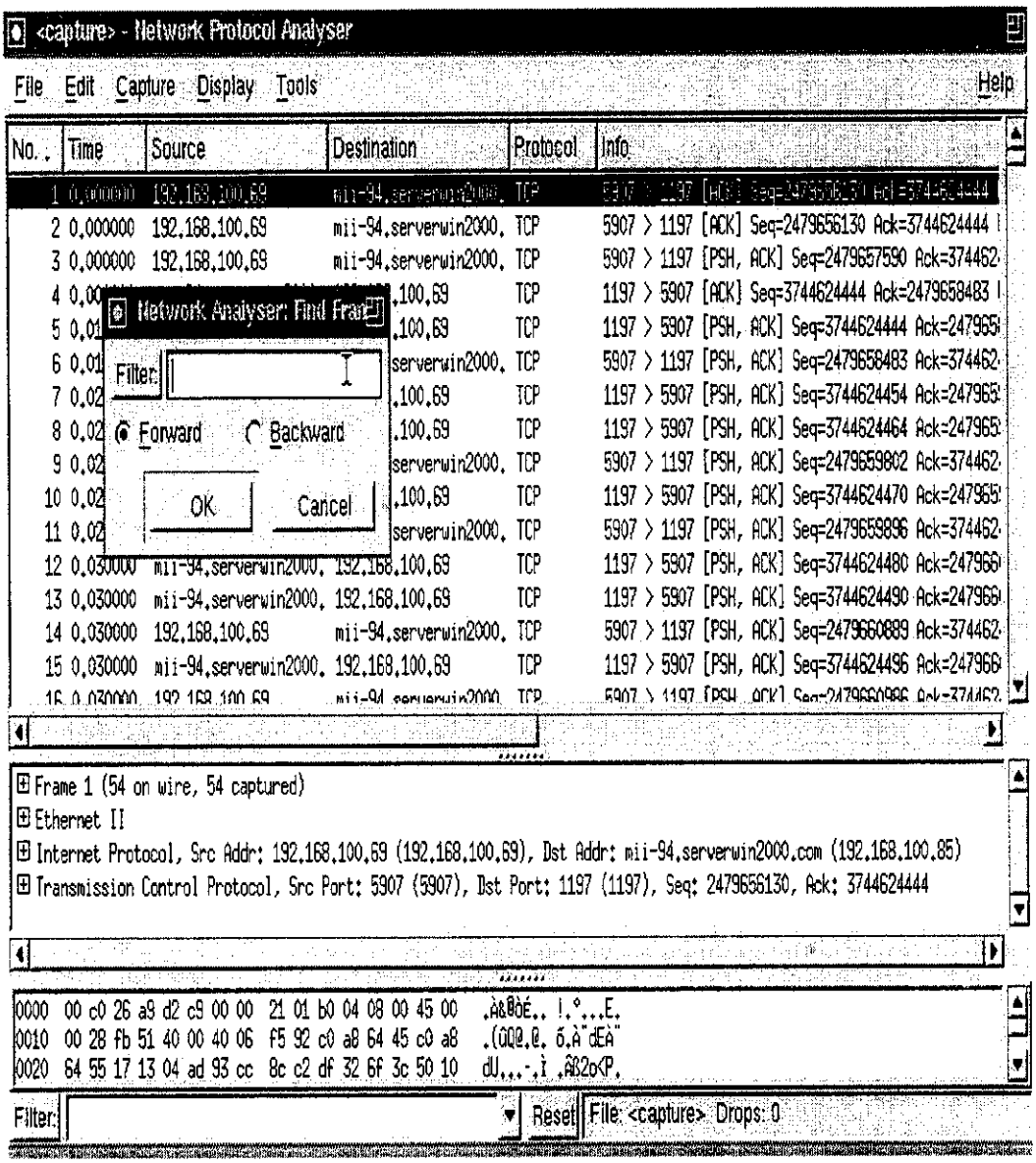
5. 3 Testing

The following figure 5.1 shows the Protocols Analysis Results for the Particular collections of Protocols in the network.



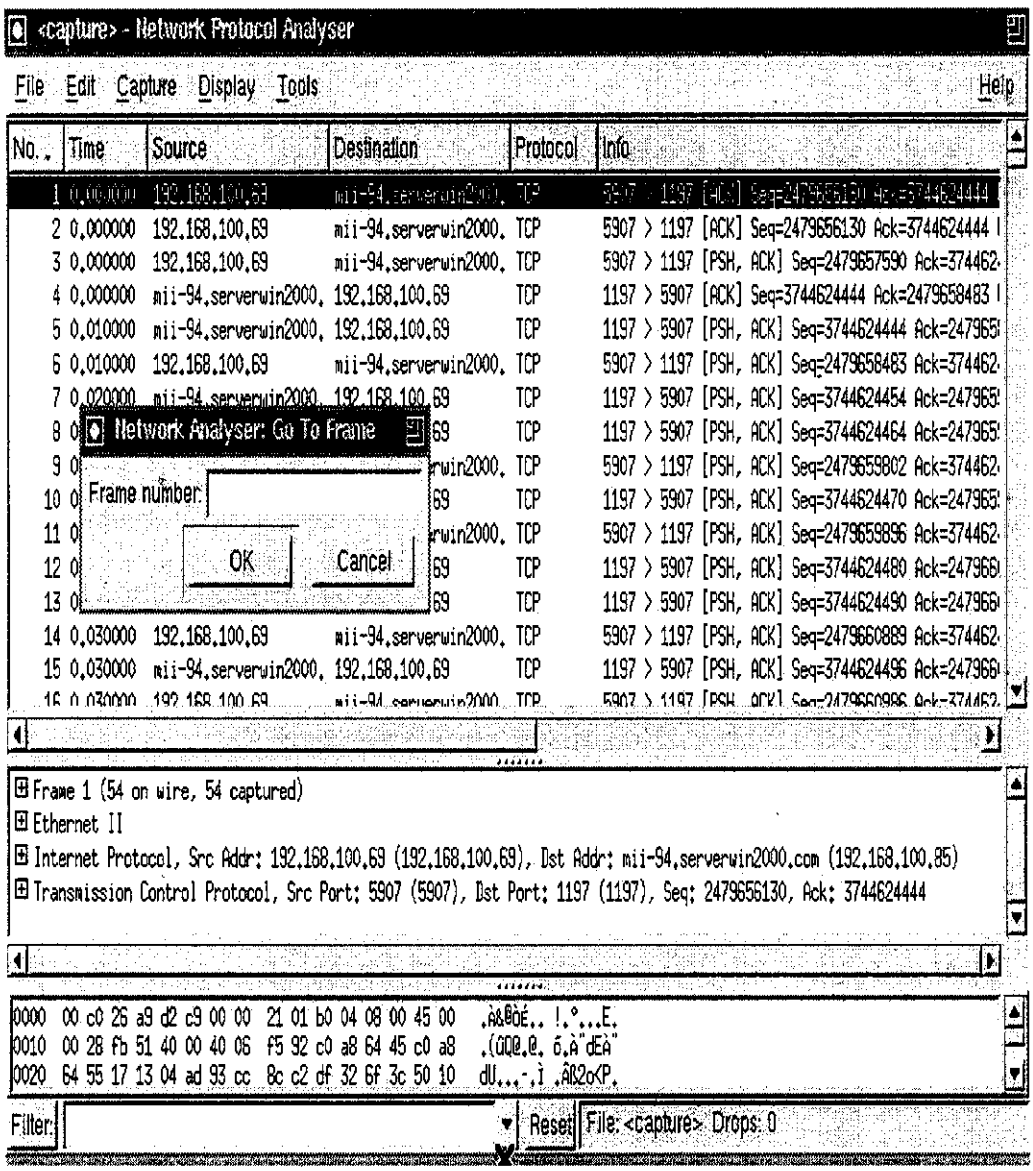
(Fig 5.1) Traffic Analysis Results

The following figure 5.2 shows the Protocols Analysis Results for the Particular collections of Protocols in the network, with option of finding the particular Filter.



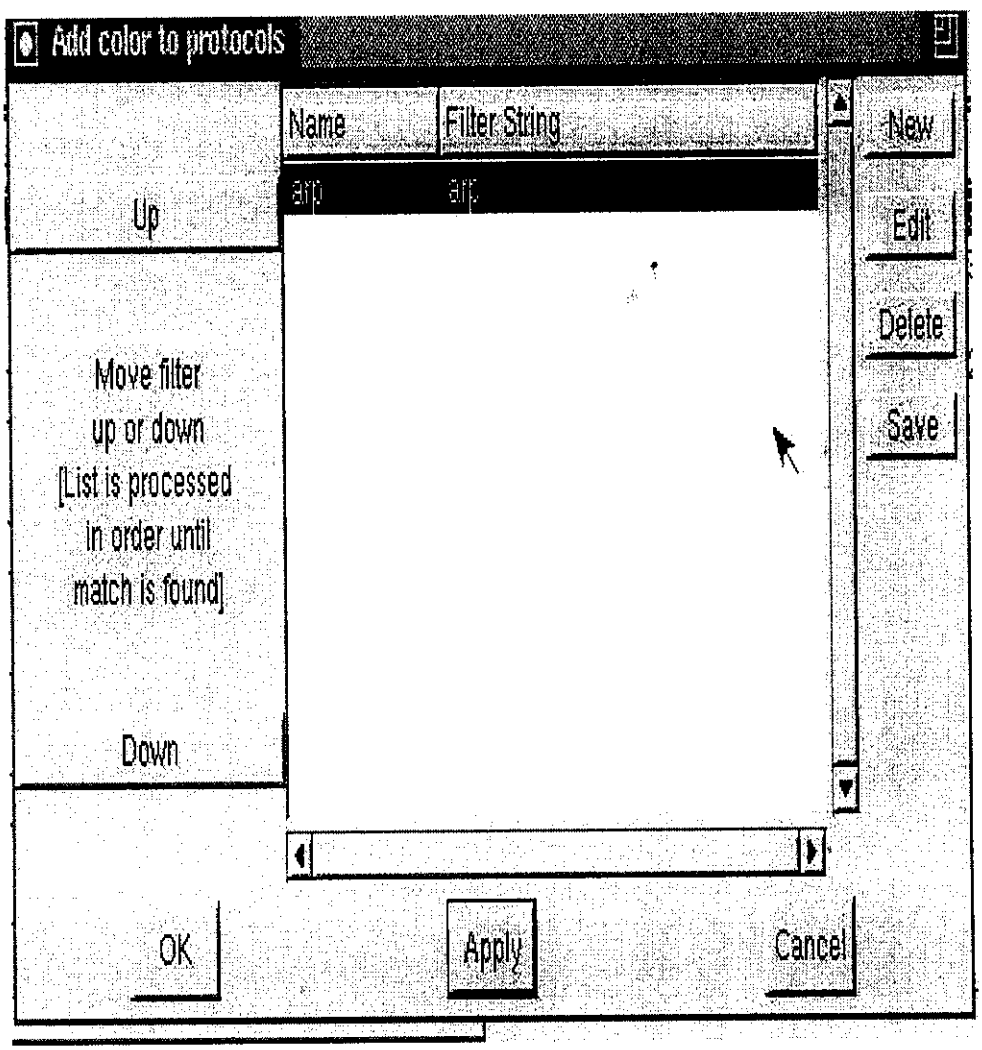
(Fig 5.2) Filters

The following figure 5.3 shows the Protocols Analysis Results for the Particular collections of Protocols in the network, with option of finding the particular Frame Number.



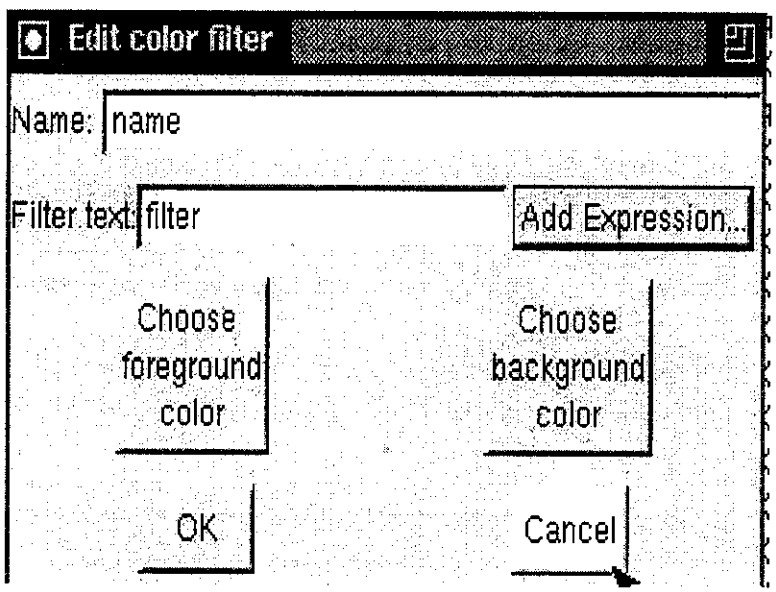
(Fig 5.3)Finding the frames

The following figure 5.4 shows the Protocols Analysis Results for the Particular collections of Protocols in the network, with option of color appearance of the particular protocol.



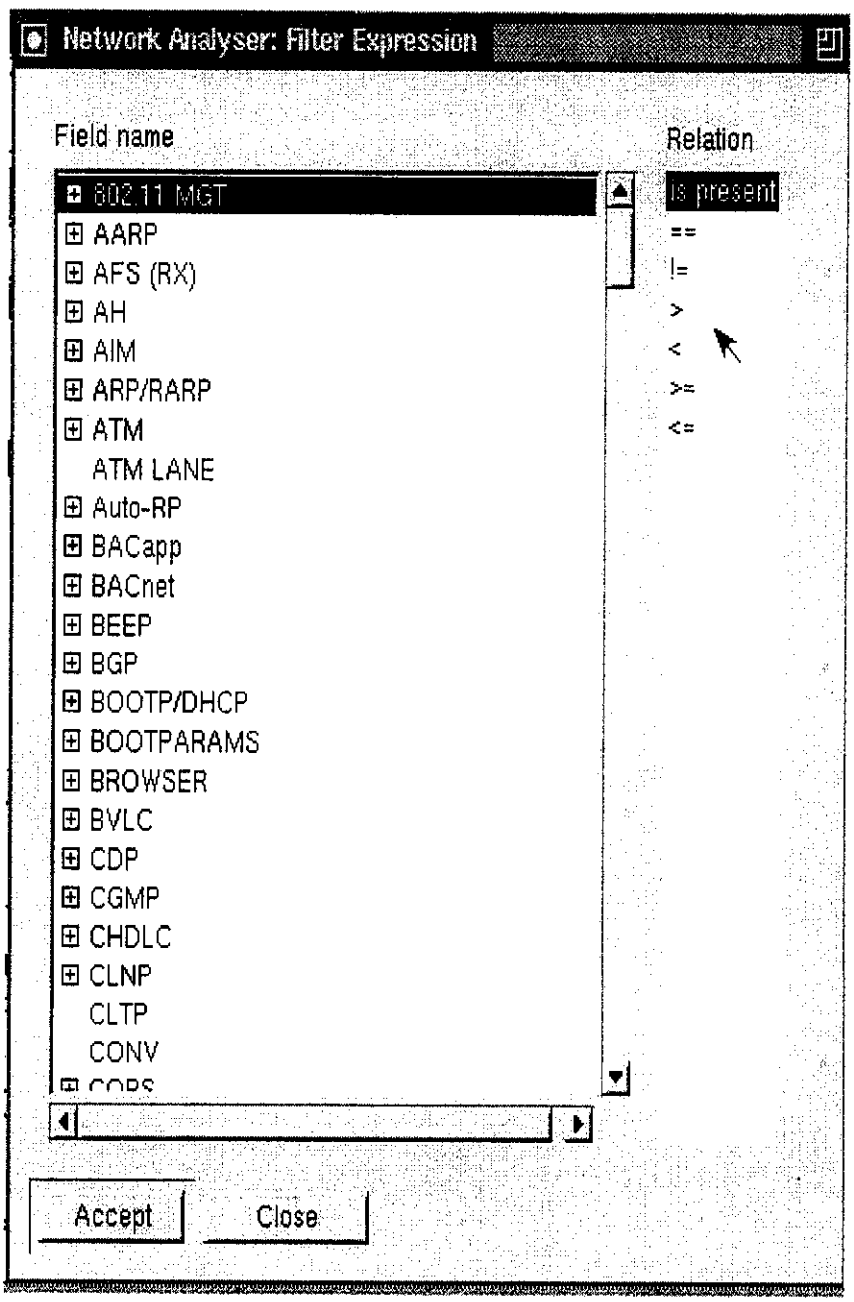
(Fig 5.4)Coloring Options

The following figure 5.5 shows the Protocols Analysis Results for the Particular collections of Protocols in the network, with option of color editing appearance of the particular protocol.



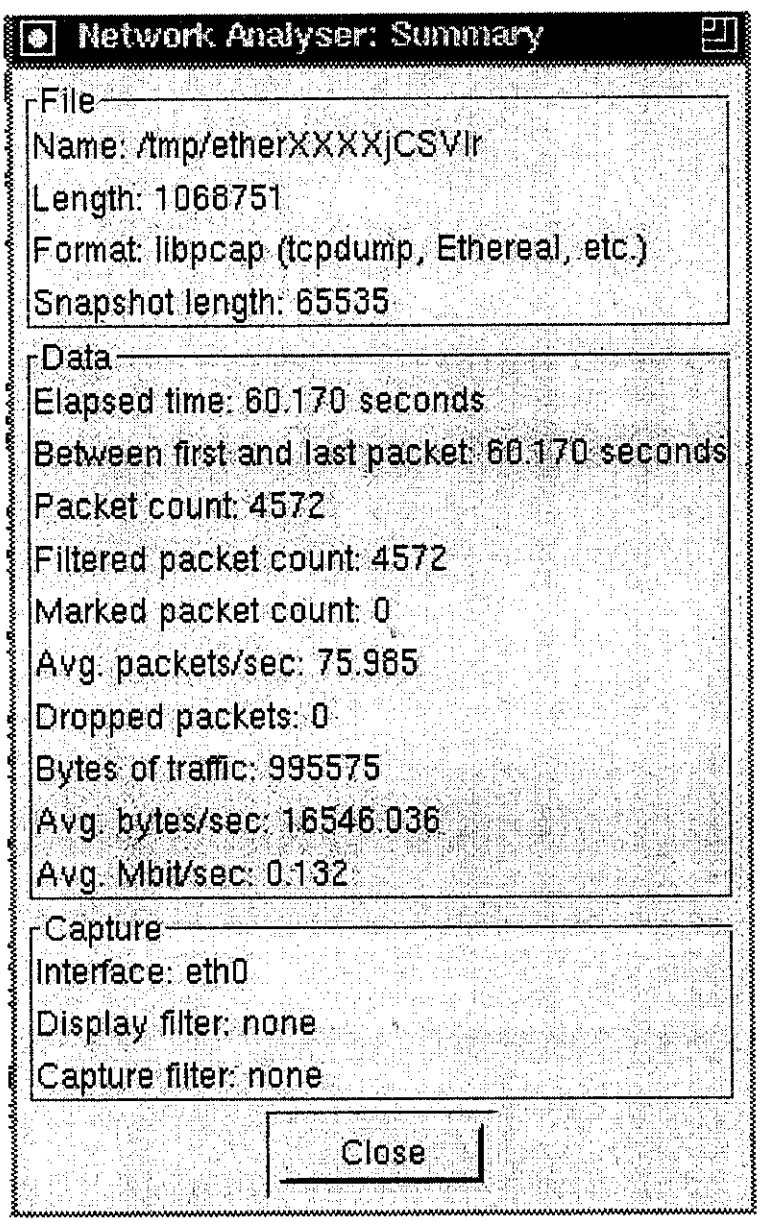
(Fig 5.5)Color Editor

The following figure 5.6 shows the Protocols Analysis Results for the Particular collections of Protocols in the network, with option of Filter Expression of the particular protocol or the group of protocols.



(Fig 5.6) Filter Expressions

The following figure 5.7 shows the Traffic Analysis Results Summary and the entire details of the Particular collections of Protocols or the single protocol in the network.



(Fig5.7)Result Summary

CHAPTER 6

SCREEN DESIGN

GUI of this proposed project contains three main windows, they are:

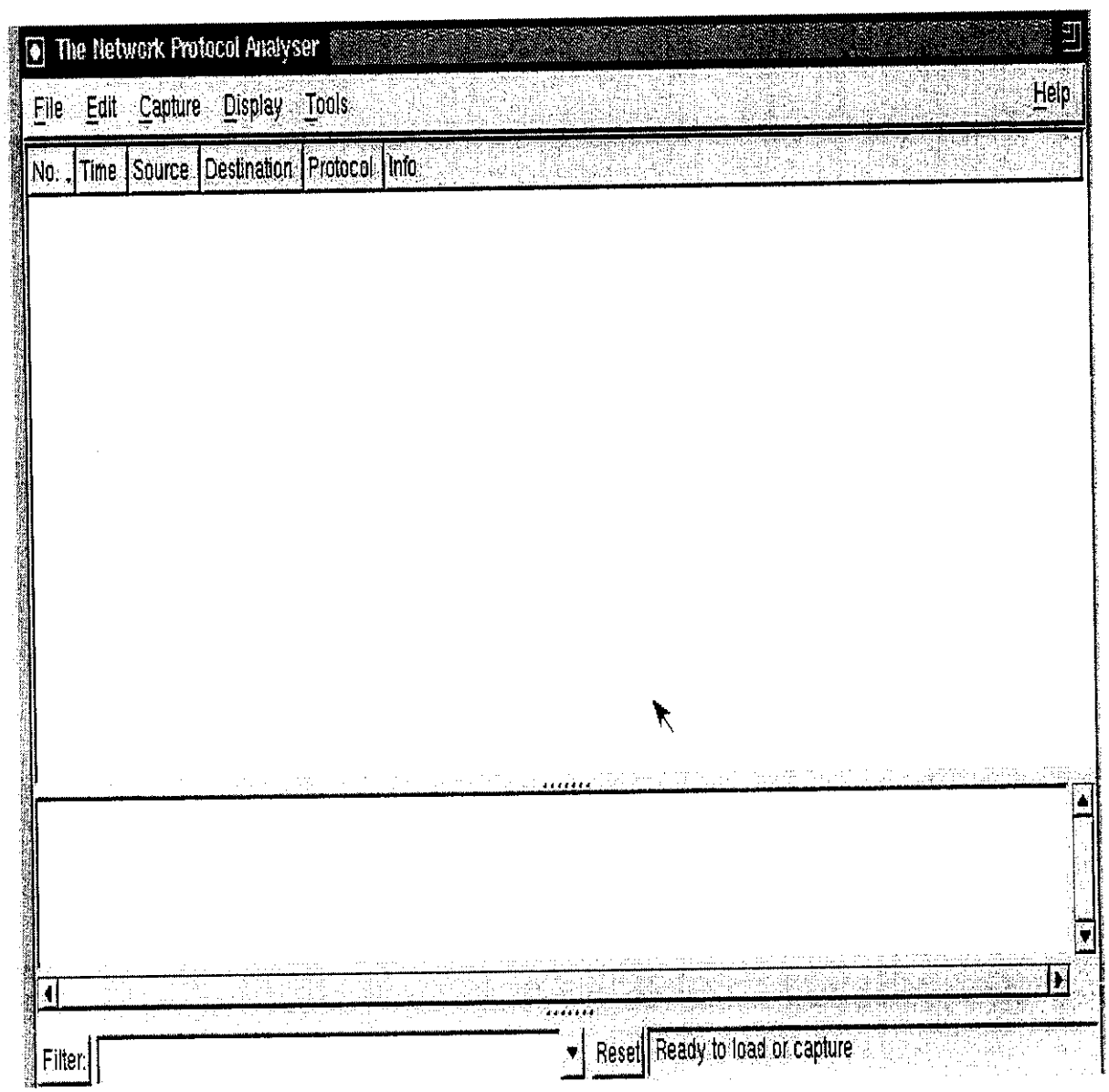
1. Menu Bar
2. Main Browser
3. Filter Browser

The Menu bar contains the following options:

- File
- Edit
- Capture
- Display
- Tools
- Help

The main browser used to list all the details about each packet such as frame. No, Time, Source and Destination Machine IP address, protocols and its information.

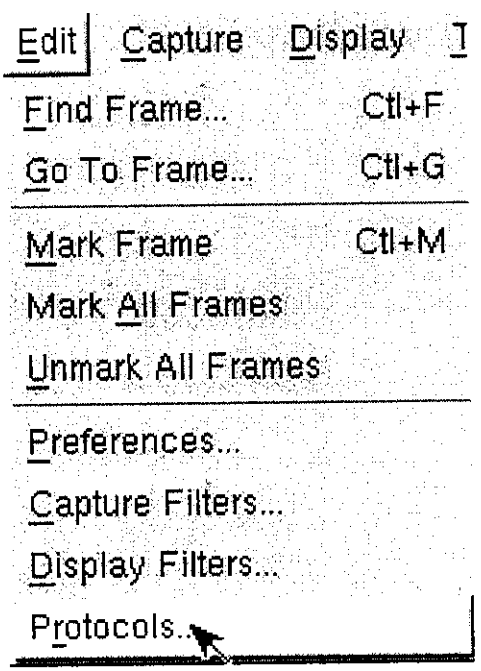
The Filter browser used to filter the particular protocol and rests the existing value.



(Fig 6.1) Main window

The Edit menu of this project contains the following options,

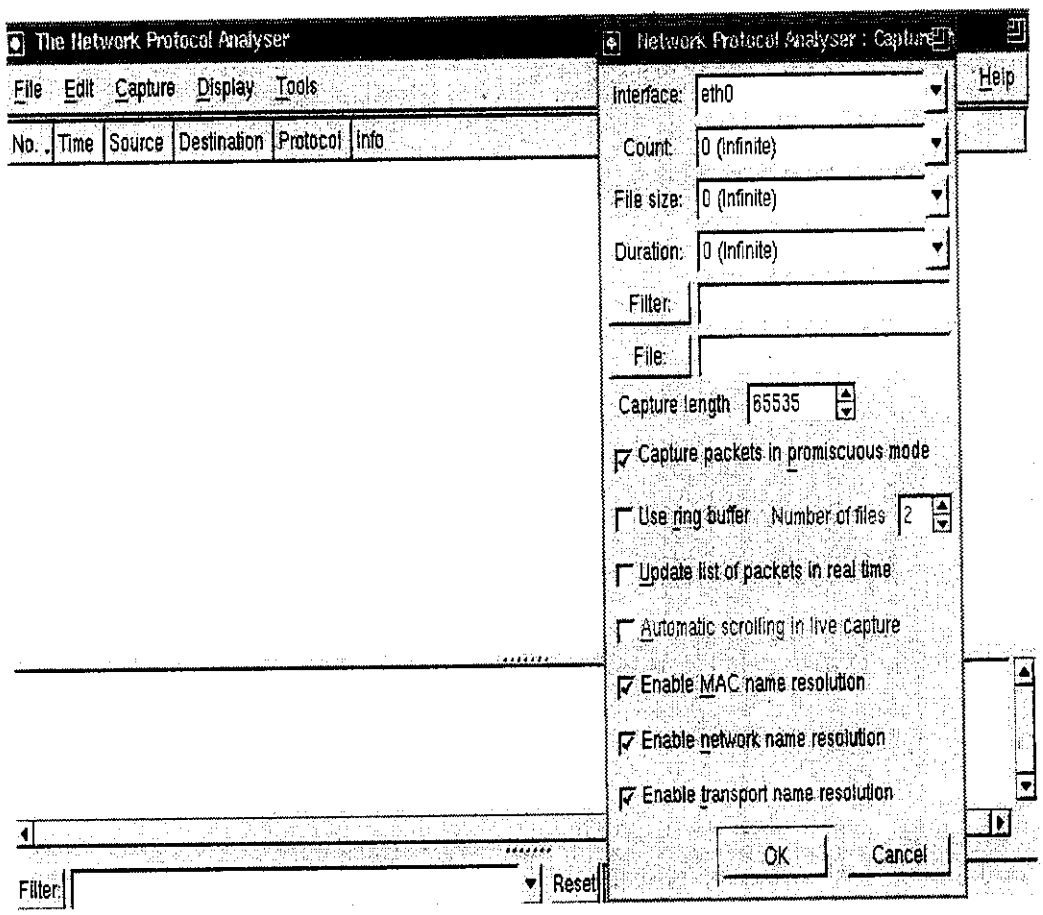
- Find the Frame,
- Go to Frame
- Mark Frame
- Mark All Frame
- Unmark Frame
- Preference
- Capture Filters
- Display Filters
- Protocols



(Fig 6.2)Edit Options

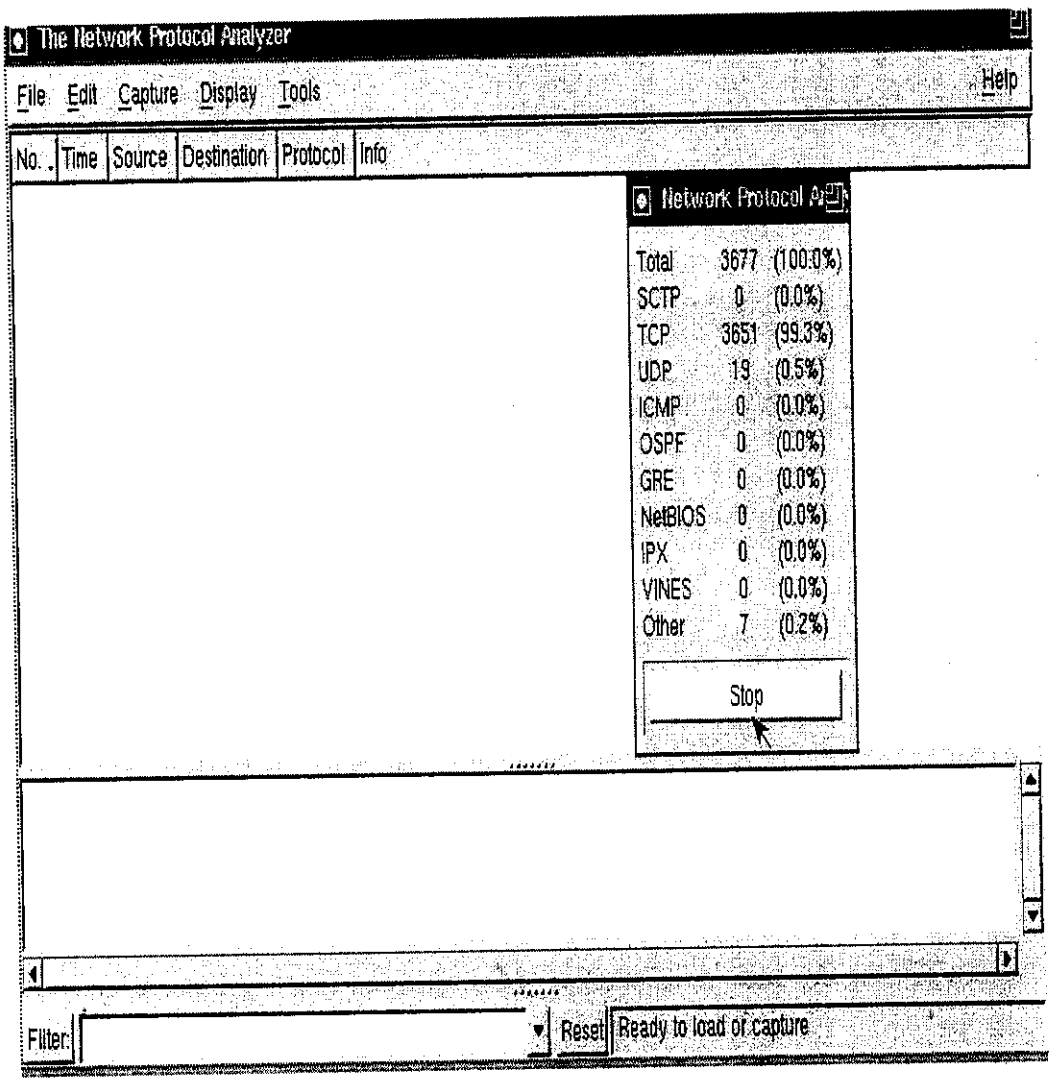
The following child window shows the capture window for the capture Protocols in the network with the choosing options like

- Interface ,
- Counts,
- File Size ,
- Duration of capture



(Fig 6.3) Capture Window

The following figure 6.4 shows the Protocols statistics and the results analysis of the protocols in the network.



(Fig 6.4) Packet Summary

CHAPTER 7

DISCUSSION

7.1. Future Scope of the Project

This project is mainly concerned with the network and flow rate transferring enhancements in (LAN/MAN) Network with user-friendly GUI environment. So that the User can use this project for to know the flow rate of the packet and to enhance the performance of the network at the time.

In future, we can port this software to implement in the satellite network such that efficient flow rate can be obtained with low bandwidth.

The following features can be added in this project in future. To tell who or what is using up valuable network resources, when, where and why. Also provisions can be provided for finding errors in the network

CHAPTER 8

CONCLUSION

The Network Traffic analyzer is an invaluable piece of software that takes the strain out of managing a network. In this System, the user is provided with a easy to use graphical user interface. We can 'see' where our packets are going, and where they're not.

This project captures conversations between two or more systems or devices. The Traffic analyzer not only captures the traffic, it also decodes (interprets) the traffic. Also provide statistics and trend information on the captured traffic.

The Traffic analyzer can show runaway traffic (broadcast or multicast storms) and its origin, system errors and retries, and whether a station is sending, trying to send, or only seeming to communicate. We will get information that is otherwise unavailable, which results in more efficient troubleshooting and better LAN/WAN health.

REFERENCES

1.*Denim Mac man (1999)* "Linux Networking" The Red hat Official Press

2.*Kallis Thomson (1993)* "Shell Programming in Linux " Tata McGraw Hill

3.*Mark Tennman (1991)* "The Protocol Programming" PUE Publications

4.*Steven Richardson (1996)* "TCP/IP Illustrate: Part 1" Tata McGraw Hill

5.*Thomas Kurt's (1995)* "Red hat Linux Networking Administration Tool" The Red hat Official Press

6.*Yashwant Kanetkar (1990)*"The Mastering C Programming Language " BPB Publications