P-2718

NETWORK INFORMATION DETECTOR AND CONNECTION CONTROLLER

By

LENIN.J.K

Register Number: **71206621029**

Of

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE-6

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

of

MASTER OF COMPUTER APPLICATIONS

ANNA UNIVERSITY

CHENNAI 600 025

July 2009

BONAFIDE CERTIFICATE

Certified that this project report titled **Network Information Detector And Connection Controller** is the bonafide work of **Lenin.J.K.** (Register Number: **71206621029**) 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.

30/6/09

Project Guide

Head of the Department

Submitted for the University Examination held on 6.7.2009

6/7/09

Internal Examiner

6/7

External Examiner

i-Serve Technologies
24/16, First Main Road United India Colony
Kodambakkam, Chennai - 600 024
Ph: 044 4204 7983
www.iservetechnologies.com
email: info@iservetechnologies.com

Date: 09.06.09
IST/CH/PL-1145

TO WHOM SO EVER IT MAY CONCERN

This is to certify that **Mr.J.K.Lenin, Reg.No: 06MCA29,** pursuing **M.C.A** in **KUMARAGURU COLLEGE OF TECHNOLOGY** have completed his academic project **"Network Information detector and connection Controller"** in our company successfully from **December 2008 to May 2009.**

Thanks and Regards,

Ms.V.Vengatesh
Project coordinator

i-Serve Technologies Chennai 600 024

Mr.Senguttuvan
Hr Admin

ABSTRACT

Network Information Detector and Connection Controller is an implementation of a Packet Sniffer that captures packet in a LAN environment using Microsoft .NET environment and written in C#. The goal is to build a network utility tool that can be an assistant to programmers, network managers and private users. The Sniffer can be useful for monitoring traffic, debugging, fault analysis, network protocol analysis, network intrusion detection, traffic measuring etc.

In order to reach our goals, project supplies some useful features that can be easily used. Features can be summarized as follows:

- Organize captured packets in a connection-oriented view.
- Smart real-time analyzer enables on-the-fly content viewing while capturing and analyzing.
- Parse and decode a variety of network protocol.
- Protocol definition tool to extend protocols that are decodable.
- Powerful filter provides a flexible mechanism to capture specific packets.
- Port Scanner utility.
- Finds process that uses each connection.
- Syntax highlighting for application data.

ACKNOWLEDGEMENT

First and foremost I thank God for his good will and blessings showered on me throughout the project.

I wish to express my deep unfathomable feeling of gratitude and indebtedness to **Prof. R. Annamalai, Vice Principal – Kumaraguru College of Technology, Coimbatore** for the successful completion of the project work.

I am very gladly taking this opportunity to express a special woFrd of thanks to **Dr. M. Gururajan, Head of the Department, Department of MCA, Kumaraguru College of Technology, Coimbatore** for encouraging me to do this work.

I am very much indebted to Project Coordinator **Dr. A. Muthukumar PhD., Professor, MCA, Kumaraguru College of Technology, Coimbatore** for his complete assistance, guidance and support given to me throughout my project.

I would express heartfelt thanks to our internal guide **Mrs. V. Geetha, Assistant Professor, MCA, Kumaraguru College of Technology** as without her best guidance it would not have been possible for me to successfully complete this project.

It's always a pleasure and privileges to be associated with a prestigious outstanding esteemed organization **"Cyber Technologies (*i* serve)", Chennai.**

My hearty thanks to my **Guide Mr.V.Vengatesh** of **Cyber Technologies**, for their valuable guidance throughout the project. Also, I am grateful to my parents and friends who were the real source of my project.

TABLE OF CONTENTS

LIST OF ABBREVATIONS

MENeT	Monitor for Ethernet Network traffic
TCP/IP	Transmission Control Protocol/Internet protocol
LAN	Local Area Network
UDP	User Datagram Protocol
SOAP	Simple Object Access Protocol
DLL	Dynamic Link Library
CLR	Common Language Runtime

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The project is an implementation of a Packet Sniffer that captures packet in a LAN environment using Microsoft .NET environment and written in C#. The goal is to build a network utility tool that can be an assistant to programmers, network managers and private users. The Sniffer can be useful for monitoring traffic, debugging, fault analysis, network protocol analysis, network intrusion detection, traffic measuring etc.

In order to reach our goals, project supplies some useful features that can be easily used. Features can be summarized like this:

- Organize captured packets in a connection-oriented view.
- Smart real-time analyzer enables on-the-fly content viewing while capturing and analyzing.
- Parse and decode a variety of network protocol.
- Protocol definition tool to extend protocols that are decodable.
- Powerful filter provides a flexible mechanism to capture specific packets.
- Port Scanner utility.
- Finds process that uses each connection.
- Syntax highlighting for application data.

Using a capture library, build a GUI that serve user an easy interface. To build capture library we use .NET network classes that provide interface to native Winsock API and asynchronous sockets that brings thread based solution to socket programming.

1.2 ORGANIZATION PROFILE

Cyber Technologies (*i* serve) India Private Limited was established in 2000 with the goal to provide customers with IT and Software Development to enhance their productivity and profitability. Today, one of India's leading companies with large, medium and small enterprise with a wide range of Technology providers relying on us as a one stop source for all of their IT and related high-tech business information needs.

Cyber Technologies (*i* serve) is a global Software Solutions Developer with a mission to provide cost effective, innovative and simplified solutions to enterprises. It is a fast growing and end-to-end HR services Provider Company, Combining proven expertise in technology and project management.

With presence in various locations across India, Singapore, US and Ireland enjoy a leading position in recruitment and training. Development centers both in Singapore and India provide the highest quality and significant cost savings, with domestic project management in the Singapore.

Cyber Technologies (*i* serve) allows focusing on core business and providing solutions to enable business to be a success. As part of our continuous efforts to increase the quality of our integrated services to different clients, looking to tie-up with more MNC's, which are looking for software professionals who have a thirst to excel.

SOFTWARE DEVELOPMENT

(i) Custom tailored Solutions
(ii) Software Architecture & Design
(iii) Software Requirement Specification
(iv) Systems Integration and Networking Solutions
(v) Outsourcing software development

There is competition, the market is price sensitive, the customers are demanding receivables are high, cost need to be controlled, people have to be more effective, new product development is critical, maintaining and optimizing the supply chain is critical quality has to be high, we have to be fast... The list is endless.

In today's intensely competitive market environment, companies need scrupulous observance of enterprise wide rules between departments for information and action growth plans entail having proper 'Enterprise-wide Integrated system' in place for Planning monitoring & control .

Well qualified and experienced team of engineers who understand an Enterprise or Customers complex business hitch and their comprehensive solution with latest and cost effective technology tools know today's business needs and how to objectify a complex business pain into a small chunk of pieces and manipulate them into ultimate solutions.

CHAPTER 2

SYSTEM STUDY

2.1 EXISTING SYSTEM

Existing system has a drawback of manual errors in identify ip address and ping etc..This project is to avoid manual errors which are listed about. The main use is to capture all packets automatically. There may be many drawbacks in the existing system. They may be as follows

(i) Less carefulness.
(ii) Manual Checking.
(iii) Continuous listening is not possible.

2.2 PROPOSED SYSTEM

By the process of automation, the errors which are experienced manually are eradicated about 80 to 90 percent. By minimizing the errors the product performance is likely increased by certain values. By automating the existing system, the workload is minimized, the performance level is increasing.

Network Monitoring is essential for companies of any size and branch.Uptime monitoring can ensure that computer systems are running smoothly and that will be notified when outages occur. A network monitoring tool is also important to increase the efficiency of network by understanding bandwidth and resource consumption through usage monitoring and bandwidth management.

2.3 SYSTEM REQUIREMENT SPECIFICATION

The Software requirement specification is a technical specification of requirements for the software product. The goal of software requirements definition is to completely specify the technical requirements for the software products in a concise and unambiguous manner

2.3.1 HARDWARE ENVIRONMENT

- MAIN PROCESSOR : Pentium III
- NETWORK : Windows Xp
- MEMORY RAM : 128MB
- HARD DISK CAPACITY : 20GB
- MONITOR : Samsung
- FLOPPY DISK DRIVE : 1.44MB
- CD ROM DRIVE : 52x
- KEYBOARD : 101 Keys

2.3.2 SOFTWARE ENVIRONMENT

The software support required for deployment

OPERATING SYSTEM : Windows Xp

FRONT END : C#.NET

2.3.3 SOFTWARE OVERVIEW

2.3.3.1 NET FRAMEWORK

The .NET Framework has many things, but it is worthwhile listing its most important aspects. In short, the .NET Framework is a Platform designed from the start for writing Internet-aware and Internet-enabled applications that embrace and adopt open standards such as XML, HTTP, and SOAP.A Platform that provides a number of very rich and powerful application development technologies, such as Windows Forms, used to build classic GUI applications, and of course ASP.NET, used to build web applications. A Platform with an extensive class library that provides extensive support for date access (relational and XML), a director services, message queuing, and much more.

A platform that has a base class library that contains hundreds of classes for performing common tasks such as file manipulation, registry access, security, threading and searching of text using regular expressions. A platform that doesn't forget its origins and has great interoperability support for existing components that or third parties have written using COM or standard DLLs. A Platform with an independent code execution and management environment called the Common Language Runtime(CLR) which ensures code is safe to run, and provides an abstract layer on top of the operating system, meaning that elements of the .NET framework can run on many operating systems and devices Objectives To provide a consistent object-oriented programming environment. To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code. The .NET Framework has two main components

(i) .Net Framework class library

(ii) Common Language Runtime (CLR).

The .Net Framework class library is a comprehensive, object-oriented collection of reusable types that can be used to develop applications ranging from traditional command line or GUI applications to applications based on the latest innovations provided by ASP.NET such as Web Forms and XML Web services.

2.4 LANGUAGE SPECIFICATION

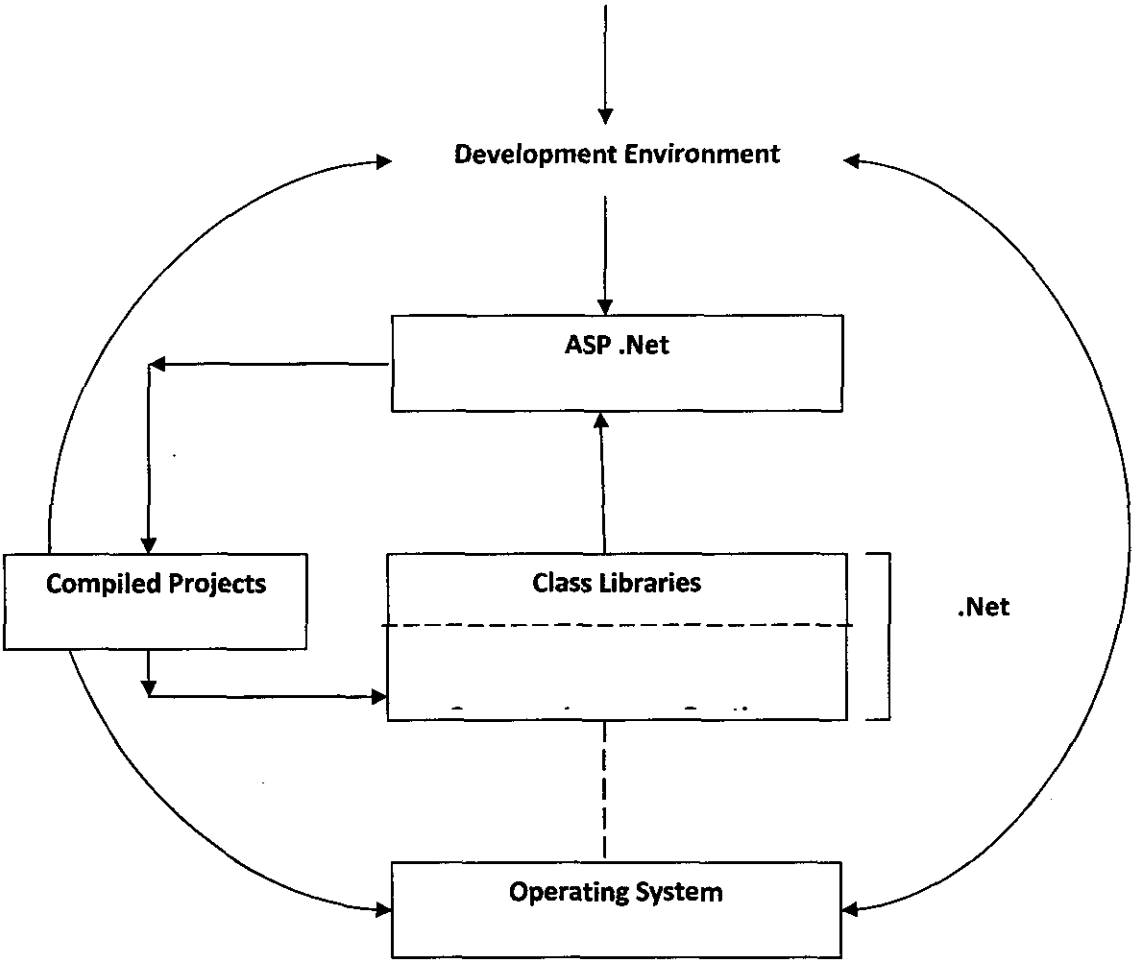

Figure 1 Components of the .Net Frame Work

C#:

Design goals

- C# is intended to be a simple, modern, general-purpose, object-oriented programming language.
- The language, and implementations thereof, should provide support for software engineering principles such as strong type checking, array bounds checking, detection of attempts to use uninitialized variables, and automatic garbage collection. Software robustness, durability, and programmer productivity are important.
- The language is intended for use in developing software components suitable for deployment in distributed environments.
- Source code portability is very important, as is programmer portability, especially for those programmers already familiar with C and C++.
- Support for internationalization is very important.
- C# is intended to be suitable for writing applications for both hosted and embedded systems, ranging from the very large that use sophisticated operating systems, down to the very small having dedicated functions.
- Although C# applications are intended to be economical with regard to memory and processing power requirements, the language was not intended to compete directly on performance and size with C or assembly language.

Features

Some notable C# distinguishing features are:

- There are no global variables or functions. All methods and members must be declared within classes. Static members of public classes can substitute for global variables and functions.
- Local variables cannot shadow variables of the enclosing block, unlike C and C++. Variable shadowing is often considered confusing by C++ texts.

- C# supports a strict Boolean datatype, bool. Statements that take conditions, such as while and if, require an expression of a boolean type. While C++ also has a boolean type, it can be freely converted to and from integers, and expressions such as if (a) require only that a is convertible to bool, allowing a to be an int, or a pointer. C# disallows this "integer meaning true or false" approach on the grounds that forcing programmers to use expressions that return exactly bool can prevent certain types of programming mistakes such as if (a = b) (use of = instead of ==).
- In C#, memory address pointers can only be used within blocks specifically marked as *unsafe*, and programs with unsafe code need appropriate permissions to run. Most object access is done through safe object references, which always either point to a "live" object or have the well-defined null value; it is impossible to obtain a reference to a "dead" object (one which has been garbage collected), or to random block of memory. An unsafe pointer can point to an instance of a value-type, array, string, or a block of memory allocated on a stack. Code that is not marked as unsafe can still store and manipulate pointers through the System.IntPtr type, but it cannot dereference them.
- Managed memory cannot be explicitly freed; instead, it is automatically garbage collected. Garbage collection addresses memory leaks by freeing the programmer of responsibility for releasing memory which is no longer needed. C# also provides direct support for deterministic finalization with the using statement (supporting the Resource Acquisition Is Initialization idiom).
- Multiple inheritance is not supported, although a class can implement any number of interfaces. This was a design decision by the language's lead architect to avoid complication, avoid dependency hell and simplify architectural requirements throughout CLI.
- C# is more typesafe than C++. The only implicit conversions by default are those which are considered safe, such as widening of integers and conversion from a derived type to a base type. This is enforced at compile-time, during JIT, and, in some cases, at runtime. There are no implicit conversions between booleans and integers, nor between enumeration members and integers (except for literal 0, which can be implicitly converted to any enumerated type). Any user-defined conversion must be explicitly marked as explicit or implicit, unlike C++ copy

- Enumeration members are placed in their own scope.
- C# provides properties as syntactic sugar for a common pattern in which a pair of methods, accessor (getter) and mutator (setter) encapsulate operations on a single attribute of a class.
- Full type reflection and discovery is available.
- C# currently (as of 3 June 2008) has 77 reserved words.

2.5 SYSTEM ARCHITECTURE

The diagram explains the architectural design of the system

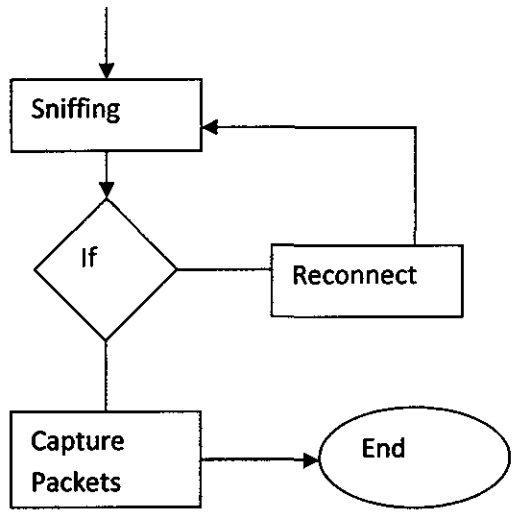

Figure 2: System Architecture

CHAPTER 3

MODULE DESCRIPTION

The following are the modules present in the system

3.1 FILE MODULE

(i) Start Sniffing
(ii) Stop Sniffing
(iii) Check Connection
(iv) Delete
(v) Delete All.

Start Sniffing

After started sniffing, the system have identify all IP addresses, source system Name, Source port, Destination Name, Destination port, Protocol and packets in LAN. Then have to identify each and every system full details like how many packets in that particular system, packet size and etc.

Stop Sniffing

When the sniffing process is going to start all the packets are captured by the network utility tool named sniffer. Then the process can be stopped by through the tool called stop sniffing. This stopped the capture packet process.

Check Connection

After select this option source system will check connection was made properly with destination system. The check connection is the process of monitoring the systems

Delete

Delete is the process of deleting the captured packets from the list. After select this option it will delete the particular packets from selected ports.

Delete All

This option deletes all the ports traced

3.2 EDIT MODULE

The edit module has the following functions:

(i) Sniffing Addresses
(ii) Add Protocol
(iii) Filter Manager
(iv) Mode- Automatic or Manual

Sniffing Addresses

Sniffing address tool is used to view all sniffing addresses.

Add Protocol

The process of adding one or more protocols in source, Computer Network protocol analysis and security auditing. It is capable of intercepting traffic on a network segment capturing passwords, and conducting active eavesdropping against a number of common protocols.

Filter Manager

The process of adding and sending filter from source to destination, Packet filtering/dropping: setting up a filter that searches for a particular string (or hexadecimal sequence) in the TCP or UDP payload and replaces it with a custom string/sequence of

3.3 TOOL MODULE

The tool module has the following functions:

(i) Net Stat
(ii) Ping
(iii) Trace Route
(iv) Check Connection
(v) Settings

Net Stat

The process of Net stat is viewing networking IP addresses with status. Character injection into an established connection: characters can be injected into a server (emulating commands) or to a client (emulating replies) while maintaining a live connection.

Trace Route

The process of Trace Route is tracing which are connected through the LAN. This used to find out the particular route which are the systems are connected as LAN connection.

Check Connection

System that constantly monitors a computer network for slow or failing components and that notifies the network administrator in case of outages via email, pager or other alarms. It is a subset of the functions involved in network management.

CHAPTER 4

IMPLEMENTATION AND TESTING

4.1 IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned into a working system. Implementation phase of the software development is concerned with translating of design specification into source code.

The primary goal of implementation is to write source code and internal documentation so that conformance of the code to its specification can be easily verified, and so that debugging documentation, and unit testing and modification are eased. The objective of the phase was to create input and output files Compatible with user's requirements to handle the operational phases through program construction to test the implemented system and provide complete documentation.

The system has been implemented in the networking application when the systems are connected in the LAN connection.

4.2 TESTING

UNIT TESTING

Unit testing is also known as module testing. It tests the software module by module. This testing focuses on unit-level verification. What constitutes a "unit" has been left imprecise-it may be as little as a single statement or as much as a set of coupled subroutines. The essential characteristic of a unit is that it can meaningfully be treated as a whole.

CHAPTER 5

RESULT AND DISCUSSION

The system is to capture the packets when the systems are connected as a LAN connection. Our goal is to build a network utility tool that can be an assistant to programmers, network managers, and private users. Our Sniffer can be useful for monitoring traffic, debugging, fault analysis, network protocol analysis, network intrusion detection, traffic measuring etc.,

This system has been designed with some new features. The developers can also add some features in this project in future as required. In this system a huge LAN environment has been used for making Network efficiency.

Network Monitoring is essential for companies of any size and branch. With uptime monitoring you can ensure that your computer systems are running smoothly and that will be notified when outages occur. A network monitoring tool is also important to increase the efficiency of network by understanding bandwidth and resource consumption through usage monitoring and bandwidth management..

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

A Packet Capture and injection multiplexer daemon that provides controlled fine grained access to the network device. On systems running packet, client measurement tools are not given direct access to the network device. Instead, they are obliged to request access via packet. By providing administrators control over the pktd mechanism, they can easily and securely enforce their desired policies concerning which clients should be granted which sorts of network access capabilities. Thus, pktd can serve as the sole trusted privileged entity for conducting measurements, eliminating the need for administrators to vet the individual measurement tools.

Organize captured packets in a connection-oriented view. Smart real-time analyzer enables on-the-fly content viewing while capturing and analyzing. Parse and decode an variety of network protocol. Protocol definition tool to extend protocols that are decodable Powerful filter provides a flexible mechanism to capture specific packets. Port Scanner utility finds process that uses each connection.

6.2 FUTURE ENHANCEMENT

The fastest growth of data communication networks over the past decades has resulted in the development of sophisticated tools to diagnose, debug and analyze such networks. have developed a toolkit called MENeT (Monitor for Ethernet Network Traffic). The toolkit is an extension to the earlier developed utility for traffic monitoring. The toolkit is operational on the platform of Windows NT/2000 and is developed by programming in the Microsoft Visual C++ environment. The MENeT captures and dissembles the packets flowing through a system and extracts TCP/IP (transmission control protocol/Internet protocol) packets only. MENeT monitors network traffic by partitioning it into a set of classes. The set of classes are defined as network traffic, broadcast traffic and workstation traffic.

APPENDICES

APPENDIX A – SCREEN SHOTS

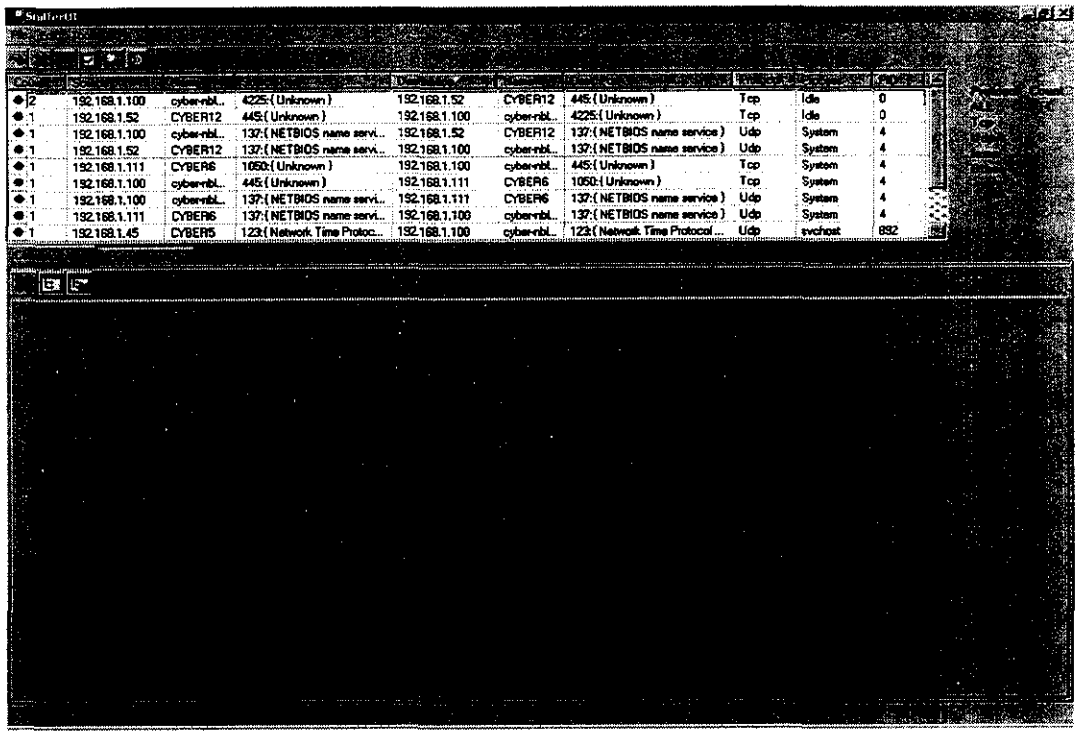

Figure A.1: Sniffer starts as soon as the exe is clicked.

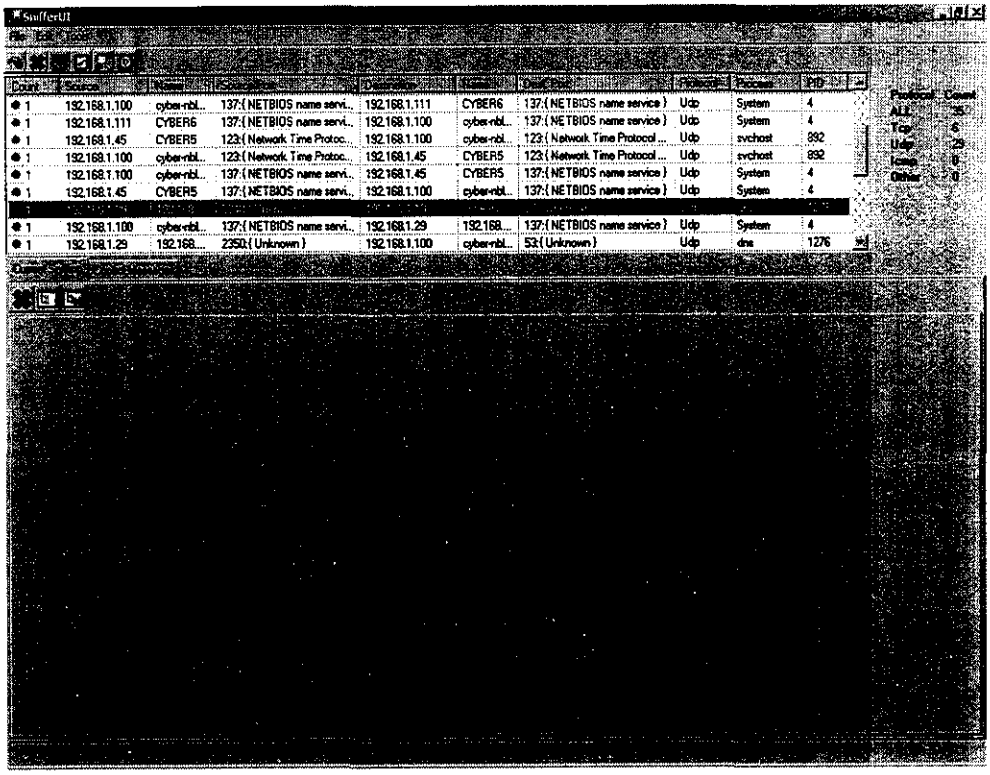

Figure A.2: particular system is selected to view its details.

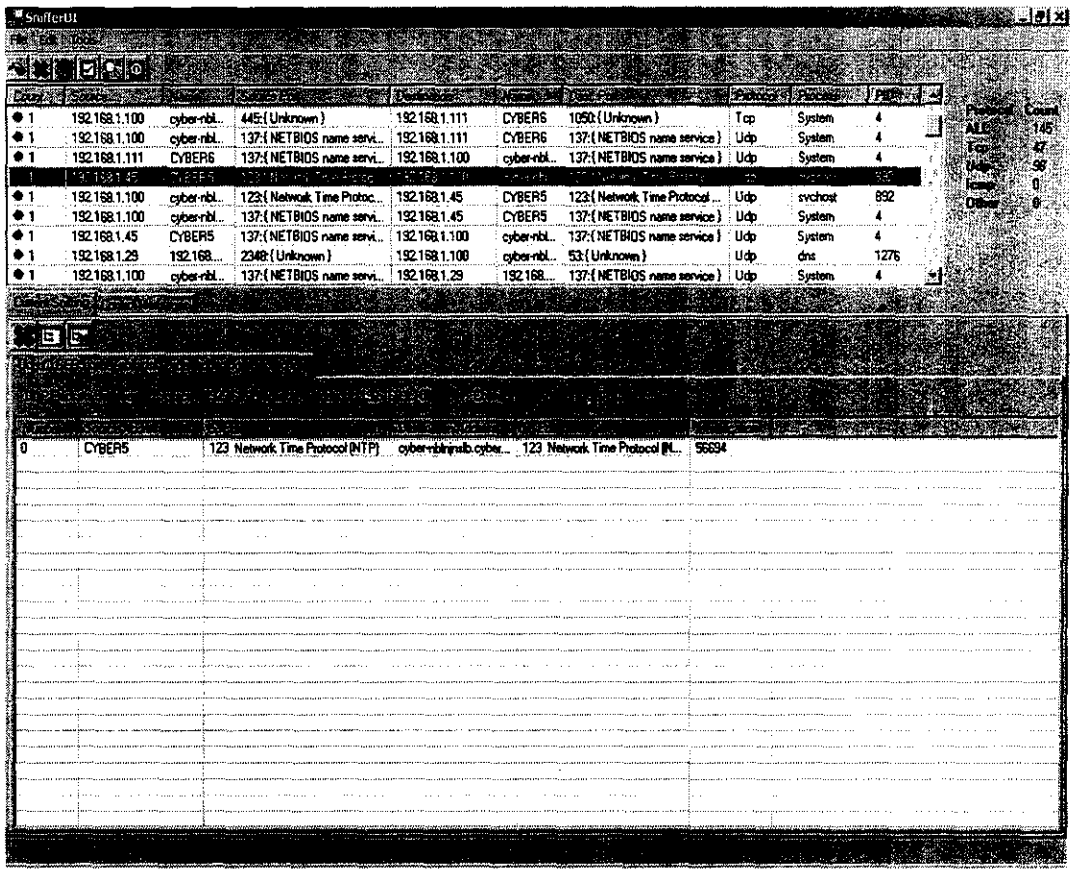

Figure A.3: The details have been displayed.

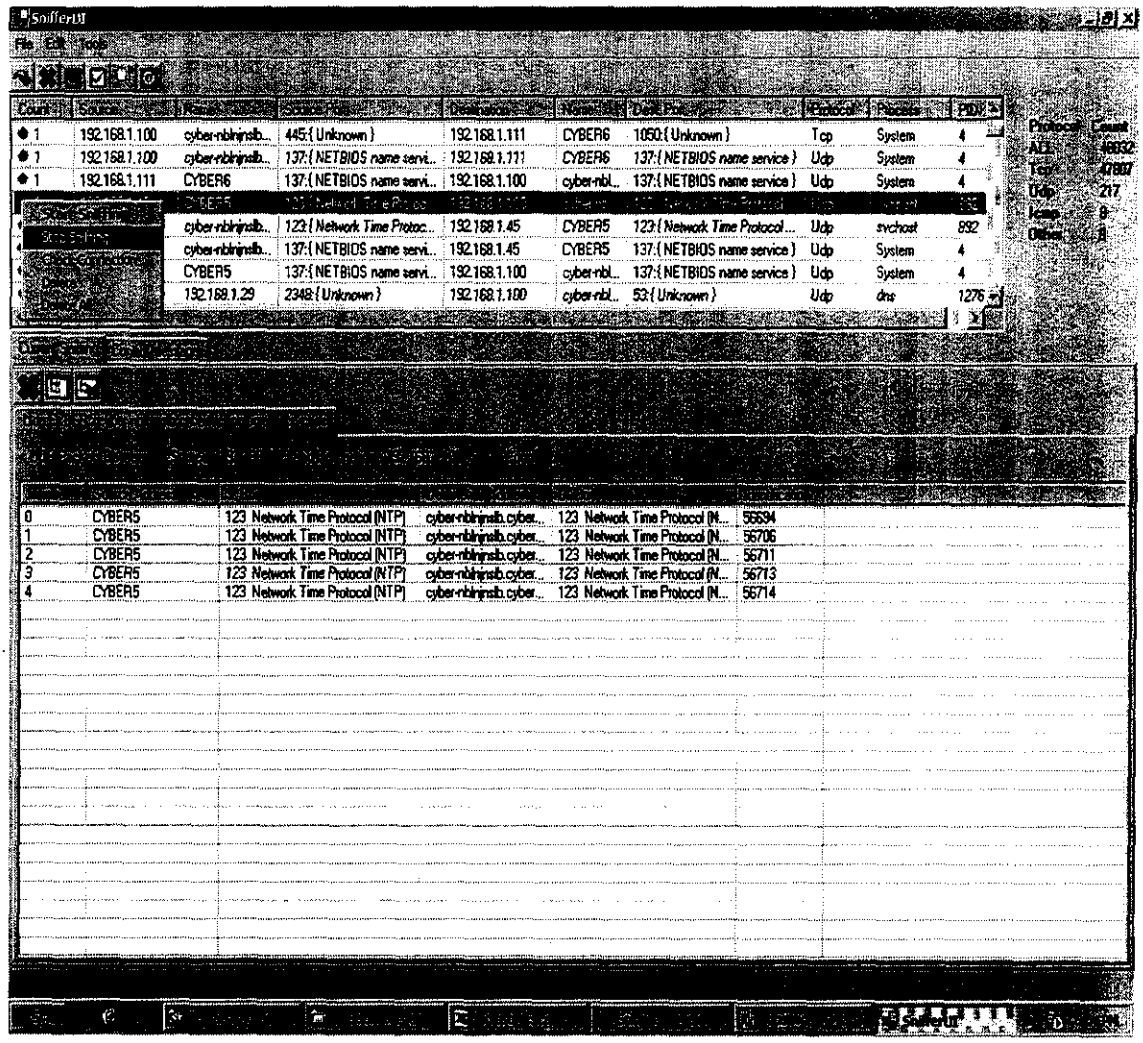

Figure A.4:.A particular source is selected and stop sniffing function is selected to show its usage.

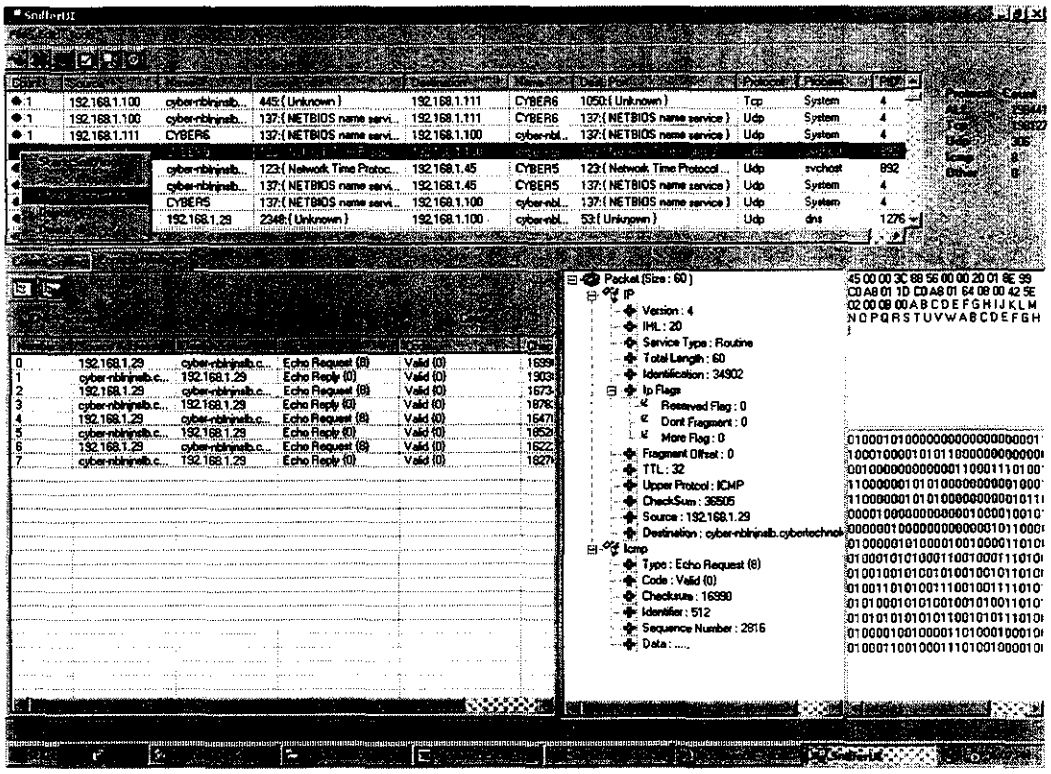

Figure A.5: Connection Establishment is checked out.

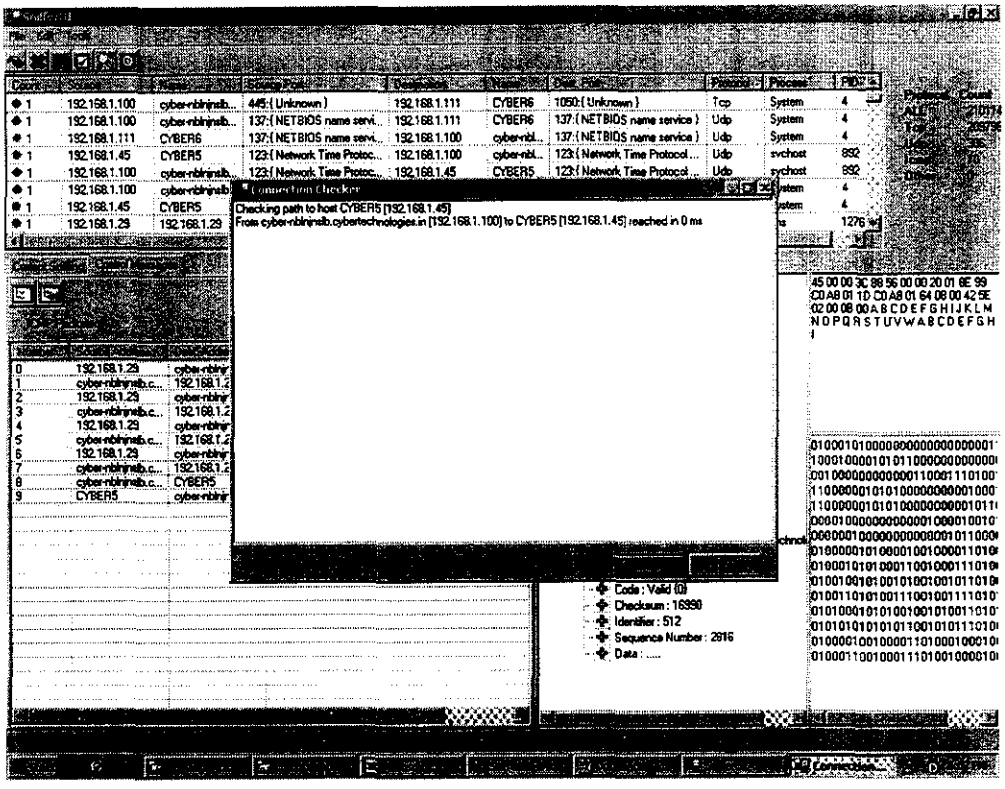

Figure A.6: Connection establishment is proved.

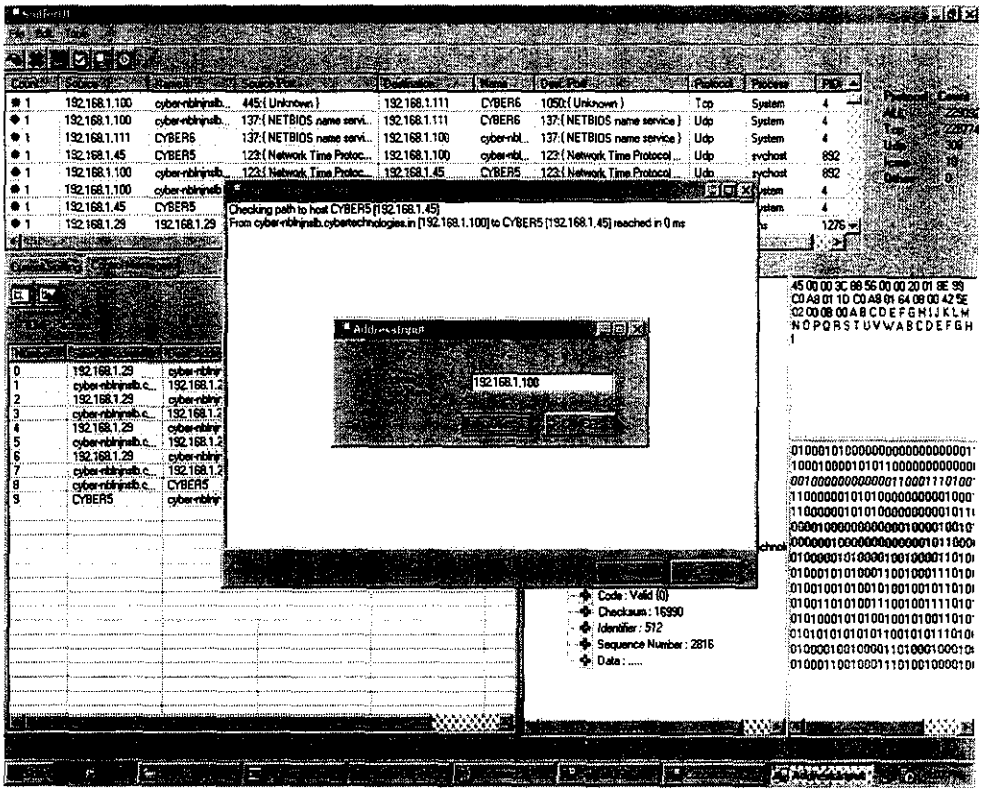

Figure A.7: The IP address is entered to establish connection.

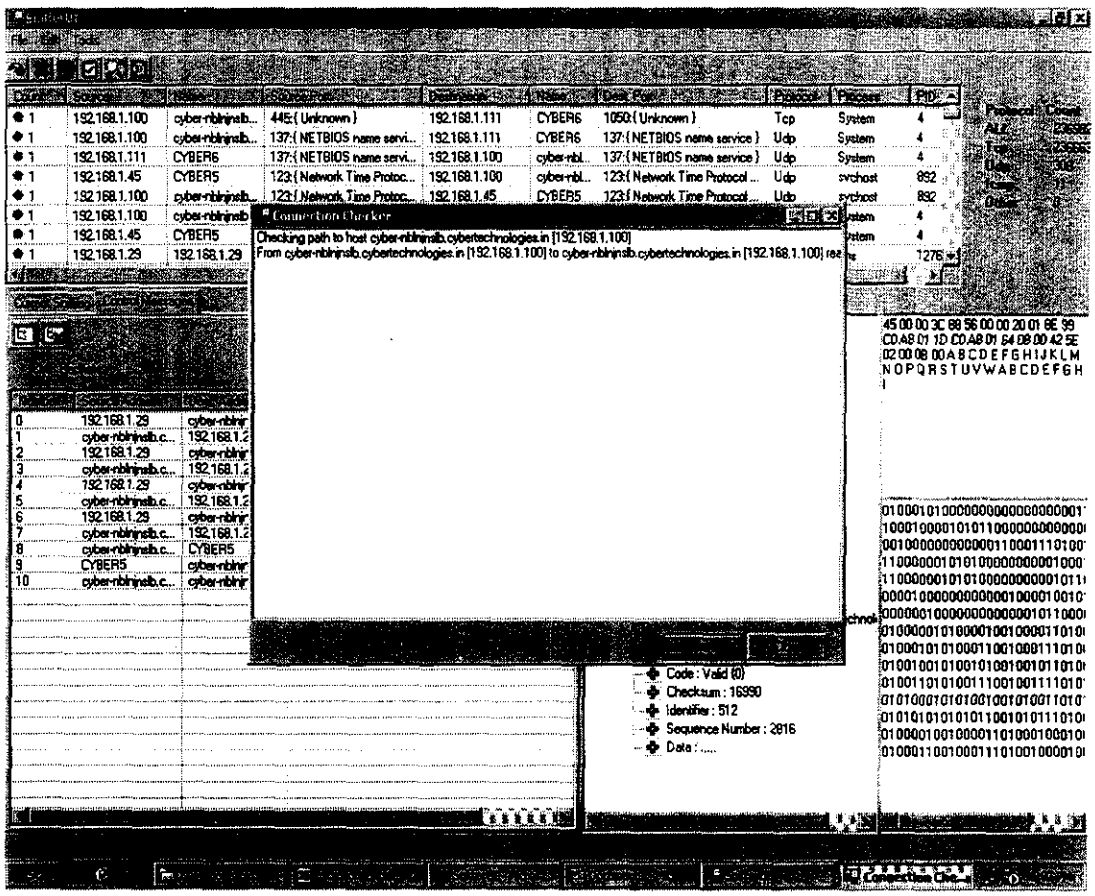

Figure A.8:.New connection is reached.

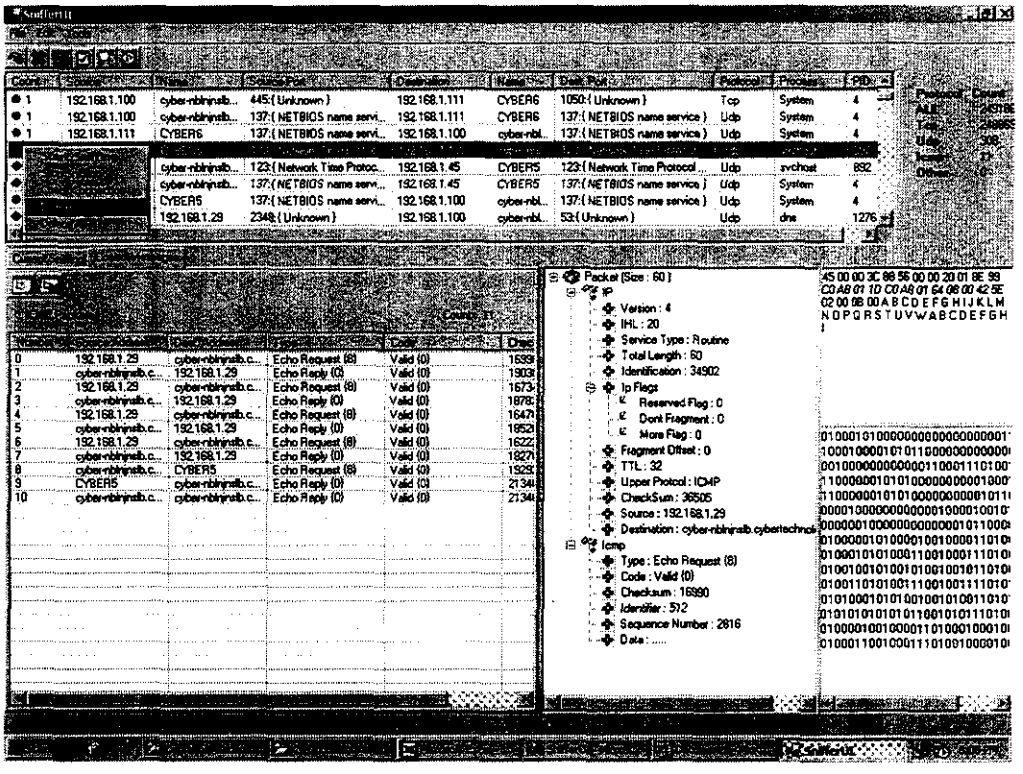

Figure A.9: Deletion process carried out.

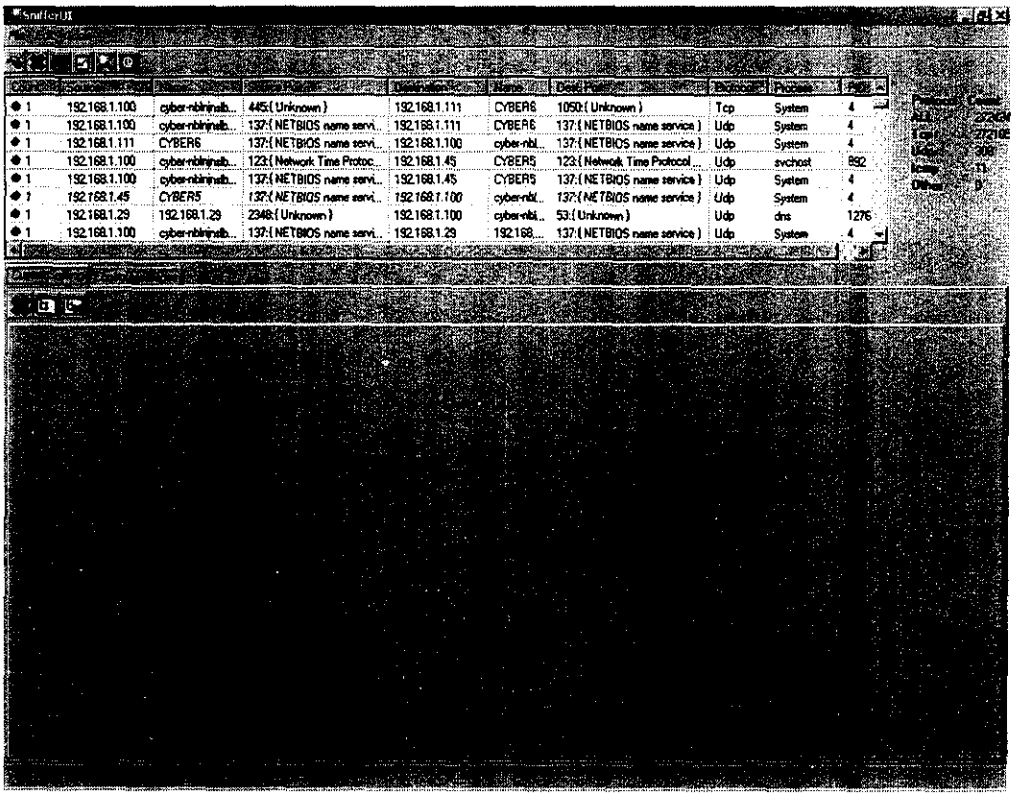

Figure A.10.Table after deletion.

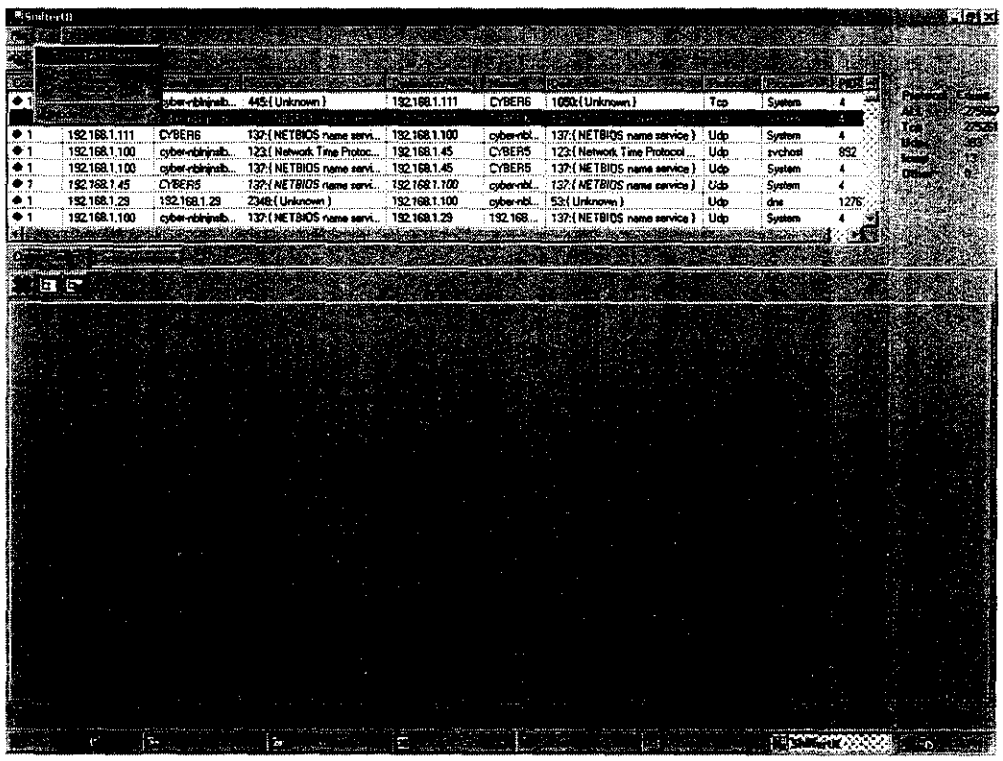

Figure A.11.Particular address is traced out.

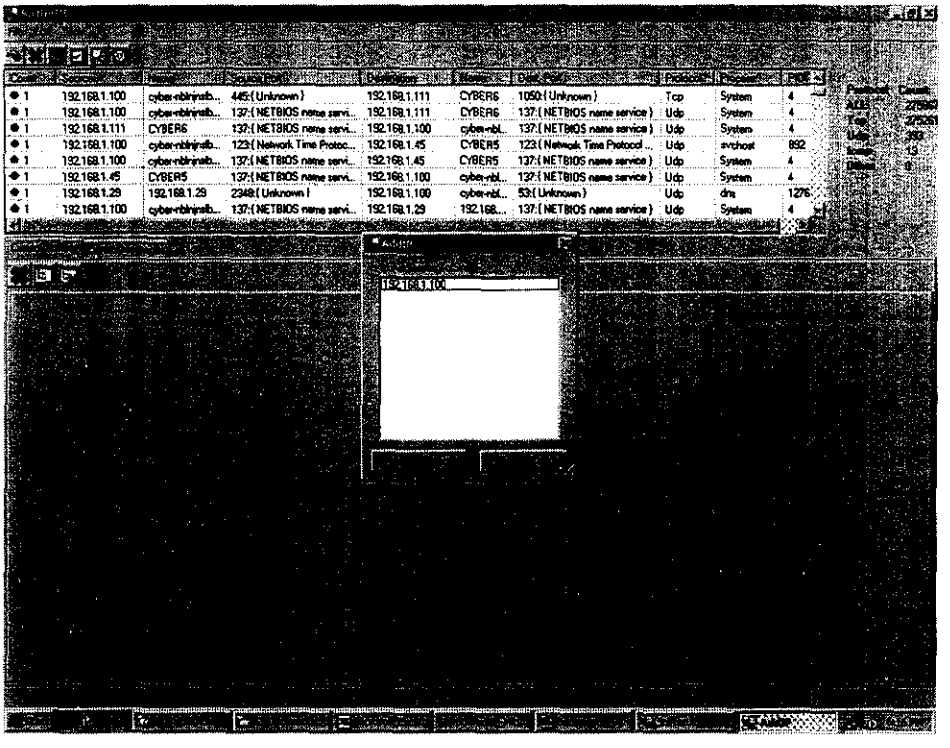

Figure A.12: Address is displayed.

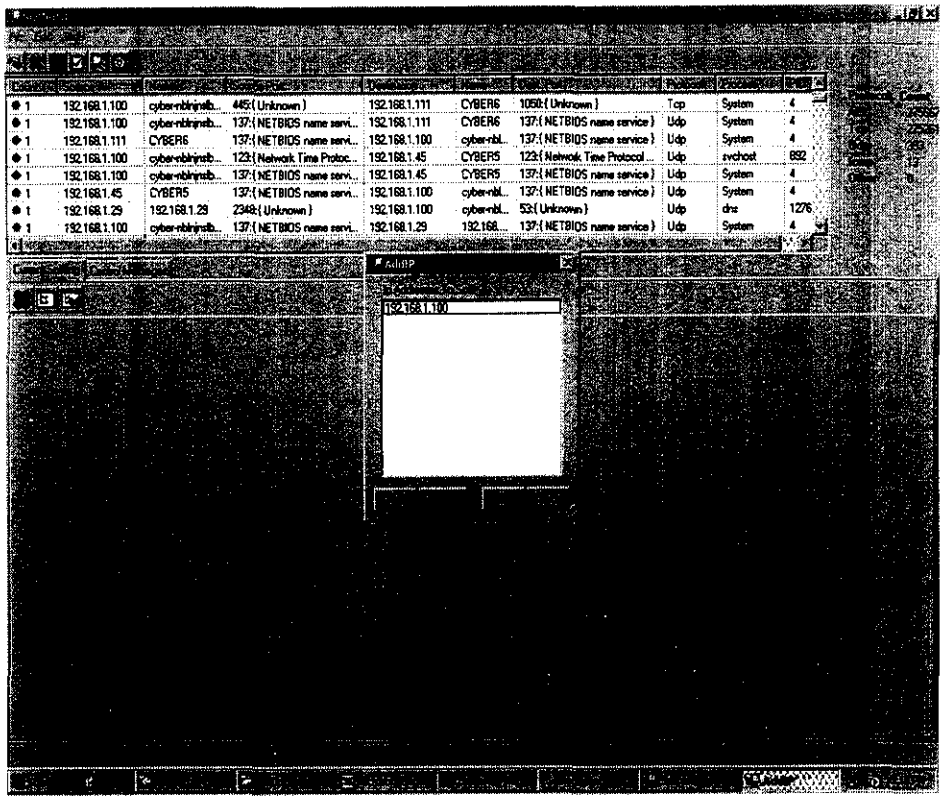

Figure A.13.Address is displayed.

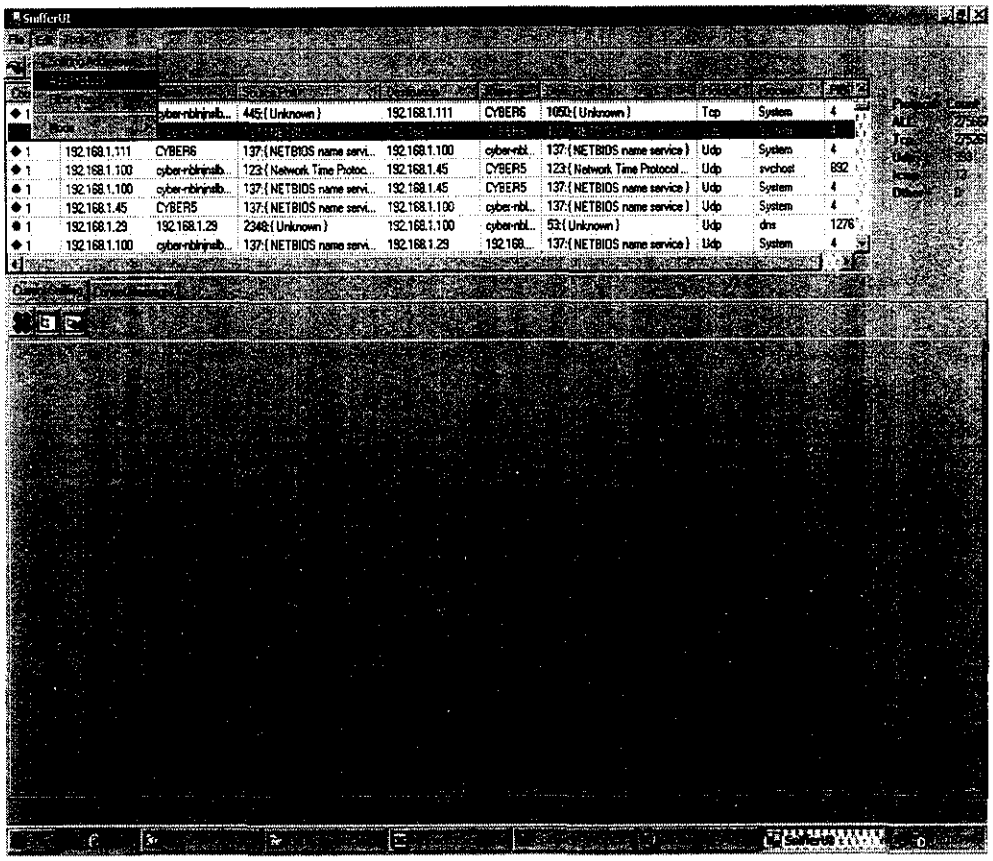

Figure A.14: Adding protocol.

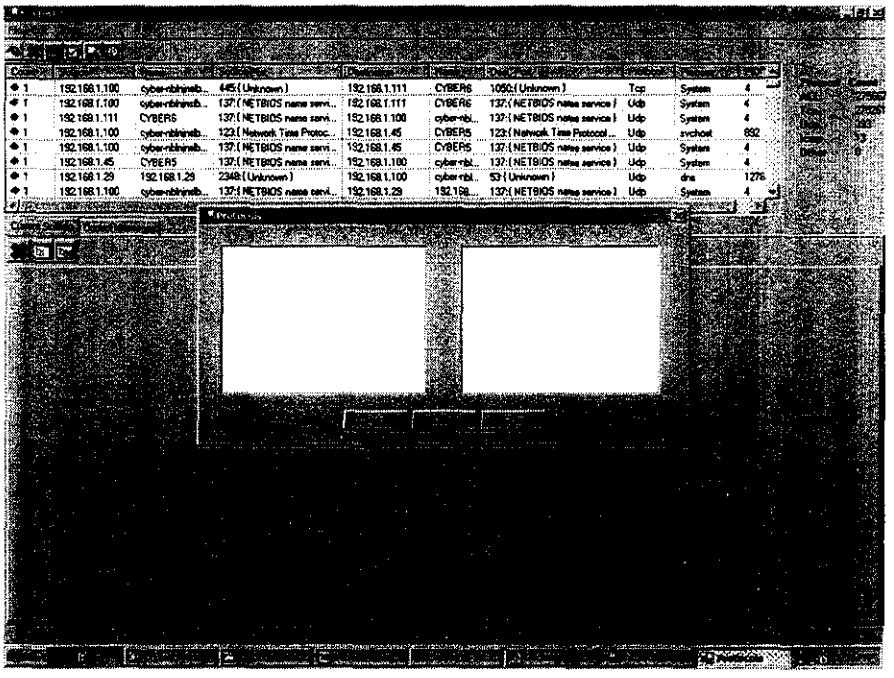

Figure A.15. To add, delete protocol option is available.

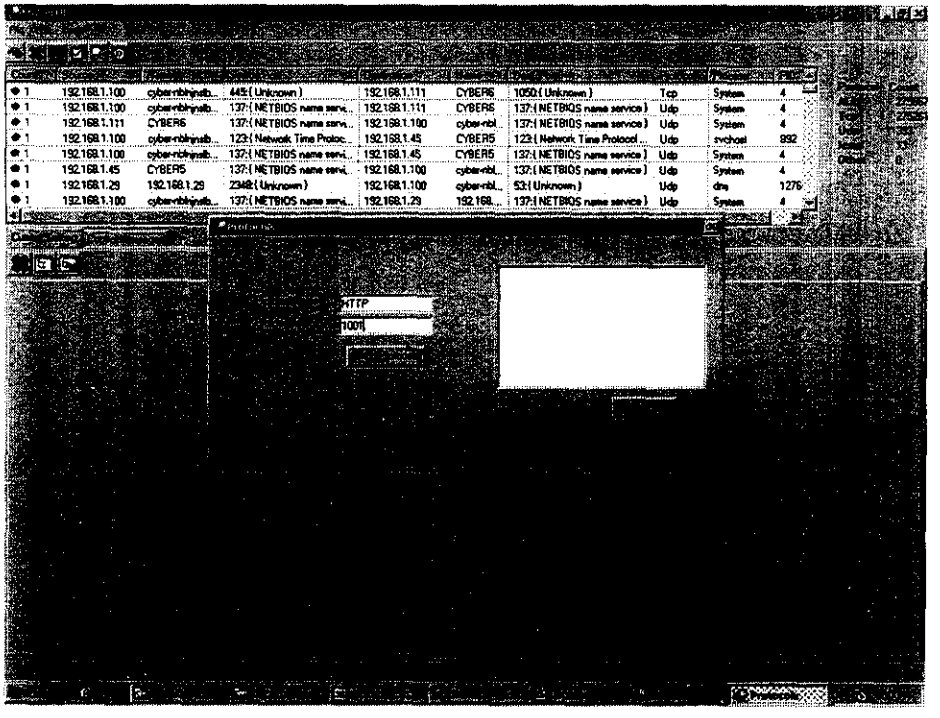

Figure A.16: Name and ID is entered.

Figure A.17.Filter Manager is worked out.

REFERENCES

BOOK REFERENCE

- Davis Chapman, **".Net Fundamentals"**, First Indian Edition-1998, Tec media.
- David J.Krulinsh, Scot Wingo and George Shepherd, **"Programming**
- **" Microsoft Visual Studio.NET "**, Second Edition – 2000, Blue Soft.
- **Joseph Albahari,** " C# 3.0 in a Nutshell ", Sep 2007, " C# 3.0 Pocket Reference", Feb2008.

WEB REFERENCES

[1] www.codeguru.com

[2] www.microsoft.com

[3] www.vbcode.com