



Jaypee University of Information Technology
Solan (H.P.)

LEARNING RESOURCE CENTER

Acc. Num. SP06071 | Call Num:

General Guidelines:

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP06071

DESIGNING A FRAMEWORK FOR ENHANCING INFORMATION SECURITY

By

ANIL KUMAR 061210

ANIL SHARMA 061211

DEEPAK BAGGA 061232

ROHIT NEGI 061283



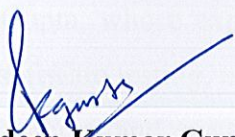
MAY – 2010

***Submitted in partial fulfillment of the Degree of Bachelor
of Technology***

**DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING
JAYPEE UNIVERSITY OF INFORMATION
TECHNOLOGY – WAKNAGHAT**

CERTIFICATE

This is to certify that the project report entitled “**Designing a framework for enhancing information security**”, submitted by Anil Kumar (061210), Anil Sharma (061211) and Deepak Bagga (061232) and Rohit Negi (061283) in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science Engineering to Jaypee University of Information Technology, Wagnaghat, Solan has been carried out under my supervision.



Sh. Pradeep Kumar Gupta

Department of Computer Science Engineering and Information Technology

Jaypee University of Information Technology

Wagnaghat.

ACKNOWLEDGEMENT

The implementation of a framework for enhancing information security over a network using a packet sniffer and firewall has presented us with an opportunity to use the technical know-how to create a real time system.

Learning through the project under the guidance of our esteemed mentor Sh. Pradeep Kumar Gupta, whose expertise knowledge in the domain of information system as well as software testing, not only cleared all our ambiguities but also generated a high level of interest and gusto in the subject. We are truly grateful for his guidance and support throughout the project. We would also like to thank our Head of the Department, Brig (Retd.) SP Ghrera for his undying faith in the department of computer science and allocation of the project as well as its resources.

The prospect of working in a group with high level of accountability fostered a spirit of team work and created a feeling of oneness which thus, expanded our ken, motivated us to perform to the best of our ability and create a report of the highest quality.

To do the best quality work, with utmost sincerity and precision has been our constant endeavor.

Anilkumar	Anil Sharma	Deepak Bagga	Rohit Negi
(Anilkumar)	(Anilsharma)	(Deepak Bagga)	(Rohit Negi)
061210	061211	061232	061283

TABLE OF CONTENT

1. General Discussion.....	09
1.1 Introduction.....	09
1.2 Problem Statement.....	10
1.3 Objective & Scope of the Project.....	11
2. Literature Survey.....	12
3. System Analysis and Design.....	20
3.1 Data flow Diagram.....	20
3.2 Use case of Firewall.....	21
3.3 Sequence Diagram.....	23
3.4 Activity Diagram.....	25
4. Implementation Details.....	29
4.1 JAVA.....	29
4.1.1 Advantages Of Java.....	30
4.1.2 Disadvantages of Java.....	34
4.2 NETBEANS.....	35
4.2.1 Advantages Of Netbeans.....	36
5. System Snapshots.....	37
5.1 Startup Window showcasing Packet sniffing before starting the process	37
5.2 Window Showcasing packets information displayed by packet Sniffer.....	38
5.3 Window Showcasing TCP/IP information of packets being transferred	40
5.4 Snapshot showcasing option bar to select network adaptors	43
6. Conclusion	44
7. Future Work	45
8. Bibliography	46
9. Appendix	47

LIST OF FIGURES

1. Fig 3.1	Level 0 DFD.....	20
2. Fig 3.2	Level 1 DFD.....	20
3. Fig 3.3	Use Case Diagram Accept Packet.....	22
4. Fig 3.4	Use Case Diagram Deny Packet.....	23
5. Fig 3.5	Sequence Diagram Accept Packet.....	24
6. Fig 3.6	Sequence Diagram Deny Packet.....	25
7. Fig 3.7	Sequence Diagram Source IP.....	26
8. Fig 3.8	Sequence Diagram Dest. IP.....	27
9. Fig 3.9	Activity Diagram of Firewall.....	28
10. Fig 5.1	Startup Window of packet sniffer.....	37
11. Fig 5.2	Window Showcasing packets information.....	38
12. Fig 5.3	Window Showcasing TCP/IP information of packet being transferred.....	40
13. Fig 5.4	Option bar to select network Adaptors.....	43

LIST OF ABBREVIATIONS

DFD	Data flow Diagram
MAC	Media Access control Address
TCP/IP	Transmission Control protocol / Internet Protocol
IDE	Independent Development Environment
JDK	Java Development kit
SDK	Software Development Kit
JVM	Java Virtual machine
GUI	graphical user Interface
SRC PORT	Source Port Address
HTTP	Hyper Text Transfer Protocol
ACK	Acknowledgement
WWW.	World Wide Web
SYN	Synchronization

ABSTRACT

This document will give you insight in the implementation of "A Framework for enhancing information security". In this document we have described the basic steps involved in monitoring packet transfer over a network and protecting a terminal's information security. Project literature forms an important part of this document as we have explained in detail the work done in this area. Followed by the description of the project we have given an overview of the concepts used in the implementation of the project. Further we have written a code for monitoring network traffic and terminating unwanted access to the network. Then we have tried implementing the project in real time and we have explained the problems and limitations of the product being produced and tested our product in real time. Finally we draw a conclusion and show our result. This project takes the advantage by studying the flaws of similar products and works in this area of technology and tries to overcome them in real time.

CHAPTER 1

General Discussion

1.1 Introduction

Due to increasing rise in networking tools there has been a constant threat to the data of individuals or a firm. Since such a important datas reside on the computers nowadays, it becomes quite important to protect that data from the people from who are not authorized to have access to that data . at the same time we also need to make sure that every terminal who is authorized to get data gets that , that too in correct form. Not only security of the data is the requirement but integrity of that data is also very much important. And we need to make sure that the data sent is not only sent but also in the exact form as we wish to send it, because wrong data is worse than the no data. We need to protect our computers against the hackers, disgruntled employees, professional spies etc. Due to increased focus on network security, network administrators often spend more effort protecting their networks than on actual network setup and administration

1.2 Problem Statement

Due to increased use of network tools in our day to day activities our computers and information stored in them becomes vulnerable to the world .we want to access World Wide Web without allowing everyone to peak into our business. Hence we need a mechanism that enhances our information security. Internet connection leaves you vulnerable to hackers who want to access your financial and personal information. Some hackers may be after your high-speed connection so that they can send malicious viruses and worms, blackening your reputation. Other intruders have the power to destroy your operating system on a whim. How can you lock that computer door but still have the freedom to do your business online. Through this project we intend to erect a barrier between an unreliable piece of software, your sensitive information that resides on your private network The firewall is part of an overall security policy that creates a perimeter defense designed to protect the information resources of the organization. Other than a firewall a packet sniffer is also a part of our framework for enhancing information security in information resources of a firm. A Packet sniffer or analyzer is computer software that can intercept the traffic passing over a network or part of a network. As data streams flow across the network, the sniffer captures each packet and, if needed, decodes and analyzes its content according to the appropriate specifications. For network monitoring purposes it may also be desirable to monitor all data packets in a LAN by using a packet sniffer whose purpose is to mirror all packets passing through all ports of the switch when systems (computers) are connected to a switch port.

Here we are suggesting a solution that enhances information security in information resources of a computer using a packet sniffer and a firewall.

1.3 Objective and Scope of the Project

Objective of the following project is to make sure that whatever packets we send or receive over a network are safe for transfers. This project will monitor all the packets being transferred over the network and make sure they are safe for transmission.

- Inspects each individual “packet” of data as it arrives at either side of the firewall.
- Inbound to or outbound from your computer.
- Determines whether it should be allowed to pass through or if it should not be.

After all the packets being analyzed a firewall will be used for ip blocking for the purpose of terminating all the unwanted connections and blocking subsequent ports.

IP Blocking prevents the connection between a server/website and certain IP addresses or ranges of addresses. IP blocking effectively bans undesired connections from those computers to a website, mail server, or other Internet server. IP banning is commonly used on to protect against brute force attacks. .It is also used for censorship.

CHAPTER 2

Literature Survey

The implementation of this project requires an extensive knowledge of the following –

Information security

Packet sniffer

Firewall

IP sec

Jpcap

Winpcap

Information Security

- Information security is intended to achieve confidentiality, availability, and integrity in the firm's information resources
 - **Confidentiality:** protecting a firm's data and information from disclosure to unauthorized persons
 - **Availability:** making sure that the firm's data and information is only available to those authorized to use it
 - **Integrity:** information systems should provide an accurate representation of the physical systems that they represent
 - Authentication –passwords, biometric devices
 - Access control-a policy by the organization to decide who will access to what
 - Resource isolation-so that damage is contained
 - Network Perimeter Protection

- **Vulnerabilities**
 - security flaws in systems
- **Attacks**
 - means of exploiting vulnerabilities
- **Countermeasures**
 - technical or procedural means of addressing vulnerabilities or thwarting specific attacks
- **Threats**
 - motivated adversaries capable of mounting attacks which exploit vulnerabilities
 - Hackers
 - Disgruntled employees
 - Industrial spies
 - Special interest groups
 - Journalists

Packet Sniffer

Packet, in computer communications, the basic unit of data over a network such as Internet .A message to be transferred the network is broken up into small units, or packets .by the sending the packets, which travel independently of one another are with the sender's address, destination address , and other pertinent . Including data about any errors introduced during the transfer, the packets arrive at the receiving computer, they are reassembled.

Packet sniffing is used within a network in order to capture and register data flows. Packet sniffing allows you to discern each individual packet and analyze its content based on predefined parameters. Packet sniffing allows for very detailed network monitoring and bandwidth usage analysis. It, however, requires a broader knowledge of networks and their inner functions, in order to be able to recognize the relevance of the data being monitored. Packet sniffer is a computer software or hardware that can intercept and log traffic passing over a digital network. Packet sniffing is the monitoring of data traffic on a computer work. Computers communicate over the internet by breaking up messages (emails, images, videos, web pages, files, etc) into small chunks called "packets", which are routed through a network of computers, until they reach their destination, where they are assembled back into a complete "message" again. Packet sniffers are programs that intercept these packets as they are traveling through the network, in order to examine their contents using other programs. A packet sniffer is an information gathering tool, but not an analysis tool. That is it gathers "messages" but it does not analyze them and figure out what they mean.

Advantages of a Packet Snifer

- The versatility of packet sniffers means they can be used to:
- Analyze network problems
- Detect network intrusion attempts
- Gain information for effecting a network intrusion
- Monitor network usage
- Gather and report network statistics
- Filter suspect content from network traffic

Firewall

Firewall is used to achieve following objectives to provide the people with access to the WWW without allowing the entire world to peak in, to erect a barrier between an unreliable pieces of software, your sensitive information that resides on your private network

- Firewalls are the devices that limit network access
- Firewalls are access control mechanisms
- Consistency, Completeness, and Compactness
- The firewall is part of an overall security policy that creates a perimeter defense designed to protect the information resources of the organization.

Firewall strengths

- Disallows incoming connections to hosts that do not offer public network services
- Allows administrators tools to control their networks from the inside and outside (i.e. do you allow your users to get access to the Web?)

Firewall weaknesses

- Because of increasing line speeds and computation-intensive protocols the firewall can become a congestion point
- There exist protocols that are difficult to process at the firewall
- Classical firewall design assumes that all internal users can be trusted

Firewall Does

- Implement security policies at a single point
- Monitor security-related events
- Provide strong authentication
- Have a specially hardened/secured operating system

Firewall doesn't

- Protect against attacks that bypass the firewall
 - Dial-out from internal host to an ISP
- Protect against the transfer of virus-infected programs or files

Types of Firewalls

- Packet-Filtering Router
- Application-Level Gateway
- Circuit-Level Gateway
- Hybrid Firewalls

IP sec

- general IP Security mechanisms
- provides
 - authentication
 - confidentiality
 - key management
- applicable to use over LANs, across public & private WANs, & for the Internet

Benefits of IP sec

- In a firewall/router provides strong security to all traffic crossing the perimeter and is resistant to bypass.
- It is below transport layer and hence transparent to applications and transparent to end users.
- It can provide security for individual users and secures routing architecture.

Jpcap

Jpcap is an open source library for capturing and sending network packets from Java applications. Jpcap is based on libpcap/winpcap, and is implemented in C and Java. It provides facilities to:

- Capture raw packets live from the wire.
- Save captured packets to an offline file, and read captured packets from an offline file.
- Automatically identify packet types and generate corresponding Java objects (for Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP, and ICMPv4 packets).
- Filter the packets according to user-specified rules before dispatching them to the application.
- send raw packets to the network.

Limitations of Jpcap

Jpcap captures and sends packets independently from the host protocols (e.g., TCP/IP). This means that Jpcap does not (cannot) block, filter or manipulate the traffic generated by other programs on the same machine: it simply "sniffs" the packets that transit on the wire. Therefore, it does not provide the appropriate support for applications like traffic shapers, QoS schedulers and personal firewalls.

WinPcap

WinPcap is an open source library for packet capture and network analysis for the Win32 platforms. Most networking applications access the network through widely used operating system primitives such as sockets. It is easy to access data on the network with this approach since the operating system copes with the low level details and provides a familiar interface that is similar to the one used to read and write files. WinPcap is the industry-standard tool for link-layer network access in Windows environments: it allows applications to capture and transmit network packets bypassing the protocol stack, and has additional useful features, including kernel-level packet filtering, a network statistics engine and support for remote packet capture. WinPcap consists of a driver, which extends the operating system to provide low-level network access, and a library that is used to easily access the low-level network layers. Thanks to its set of features, WinPcap is the packet capture and filtering engine of many networking tools, including protocol analyzers, network monitors, network intrusion detection systems, sniffers, traffic generators and network testers.

Facilities provided by WinPcap :

It capture raw packets, both the ones destined to the machine where it's running and the ones exchanged by other hosts. filter the packets according to user-specified rules before dispatching them to the application, transmit raw packets to the network and gather statistical information on the network traffic.

Network Adaptors

A Network adaptor is a hardware device that handles an interface to a computer network and allows a network-capable device to access that network. The NIC has a ROM chip that contains a unique number, the media access control (MAC) Address burned into it. The MAC address identifies the device uniquely on the LAN. The NIC exists on the 'Data Link Layer' (Layer 2) of the OSI model. It is a computer hardware component designed to allow computers to communicate over a computer network. It is both an OSI layer 1 (physical layer) and layer 2 (data link layer) device, as it provides physical access to a networking medium and provides a low-level addressing system through the use of MAC addresses. It allows users to connect to each other either by using cables or wirelessly. Hence In a computer network, each machine must have a network card installed to communicate with the network router. The router directs traffic on the local network and also handles requests made to the Internet, and subsequent responses. The network card in each machine must support the same protocols as the network router so that all devices are speaking the same language. A network adapter can substitute for an internal network card when a card is not present, or when the internal card does not support the required standard.

CHAPTER 3

System Analysis and Design

3.1 Data Flow Diagram

LEVEL 0 DFD

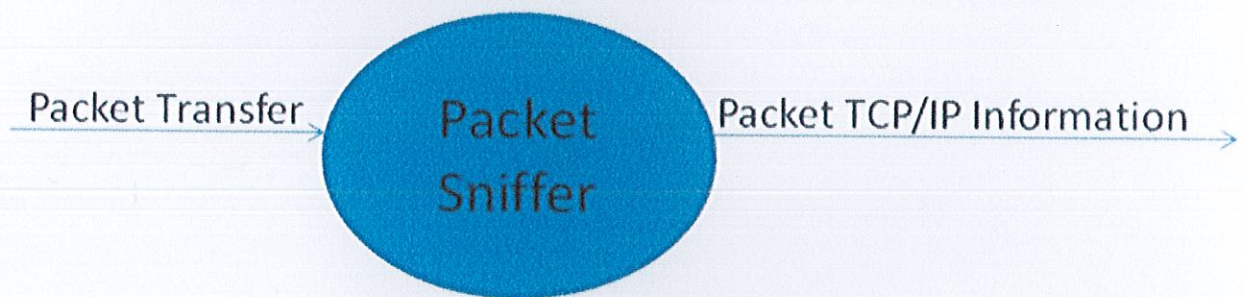


Fig 3.1 Level 0 DFD

LEVEL 1 DFD

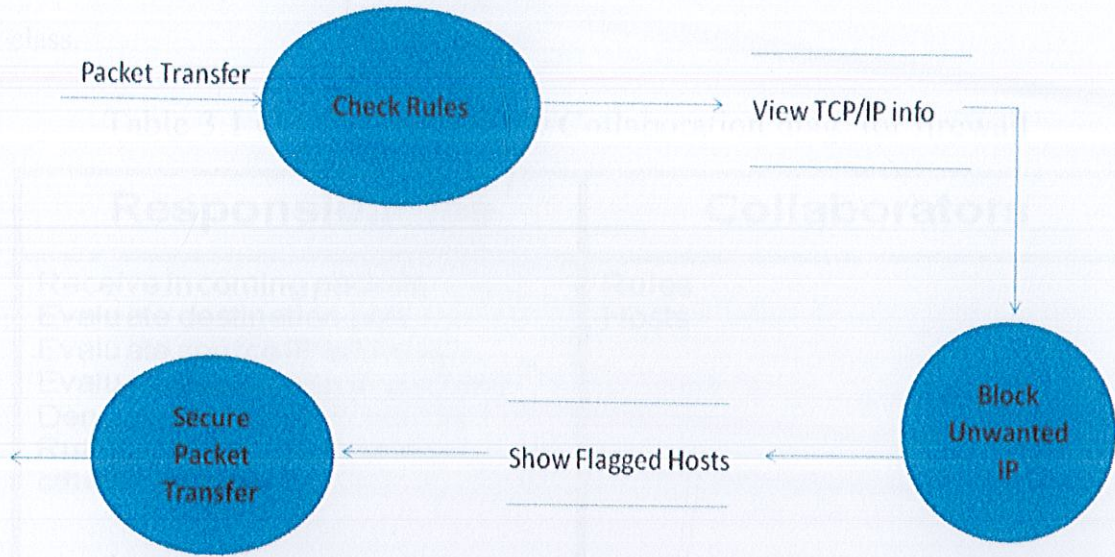


Fig 3.2 Level 1DFD



3.2 Class Responsibility Collaboration

CRC (Class Responsibility Collaboration) :- It is used to find the characteristics of a class.

Table 3.1 Class responsibility Collaboration diag. for firewall

Responsibilities	Collaborators
Receive incoming packets Evaluate destination port Evaluate source IP address Evaluate destination IP address Deny packet if not allowed by Rules in above evaluations, otherwise Accept.	Rules Hosts

Table 3.2 CRC for Hosts and Rules

Host	
Responsibilities	Collaborators
Send "Good" Packets Send "Bad" Packets	Firewall

Rules	
Responsibilities	Collaborators
1) Set priorities and sequence for rules by which to accept or deny packets.	Firewall
2) Provide Guidance for Accepting or Denying Packets	

3.3 Use Case Diagram

Table 3.3 To accept packet in firewall

Actor(s):	Normal Host and the Firewall
Description:	Normal host, who should have access, attempts to send a packet to the hosts located behind the firewall.
Normal Course:	The Firewall receives the packet, checks the source and destination IP addresses and ports, evaluates the IP addresses and ports against the rules, accepts the packet.
Alternative Course:	The Firewall fails to accept the packet and the Normal Host is denied access to the hosts behind the firewall.
Pre-condition:	1) All internal hosts are behind the firewall with no "back doors" to allow internal hosts access to the internet without going through the Firewall first 2) Rules are set for deny all and then exceptions for access are located in a lower priority position.
Post-condition:	The packet is accepted by the firewall.

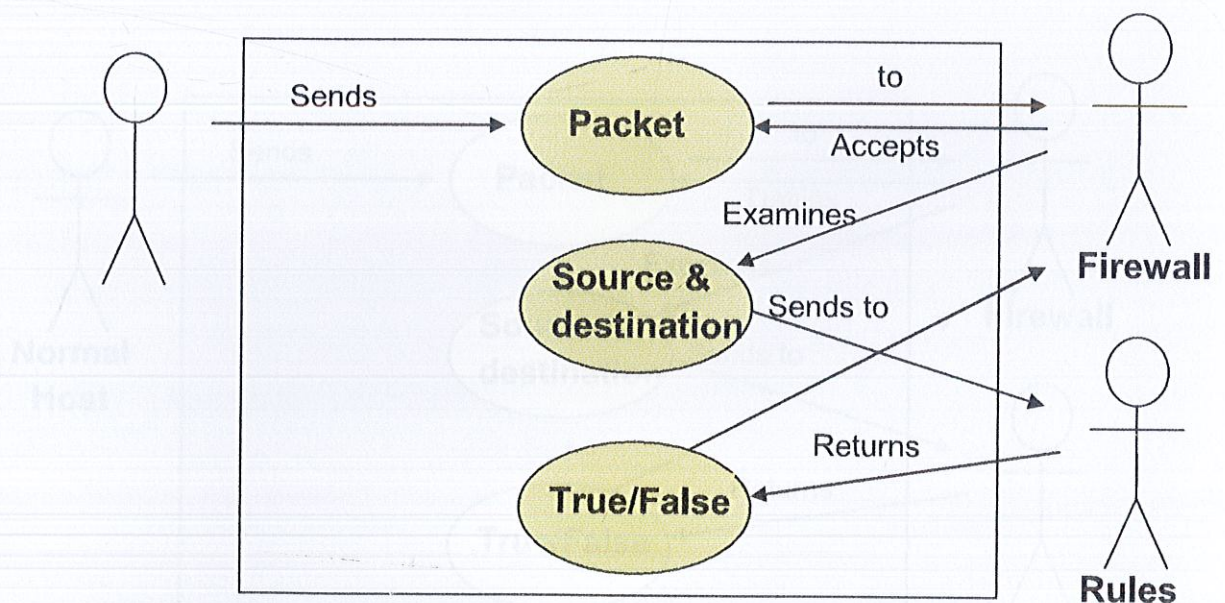


Fig 3.3 Use case diag. to accept packet in firewall

Table 3.4 To deny packet in firewall

Use Case Name:	Deny Packet
Actor(s):	Hacker Host and the Firewall
Description:	Hacker host, who should not have access, attempts to send a packet to the hosts located behind the firewall.
Normal Course:	The Firewall receives the packet, checks the source and destination IP addresses and ports, evaluates the IP addresses and ports against the rules, denies the packet.
Alternative Course:	The Firewall fails to deny the packet and the Hacker Host gains access to the hosts behind the firewall.
Pre-condition:	1) All internal hosts are behind the firewall with no "back doors" to allow internal hosts access to the internet without going through the Firewall first 2) Rules are set for deny all and then exceptions for access are located in a lower priority position.
Post-condition:	The packet is denied by the firewall.

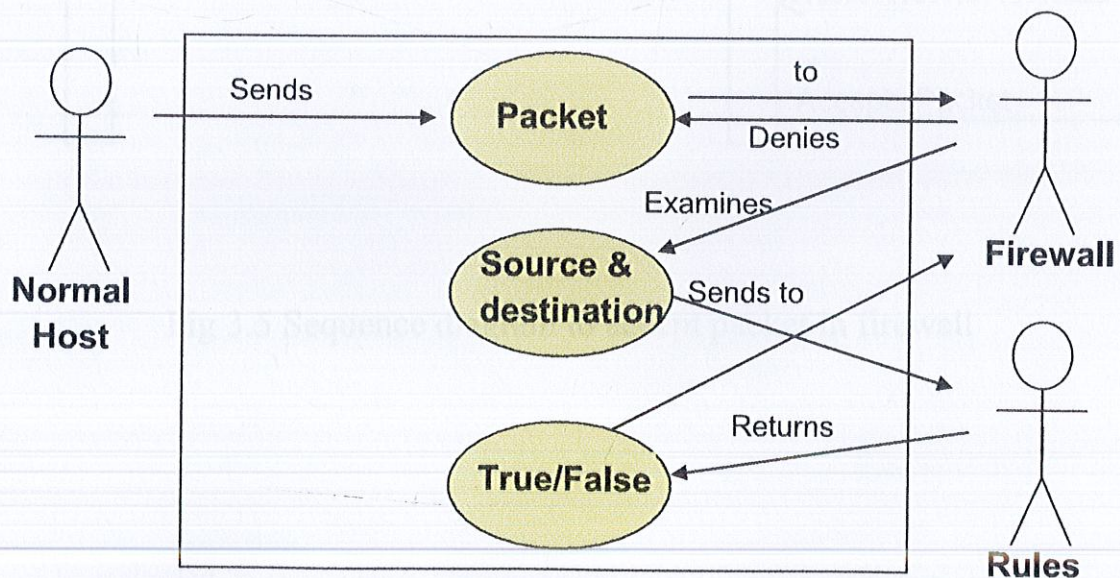


Fig. 3.4 Use case diag. to accept packet in firewall

3.4 Sequence Diagram

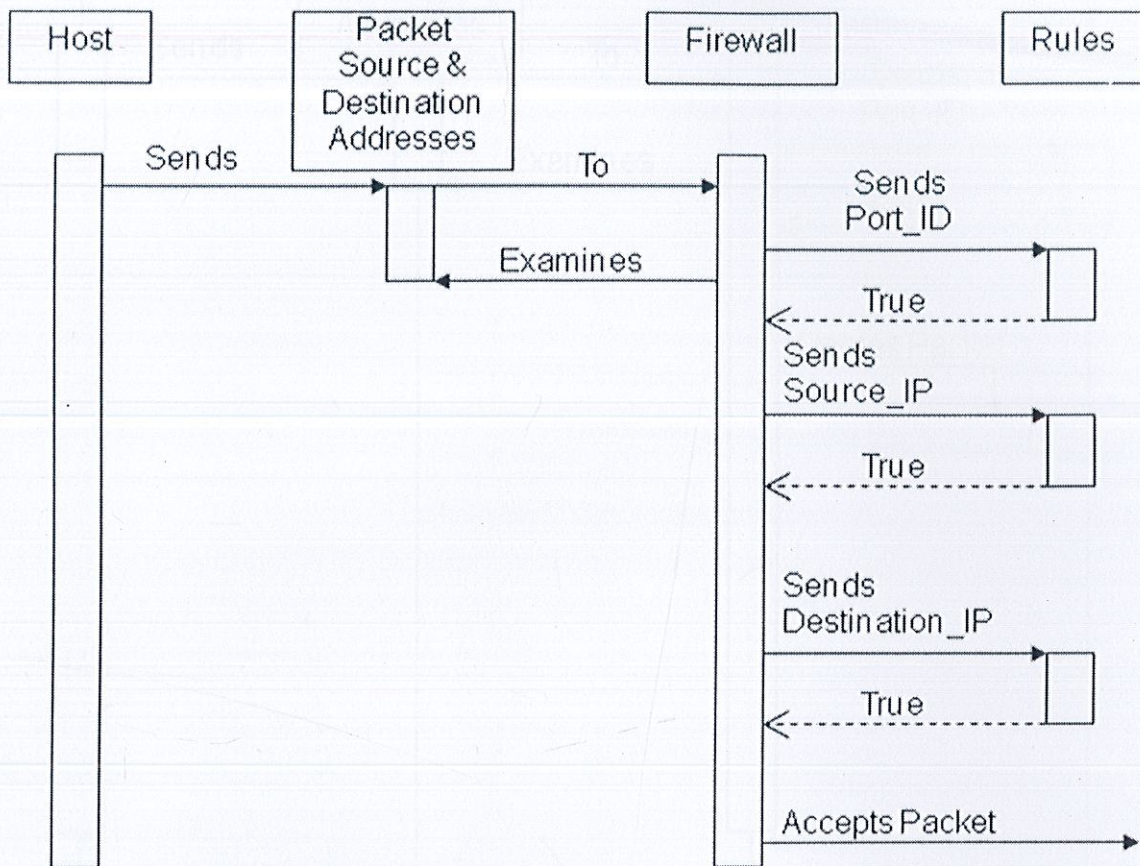


Fig 3.5 Sequence diagram to accept packet in firewall

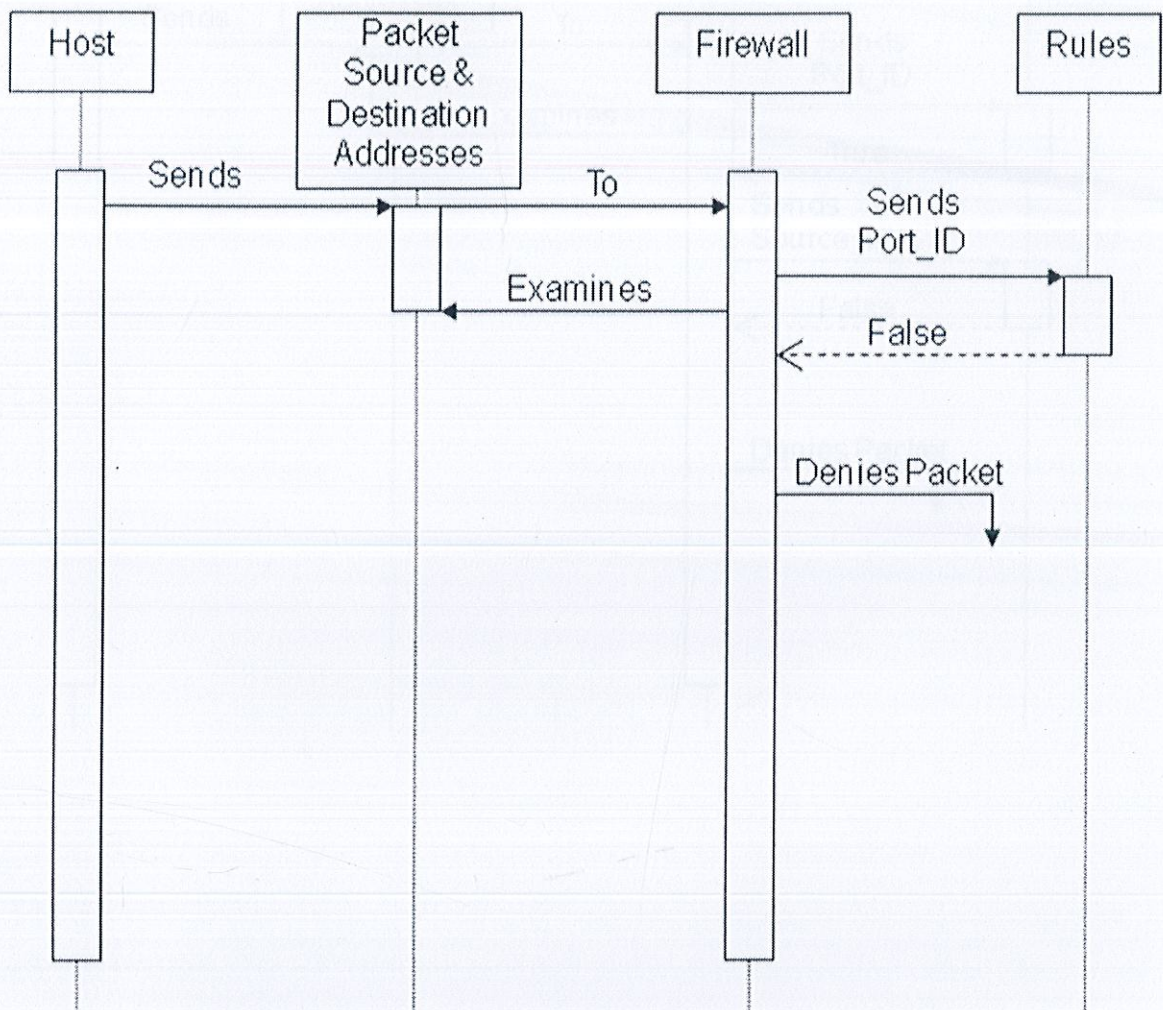


Fig 3.6 Use case diag. to deny packet in firewall

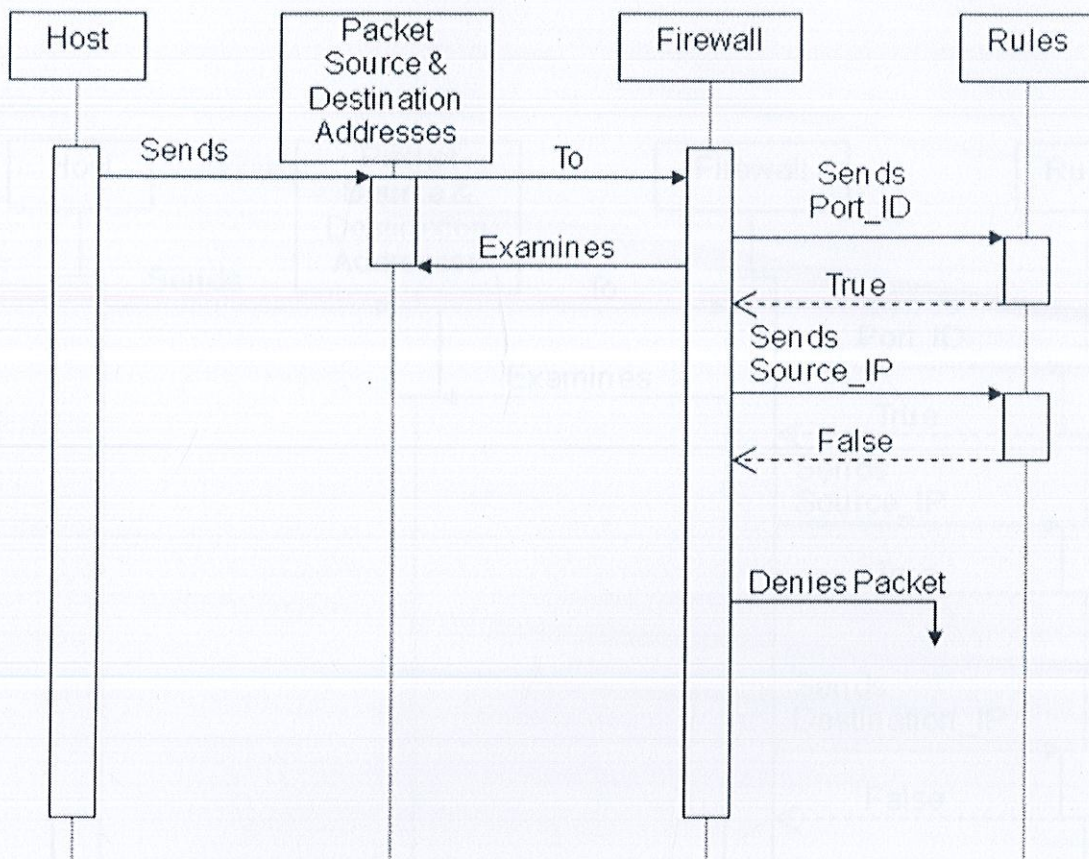


Fig 3.7 Sequence diag. to Deny Packet (Source IP)

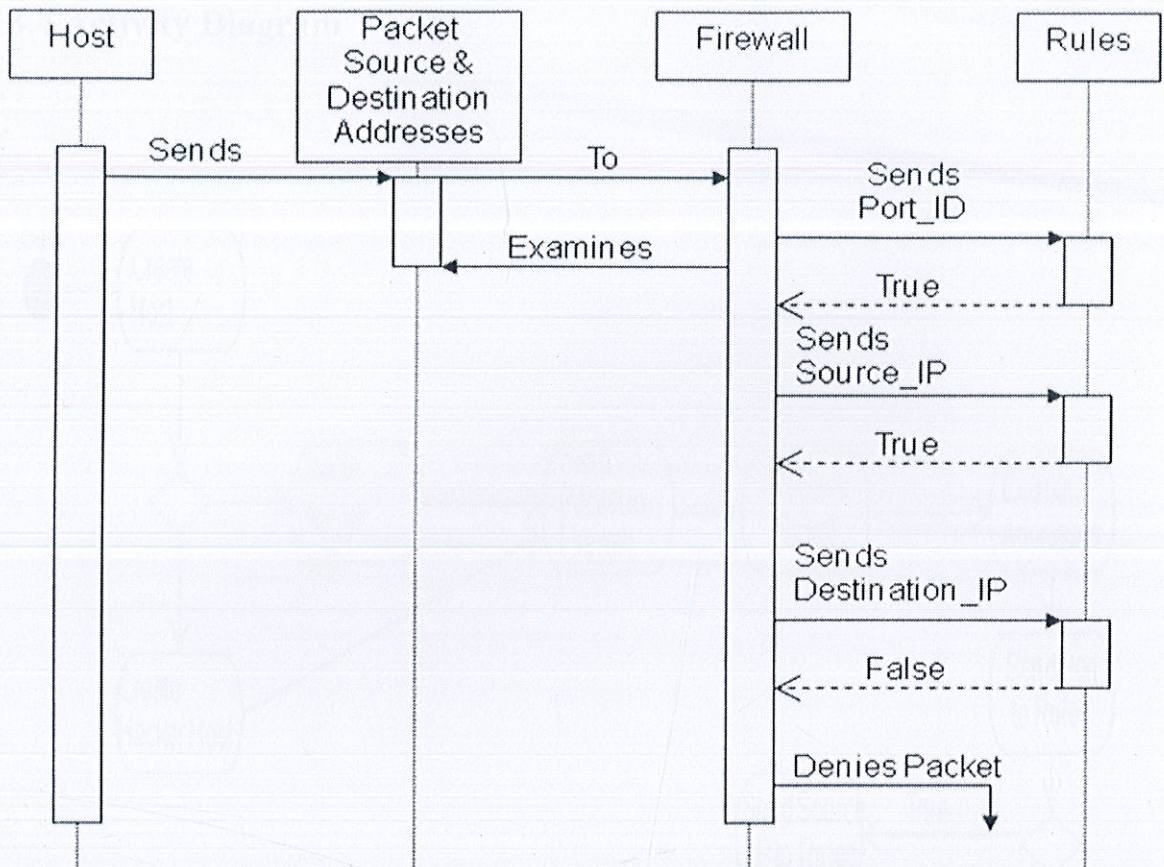


Fig 3.8 Sequence diag. to Deny Packet(Destination IP)

3.4 Activity Diagram

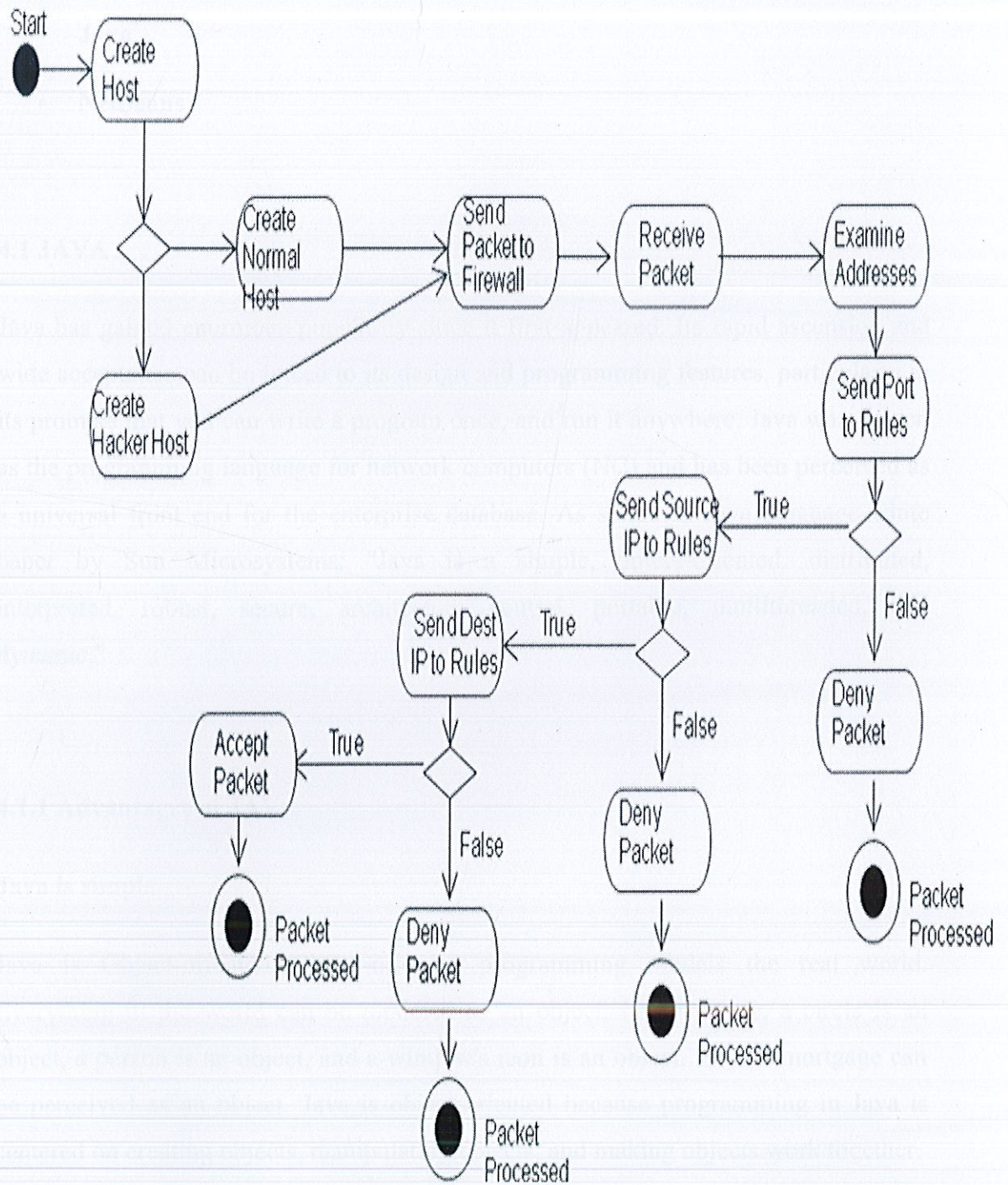


Fig 3.9 Activity diag. for checking rules in firewall

CHAPTER 4

Implementation Details

Technologies used

- **Java**
- **Netbeans**

4.1 JAVA

Java has gained enormous popularity since it first appeared. Its rapid ascension and wide acceptance can be traced to its design and programming features, particularly in its promise that you can write a program once, and run it anywhere. Java was chosen as the programming language for network computers (NC) and has been perceived as a universal front end for the enterprise database. As stated in Java language white paper by Sun Microsystems: "Java is a simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, multithreaded, and dynamic."

4.1.1 Advantages of JAVA

Java is simple

Java is Object-oriented Object-oriented programming models the real world. Everything in the world can be modeled as an object. For example, a circle is an object, a person is an object, and a window's icon is an object. Even a mortgage can be perceived as an object. Java is object-oriented because programming in Java is centered on creating objects, manipulating objects, and making objects work together.

An object has properties and behaviors. Properties are described by using data, and behaviors are described by using methods. Objects are defined by using classes in Java. A class is like a template for the objects. An object is a concrete realization of a class description. The process of creating an object class is called instantiation. Java consists of one or more classes that are arranged in a treelike hierarchy, so that a child class is able to inherit properties and behaviors from its parent class. An extensive set of pre-defined classes, grouped in packages that can be used in programs are found in Java.

Object-oriented programming provides greater flexibility, modularity and reusability. For years, object-oriented technology has been perceived as an elitist, requiring substantial investments in training and infrastructure. Java has helped object-oriented technology enter the mainstream of computing, with its simple and clean structure that allows the programmer to write easy to read and write programs.

Java is Distributed

Distributed computing involves several computers on a network working together. Java is designed to make distributed computing easy with the networking capability that is inherently integrated into it. Writing network programs in Java is like sending and receiving data to and from a file. For example, the diagram below shows three programs running on three different systems, communicating with each other to perform a joint task.

Portability: Program once, Run anywhere (Platform Independence)

One of the most compelling reasons to move to Java is its platform independence. Java runs on most major hardware and software platforms, including Windows 95 and NT, the Macintosh, and several varieties of UNIX. Java applets are supported by all Java-compatible browsers. By moving existing software to Java, you are able to make

it instantly compatible with these software platforms. JAVA programs become more portable. Any hardware and operating system dependencies are removed.

Java is Interpreted

An interpreter is needed in order to run Java programs. The programs are compiled into Java Virtual Machine code called bytecode. The bytecode is machine independent and is able to run on any machine that has a **Java** interpreter. Normally, a compiler will translate a high-level language program to machine code and the code is able to only run on the native machine. If the program is run on other machines, the program has to be recompiled on the native machine. For example, if you compile a C++ program in Windows, the executable code that is generated by the compiler can only be run on a Windows platform. With Java, the program need only be compiled once, and the bytecode generated by the Java compiler can run on any platform.

Security

Java is one of the first programming languages to consider security as part of its design. The Java language, compiler, interpreter, and runtime environment were each developed with security in mind. The compiler, interpreter, and Java-compatible browsers all contain several levels of security measures that are designed to reduce the risk of security compromise, loss of data and program integrity, and damage to system users. Considering the enormous security problems associated with executing potentially untrusted code in a secure manner and across multiple execution environments, Java's security measures are far ahead of even those developed to secure military systems. C and C++ do not have any intrinsic security capabilities.

Reliability

Security and reliability go hand in hand. Security measures cannot be implemented with any degree of assurance without a reliable framework for program execution. Java provides multiple levels of reliability measures, beginning with the Java language itself. Many of the features of C and C++ that are detrimental to program reliability, such as pointers and automatic type conversion, are avoided in Java. The Java compiler provides several levels of additional checks to identify type mismatches and other inconsistencies. The Java runtime system duplicates many of the checks performed by the compiler and performs additional checks to verify that the executable bytecodes form a valid Java program.

Java is Dynamic

The Java programming language was designed to adapt to an evolving environment. New methods and properties can be added freely in a class without affecting their clients. Also, Java is able to load classes as needed at runtime. As an example, you have a class called 'Square'. This class has a property to indicate the color of the square, and a method to calculate the area of the square. You can add a new property to the 'Square' class to indicate the length and width of the square, and a new method to calculate the perimeter of the square, and the original client program that uses the 'Square' class remains the same.

4.1.2 Disadvantages of JAVA

Some organizations only allow software installed by the administrators. As a result, some users can only view applets that are important enough to justify contacting the administrator to request installation of the Java plug-in. As with any client-side scripting, security restrictions may make it difficult or even impossible for an untrusted applet to achieve the desired goals. Some applets require a specific JRE. If an applet requires a newer JRE than available on the system, or a specific JRE, the user running it the first time will need to wait for the large JRE download to complete. Java automatic installation or update may fail if a proxy server is used to access the web. This makes applets with specific requirements impossible to run unless Java is manually updated. The Java automatic updater that is part of a Java installation also may be complex to configure if it must work through a proxy.

4.2 NETBEANS

The NetBeans IDE is written in Java and runs everywhere where a JVM is installed, including Windows, Mac OS, Linux, and Solaris. A JDK is required for Java development functionality, but is not required for development in other programming languages. The NetBeans Platform allows applications to be developed from a set of modular software components called modules.

The NetBeans Platform is a reusable framework for simplifying the development of Java Swing desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required. This platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application.

Among the features of the platform are:

- User interface management (e.g. menus and toolbars)
- User settings management
- Storage management (saving and loading any kind of data)
- Window management
- Wizard framework (supports step-by-step dialogs)
- NetBeans Visual Library

4.2.1 Advantages of Netbeans:

1. GUI :The major requirement of today's developers is to have a good User Interface for their users. They can provide whatever functionality they need but it the GUI that lets the user better knows the existence of that particular functionality and its easier for them to click and select than type something on a black boring screen. Thus, today's developers need IDE's such as netbeans that develop ready made windows forms with all the required buttons, labels, text boxes are tailor made for many applications.

2. Database Integration: Database based program developers know how hard it is to interface your back-end database to your front-end program. This is where netbeans packs the punch by providing you a CRUD(create, Read, Update, Delete) application shell.

CHAPTER 5

System Snapshots

5.1 Startup Window showcasing Packet sniffing before starting the process.

Select Network Adapter: Realtek RTL8139 Family Fast Ethernet Adapter (Microsoft's Packet Scheduler) ▼ Start

Device ID: Device\NPF_{CC54DEA4-A153-4452-B0C6-11D2F557E876}

Source IP	Source MAC	Dest IP	Dest MAC
-----------	------------	---------	----------

Above Screenshot shows the status of the Packet sniffer before we have started the process.

5.2 Window Showcasing packets information displayed by packet Sniffer.

The screenshot shows a packet sniffer application window. At the top, there is a blue header bar. Below it, the 'Select Network Adapter:' dropdown menu is set to 'Realtek RTL8139 Family Fast Ethernet Adapter'. To the right of this menu is a 'Stop' button. Below the adapter selection, the 'Device ID:' is displayed as 'Device\NPF_{9043803D-9BA0-4A06-AB0F-A1CCB90960ED}'. The main area of the window contains three tabs: 'Packets', 'Flagged Hosts', and 'Breach of Flag'. The 'Packets' tab is active, displaying a table with four columns: 'Source IP', 'Source MAC', 'Dest IP', and 'Dest MAC'. The table contains 16 rows of captured packet data.

Source IP	Source MAC	Dest IP	Dest MAC
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.73.12	00:0b:ac:f6:14:05	172.16.8.173	00:16:ec:e6:53:44
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05
172.16.8.173	00:16:ec:e6:53:44	172.16.73.12	00:0b:ac:f6:14:05

In above Snapshot packet sniffer has started to execute and Packet sniffer Displays following attributes related to the packets as output.

IP Addresses : Internet Protocol address is the address of a device attached to an IP network (TCP/IP network). Every client, server and network device is assigned an IP address, and every IP packet traversing an IP network contains a source IP address and a destination IP address. Every IP address that is exposed to the public Internet is unique. In contrast, IP addresses within a local network use the same private addresses; thus, a user's computer in company A can have the same address as a user

in company B and thousands of other companies. However, private IP addresses are not reachable from the outside world. Packet Sniffer displays the IP addresses of source and destination of the terminals that are communicating through interconnection. This gives user an opportunity to find out which terminals he is connected to and determine whether he wants to continue with the connection or terminate it.

MAC Address: In computer networking, a Media Access Control address is a unique identifier assigned to most network adapters or network interface cards by the manufacturer for identification, and used in the Media Access Control protocol sub-layer. If assigned by the manufacturer, a MAC address usually encodes the manufacturer's registered identification number. It may also be known as an Ethernet Hardware Address , hardware address, adapter address, or physical address.

Mac Addresses displayed by the packet Sniffer helps in Identifying whether communication between two terminals is safe from spoofing. Once you know MAC Address of both the terminals it becomes easier to track hacker try to attack your system through IP spoofing .

5.3 Window Showcasing TCP/IP information of packets being transferred.

```
Source Host Name: 172.16.73.12
Destination Host Name: ANIL-DF7EE324FC
ACK: true
ACK No.: 961584611
Dest. Port: 2996
FIN flag: false
TCP Option: null
PSH flag: false
RST flag: false
RSV flag1: false
RSV2 flag: false
Sequence: 1768851150
Src. Port: 3128
SYN flag: false
URG flag: false
```

This window shows a output of the packet sniffer after one clicks on any of the information about packets being transferred. Which generates a event and further displays the information about that packet in terms of acknowledgement no. and status of different flags associated which are following.

ACK : it is a packet message used in the Transmission Control Protocol to acknowledge receipt of a packet. if true then packet received if false then packet not received yet. If you run a packet sniffer while transferring data using the TCP, you will notice that, in most cases, for every packet you send or receive, an ACKnowledgement follows. So if you received a packet from a remote host, then your workstation will most probably send one back with the ACK field set to "1". In some cases where the sender requires one ACKnowledgement for every 3 packets sent, the receiving end will send the ACK expected once (the 3rd sequential packet is received). This is also called Windowing.

SYN Flag : SYN is initially sent while establishing three way handshake between two hosts. During the three way handshake we can count a total of two SYN Flags being transmitted one by each host. As files are received and sent we will see more SYN flags being sent.

FIN Flag : This flag is used to tear down the virtual connection established using previous flag which is syn or synchronization flag. FIN flag always appear when last Packets are exchanged between a connection.

PSH Flag : The Push flag exists to ensure that the data is given the priority that it deserves and is processed at the sending or receiving end. This particular flag is used quite frequently at the beginning and end of a data transfer, affecting the way the data is handled at both ends. When a host sends its data, it is temporarily queued in the TCP buffer, a special area in the memory, until the segment has reached a certain size and is then sent to the receiver. This design guarantees that the data transfer is as efficient as possible, without wasting time and bandwidth by creating multiple segments, but combining them into one or more larger ones. the Push flag is usually set on the last segment of a file to prevent buffer deadlocks. It is also seen when used to send HTTP or other types of requests through a proxy - ensuring the request is handled appropriately and effectively.

RST flag: The reset flag is used when a segment arrives that is not intended for the current connection. In other words, if you were to send a packet to a host in order to establish a connection, and there was no such service waiting to answer at the remote host, then the host would automatically reject your request and then send you a reply with the RST flag set. This indicates that the remote host has reset the connection.

URG flag: The first flag is the Urgent Pointer flag. This flag is used to identify incoming data as 'urgent'. Such incoming segments do not have to wait until the previous segments are consumed by the receiving end but are sent directly and processed immediately.

An Urgent Pointer could be used during a stream of data transfer where a host is sending data to an application running on a remote machine. If a problem appears, the host machine needs to abort the data transfer and stop the data processing on the other end. Under normal circumstances, the abort signal will be sent and queued at the remote machine until all previously sent data is processed, however, in this case, we need the abort signal to be processed immediately. By setting the abort signal's segment Urgent Pointer flag to '1', the remote machine will not wait till all queued data is processed and then execute the abort. Instead, it will give that specific segment priority, processing it immediately and stopping all further data processing.

SRC Port and DEST Port: The SrcPort and DstPort fields identify the source and destination ports, respectively. These two fields plus the source and destination IP addresses, combine to uniquely identify each TCP connection.

Sequence Number: The sequence number identifies the byte in the stream of data from the sending TCP to the receiving TCP that the first byte of data in this segment represents.

ACK Number : The Acknowledgement number field contains the next sequence number that the sender of the acknowledgement expects to receive. This is therefore the sequence number plus 1 of the last successfully received byte of data. This field is valid only if the ACK flag is on. Once a connection is established the Ack flag is always on.

5.4 Snapshot showcasing option bar to select network Adaptors.

The screenshot displays a software interface for network configuration. At the top, there is a blue header bar with a small icon on the left. Below the header, the 'Select Network Adapter:' label is followed by a dropdown menu showing 'Realtek RTL8139 Family Fast Ethernet Adapter (Microsoft's Packet Scheduler)' with a downward arrow. To the right of the dropdown is a 'Start' button. Below this, the 'Device ID:' label is followed by a text box containing the same adapter name and its device ID: 'DeviceID_{CC34DC4F-A153-4932-B0C0-11D2F337E070}'. Below the device ID, there are three tabs: 'Packets', 'Flagged Hosts', and 'Breach of Flag'. The 'Flagged Hosts' tab is currently selected. Under this tab, the 'Host Name:' label is followed by a text box containing the IP address '172.16.8.203'. Below the text box is a large, empty rectangular area. In the bottom left corner of the interface, there is an 'Unblock all' button.

Above Snapshot shows option of choosing network adaptors in case we have multiple Adaptors. This also shows how we can flag the unwanted host using port blocker in the framework, by inputting the IP of the unwanted connection to the port blocker.

CHAPTER 6

CONCLUSION

The framework for Enhancing Information Security protects the system from various threats over network. It helps keeping intact the information resources of a individual or a firm against various vulnerabilities and threats. It acts as a counter measure to ever increasing security threats.

This project will monitor all the packets being transferred over the network and make sure they are safe for transmission.

- Inspects each individual “packet” of data as it arrives at either side of the firewall.
- Inbound to or outbound from your computer.
- Determines whether it should be allowed to pass through or not.
- Using IP blocker we can block unwanted traffic through our Network.

The Problems faced during the construction

- Since Java Does not allow accessing operating system API so finding a way to do so was really time consuming.
- Configuring framework for different internet connections.

CHAPTER 7

FUTURE WORK

This analysis can be extended to the level of granularity necessary to move directly into creating a program that will effectively model a firewall in a simple network. It could also be extended to model a more complex firewall and or proxy server. we can also use knowledge gained through this project for developing a framework that is applicable for more number of internet browser and across different platforms.

We would like to use domain knowledge gained through this project to make a better Automated firewall that runs across spectrum of browsers and platforms.

BIBLIOGRAPHY

1. Bennett, S., Skelton, J., and Lunn, K. *Schaum's Outline of UML*. McGrawHill 1997.
2. Noonan, Wes, Ido Dubrawsky *Firewall Fundamentals* By, Pearson Education India 2008
3. NIIT , *Information Security: An Overview* By Prentice hall of India 2006
4. www.wikipedia.com
5. www.google.co.in
6. www.firewallguide.com
7. www.wisegeek.com
8. www.ipsec-tools.sourceforge.net

CHAPTER 8

APPENDIX

SOURCE CODE

```
package packetsniffer;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import jpcap.*;
import jpcap.packet.*;
import java.util.*;
import javax.swing.JOptionPane;
import javax.swing.table.*;

public class frmMain extends javax.swing.JFrame {
    NetworkInterface[] devices=JpcapCaptor.getDeviceList();
    packetReader pr;
    timer tr;
    Vector tcpPackets=new Vector();
    float maxBandwidth=256;
    String appicationPath="c:\\jvproj\\PacketSniffer\\IPSecCmd.exe";
    int totSize=0;
    int prevSize=0;
    int secondsElapsed=0;

    /** Creates new form frmMain */
    public frmMain() {
        initComponents();
    }
}
```



```

        getNetworkAdapters();
        lblWarning.setVisible(false);
    }

    void getNetworkAdapters(){
        //adding each device to the list
        for(int i=0;i<devices.length;i++){
            cmbAdapters.addItem(devices[i].description);
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc="Generated Code"> //GEN-

```

BEGIN: initComponents

```

private void initComponents() {

    jLabel1 = new javax.swing.JLabel();
    cmbAdapters = new javax.swing.JComboBox();
    jLabel2 = new javax.swing.JLabel();
    lblDeviceName = new javax.swing.JLabel();
    lstPackets = new java.awt.List();
    cmdStart2 = new javax.swing.JToggleButton();
    jTabbedPane1 = new javax.swing.JTabbedPane();
    jPanel1 = new javax.swing.JPanel();
    jScrollPane1 = new javax.swing.JScrollPane();
    tblTCP = new javax.swing.JTable();
    jLabel4 = new javax.swing.JLabel();

```



```

txtMaxBandwidth = new javax.swing.JTextField();
lblUsage = new javax.swing.JLabel();
lblTime = new javax.swing.JLabel();
lblWarning = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jLabel1.setText("Select Network Adapter:");

cmbAdapters.addItemListener(new java.awt.event.ItemListener() {
    public void itemStateChanged(java.awt.event.ItemEvent evt) {
        cmbAdaptersItemStateChanged(evt);
    }
});

jLabel2.setText("Device ID:");

cmdStart2.setText("Start");
cmdStart2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cmdStart2ActionPerformed(evt);
    }
});

tblTCP.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {

    },
    new String [] {
        "Source IP", "Source MAC", "Dest IP", "Dest MAC", "Protocol"
    }
) {
    boolean[] canEdit = new boolean [] {

```



```

        false, false, false, false, false
    };

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return canEdit [columnIndex];
    }
});

tblTCP.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        tblTCPMouseClicked(evt);
    }
});

tblTCP.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyPressed(java.awt.event.KeyEvent evt) {
        tblTCPKeyPressed(evt);
    }
});

jScrollPane1.setViewportViewView(tblTCP);

javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE,
694, Short.MAX_VALUE)
);
jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()

```



```
        .addComponent(jScrollPane1,  
        javax.swing.GroupLayout.PREFERRED_SIZE, 254,  
        javax.swing.GroupLayout.PREFERRED_SIZE)  
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,  
        Short.MAX_VALUE))  
    );
```

```
jTabbedPane1.addTab("Packets", jPanel1);
```

```
jLabel4.setText("Max. Bandwidth:");
```

```
txtMaxBandwidth.setText("256");
```

```
lblUsage.setText("00");
```

```
lblTime.setText("00");
```

```
lblWarning.setForeground(new java.awt.Color(255, 0, 0));
```

```
lblWarning.setText("Warning: Conjestion!!");
```

```
javax.swing.GroupLayout layout = new  
javax.swing.GroupLayout(getContentPane());  
getContentPane().setLayout(layout);  
layout.setHorizontalGroup(  
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,  
            layout.createSequentialGroup()  
                .addContainerGap()
```



```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
```

```
    .addComponent(jTabbedPane1,
```

```
        javax.swing.GroupLayout.Alignment.LEADING,
```

```
        javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
```

```
    .addComponent(lstPackets,
```

```
        javax.swing.GroupLayout.Alignment.LEADING,
```

```
        javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
```

```
    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
```

```
        layout.createSequentialGroup())
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
    .addComponent(jLabel1)
```

```
    .addGroup(layout.createSequentialGroup())
```

```
        .addGap(7, 7, 7)
```

```
        .addComponent(jLabel2)))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
    .addGroup(layout.createSequentialGroup())
```

```
        .addComponent(lblDeviceName,
```

```
        javax.swing.GroupLayout.DEFAULT_SIZE, 525, Short.MAX_VALUE)
```

```
        .addGap(50, 50, 50))
```

```
    .addGroup(layout.createSequentialGroup())
```

```
        .addComponent(cmbAdapters,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE, 469,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE)
```



```

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(cmdStart2))))
        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
            .addComponent(jLabel4)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(txtMaxBandwidth,

```

```

        javax.swing.GroupLayout.PREFERRED_SIZE, 57,
        javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(18, 18, 18)
            .addComponent(lblUsage)
            .addGap(138, 138, 138)
            .addComponent(lblTime)
            .addGap(99, 99, 99)
            .addComponent(lblWarning)))
        .addContainerGap()
    );
    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE,
694, Short.MAX_VALUE)
    );
    jPanel1Layout.setVerticalGroup(

```



```

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 254,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
);

```

```

jTabbedPane1.addTab("Packets", jPanel1);

```

```

jLabel4.setText("Max. Bandwidth:");

```

```

txtMaxBandwidth.setText("256");

```

```

lblUsage.setText("00");

```

```

lblTime.setText("00");

```

```

lblWarning.setForeground(new java.awt.Color(255, 0, 0));

```

```

lblWarning.setText("Warning: Conjestion!!");

```

```

javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

```



```

        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
        .addContainerGap()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

        .addComponent(jTabbedPane1,

javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
        .addComponent(lstPackets,

javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()

        layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
        .addContainerGap()

        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

        .addComponent(jLabel1)
        .addComponent(cmbAdapters,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addComponent(cmdStart2))

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

```



```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
G)
```

```
    .addComponent(jLabel2)
```

```
    .addComponent(lblDeviceName,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE, 23,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
    .addComponent(jTabbedPane1,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE, 284,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addGap(2, 2, 2)
```

```
    .addComponent(lstPackets, javax.swing.GroupLayout.PREFERRED_SIZE,  
144, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
NE)
```

```
    .addComponent(jLabel4)
```

```
    .addComponent(txtMaxBandwidth,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
        javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
        javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
    .addComponent(lblUsage)
```

```
    .addComponent(lblTime)
```

```
    .addComponent(lblWarning))
```



```

        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))

    );

    pack();
} // </editor-fold> // GEN-END: initComponents

jScrollPane1.setViewportView(tblTCP);

javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE,
694, Short.MAX_VALUE)
    );
jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 254,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

```



```

jTabbedPane1.addTab("Packets", jPanel1);

jLabel4.setText("Max. Bandwidth:");

txtMaxBandwidth.setText("256");

lblUsage.setText("00");

// TODO: add your handling code here

lblTime.setText("00");

lblWarning.setForeground(new java.awt.Color(255, 0, 0));
lblWarning.setText("Warning: Conjestion!!");


javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
            layout.createSequentialGroup()
                .addContainerGap()

                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                        .addComponent(jTabbedPane1,
                            javax.swing.GroupLayout.Alignment.LEADING,
                            javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
                        .addComponent(lstPackets,
                            javax.swing.GroupLayout.Alignment.LEADING,
                            javax.swing.GroupLayout.DEFAULT_SIZE, 699, Short.MAX_VALUE)
                    )
                )
            )
        )
    );

```



```

        .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()

```

```

private void cmbAdaptersItemStateChanged(java.awt.event.ItemEvent evt) {GEN-
FIRST:event_cmbAdaptersItemStateChanged
    // TODO add your handling code here:
    lblDeviceName.setText(devices[cmbAdapters.getSelectedIndex()].name);
}GEN-LAST:event_cmbAdaptersItemStateChanged

```

```

private void cmdStart2ActionPerformed(java.awt.event.ActionEvent evt) {GEN-
FIRST:event_cmdStart2ActionPerformed
// TODO add your handling code here:
    if(cmdStart2.getText().equals("Start")){
        try{
            maxBandwidth=Float.parseFloat(txtMaxBandwidth.getText());
        }
        catch(Exception ex){
            JOptionPane.showMessageDialog(this, "Invalid max. bandwidth entered!");
            txtMaxBandwidth.requestFocus();
            return;
        }
        pr=new packetReader();
        pr.start();
        tr=new timer();
        tr.start();
        cmdStart2.setText("Stop");
        cmbAdapters.setEnabled(false);
    }
    else{
        pr.stop();
    }
}

```



```

        cmdStart2.setText("Start");
        cmbAdapters.setEnabled(true);
    }    // TODO add your handling code here:
} //GEN-LAST:event_cmdStart2ActionPerformed

private void tblTCPKeyPressed(java.awt.event.KeyEvent evt) { //GEN-FIRST:event_tblTCPKeyPressed
    // TODO add your handling code here:
} //GEN-LAST:event_tblTCPKeyPressed

private void tblTCPMouseClicked(java.awt.event.MouseEvent evt) { //GEN-FIRST:event_tblTCPMouseClicked
    // TODO add your handling code here:
    showTCPInfo();
} //GEN-LAST:event_tblTCPMouseClicked

private void cmdStart2ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_cmdStart2ActionPerformed
    // TODO add your handling code here:
    if(cmdStart2.getText().equals("Start")){
        try{
            maxBandwidth=Float.parseFloat(txtMaxBandwidth.getText());
        }
        catch(Exception ex){
            JOptionPane.showMessageDialog(this, "Invalid max. bandwidth entered!");
            txtMaxBandwidth.requestFocus();
            return;
        }
        pr=new packetReader();
        pr.start();
        tr=new timer();
        tr.start();
        cmdStart2.setText("Stop");
    }
}

```



```

        cmbAdapters.setEnabled(false);
    }
    else{
        pr.stop();
        cmdStart2.setText("Start");
        cmbAdapters.setEnabled(true);
    }    // TODO add your handling code here:
} //GEN-LAST:event_cmdStart2ActionPerformed

```

```

void showTCPInfo(){
    if(tblTCP.getSelectedRow() != -1){
        TCPPacket tcp=(TCPPacket)tcpPackets.get(tblTCP.getSelectedRow());
        lstPackets.removeAll();
        lstPackets.add("Source Host Name: " + tcp.src_ip.getHostName());
        lstPackets.add("Destination Host Name: " + tcp.dst_ip.getHostName());
        lstPackets.add("ACK: " + tcp.ack);
        lstPackets.add("ACK No.: " + tcp.ack_num);
        lstPackets.add("Dest. Port: " + tcp.dst_port);
        lstPackets.add("FIN flag: " + tcp.fin);
        lstPackets.add("TCP Option: " + tcp.option);
        lstPackets.add("PSH flag: " + tcp.psh);
        lstPackets.add("RST flag: " + tcp.rst);
        lstPackets.add("RSV flag1: " + tcp.rsv1);
        lstPackets.add("RSV2 flag: " + tcp.rsv2);
        lstPackets.add("Sequence: " + tcp.sequence);
        lstPackets.add("Src. Port: " + tcp.src_port);
        lstPackets.add("SYN flag: " + tcp.syn);
        lstPackets.add("URG flag: " + tcp.urg);
        lstPackets.add("Urgent Pointer: " + tcp.urgent_pointer);
        lstPackets.add("Window Size: " + tcp.window);
        lstPackets.add("Data: " + new String(tcp.data));
    }
}

```



```

    }
}
/**
 * @param args the command line arguments
 */

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new frmMain().setVisible(true);
        }
    });
}

private void cmdStart2ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_cmdStart2ActionPerformed
// TODO add your handling code here:
    if(cmdStart2.getText().equals("Start")){
        try{
            maxBandwidth=Float.parseFloat(txtMaxBandwidth.getText());
        }
        catch(Exception ex){
            JOptionPane.showMessageDialog(this, "Invalid max. bandwidth entered!");
            txtMaxBandwidth.requestFocus();
            return;
        }
        pr=new packetReader();
        pr.start();
        tr=new timer();
        tr.start();
        cmdStart2.setText("Stop");
        cmbAdapters.setEnabled(false);
    }
}

```



```

else{
    pr.stop();
    cmdStart2.setText("Start");
    cmbAdapters.setEnabled(true);
}    // TODO add your handling code here:
} //GEN-LAST:event_cmdStart2ActionPerformed

```

```

class timer extends Thread{
    @Override
    public void run(){
        try{
            while(true){
                int curSize=totSize-prevSize;

                float inKB=0;
                if(curSize>0){
                    inKB=curSize/1000;
                }
                if(inKB>maxBandwidth)
                    lblWarning.setVisible(true);
                else
                    lblWarning.setVisible(false);
                lblUsage.setText(inKB + " Kb/s");
                if(prevSize!=totSize)
                    prevSize=totSize;
                this.sleep(1000);
                secondsElapsed++;
                lblTime.setText("" + secondsElapsed);
            }
        }
    }
}

```



```

    }

    catch(Exception ex){
        System.out.println(ex.getMessage());
    }

}

//the packet capturing thread
class packetReader extends Thread{
    @Override
    public void run(){
        System.out.println("n: " + devices.length);
        try{
            JpcapCaptor
captor=JpcapCaptor.openDevice(devices[cmbAdapters.getSelectedIndex()], 65535,
false, 20);
            while(true){
                Packet p=captor.getPacket();
                if(p!=null){
                    //getting the data link layer information
                    EthernetPacket etp=(EthernetPacket)p.datalink;
                    //implementation for TCP Packets
                    if(p.toString().indexOf("TCP")!=-1){
                        TCPPacket tcp=(TCPPacket)p;
                        if(tcp!=null){
                            totSize+=tcp.data.length; //adding bytes
                            lblUsage.setText("" + totSize);
                            tcpPackets.add(tcp);
                            DefaultTableModel
model=(DefaultTableModel)tblTCP.getModel();
                            model.addRow(new Object[] {"", "", "", ""});

```



```

        //adding the source ip address

tblTCP.setValueAt(tcp.src_ip.getHostAddress(),tblTCP.getRowCount()-1, 0);
        //adding the source host name

tblTCP.setValueAt(etp.getSourceAddress(),tblTCP.getRowCount()-1, 1);
        //adding the source ip address

tblTCP.setValueAt(tcp.dst_ip.getHostAddress(),tblTCP.getRowCount()-1, 2);
        //adding the source host name

tblTCP.setValueAt(etp.getDestinationAddress(),tblTCP.getRowCount()-1, 3);
        //getting the protocol name

if(tcp.src_port==20 || tcp.dst_port==20 || tcp.src_port==21 || tcp.dst_port==21){
    tblTCP.setValueAt("FTP",tblTCP.getRowCount()-1, 4);
}
else if(tcp.src_port==80 || tcp.dst_port==80){
    tblTCP.setValueAt("HTTP",tblTCP.getRowCount()-1, 4);
}
else if(tcp.src_port==110 || tcp.dst_port==110){
    tblTCP.setValueAt("POP3",tblTCP.getRowCount()-1, 4);
}
else if(tcp.src_port==25 || tcp.dst_port==25){
    tblTCP.setValueAt("SMTP",tblTCP.getRowCount()-1, 4);
}
else if(tcp.src_port==22 || tcp.dst_port==22){
    tblTCP.setValueAt("SSH",tblTCP.getRowCount()-1, 4);
}
else if(tcp.src_port==23 || tcp.dst_port==23){
    tblTCP.setValueAt("TELNET",tblTCP.getRowCount()-1, 4);
}
}

```



```

        //checking breach of flag

        for(int i=0;i<lstHosts.getItemCount();i++){

            if(lstHosts.getItem(i).equals(tcp.dst_ip.getHostName()) ||
            lstHosts.getItem(i).equals(tcp.src_ip.getHostName())){

                lstBreach.add "[" + new Date().toString() + "] Packet to/from
                flagged host " + lstHosts.getItem(i) + " by " + tcp.dst_ip.getHostName());

            }

        }

    }

    catch(IOException ex){
        System.out.println(ex.toString());
    }

}

```

```

}

//class PacketPrinter implements PacketReceiver{
    //public void ReceivePacket(Packet packet){

        //}
    //}

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JComboBox cmbAdapters;
    private javax.swing.JToggleButton cmdStart2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;

```



```

private javax.swing.JLabel jLabel4;
private javax.swing.JPanel jPanel1;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTabbedPane jTabbedPane1;
private javax.swing.JLabel lblDeviceName;
private javax.swing.JLabel lblTime;
private javax.swing.JLabel lblUsage;
private javax.swing.JLabel lblWarning;
private java.awt.List lstPackets;
private javax.swing.JTable tblTCP;
private javax.swing.JTextField txtMaxBandwidth;
// End of variables declaration//GEN-END:variables
}

```

Sample code for main program

```

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

```

```

package packetsniffer;

```

```

public class Main {

```

```

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    // TODO code application logic here
    new frmMain().setVisible(true);
}

```