ANOMALY BASED DENIAL OF SERVICE DETECTION SYSTEM

By

NC Aamna Saeed

NC Rozina Nisa

NC Nighat Perveen

NC Rubina Shaheen

Submitted to Faculty of Computer Science, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment for the requirements of BE Degree in Computer Software Engineering

March 2008

ABSTRACT

ANOMALY BASED DENIAL OF SERVICE

DETECTION SYSTEM

Denial of Service (DoS) attack is one of the greatest network security problem faced these days. In DDoS attack the attacker uses multitude of compromised systems (zombies or bots) to exhaust or deplete the resources (CPU cycles, memory, disk space, and network bandwidth) of victim server, rendering it useless. The nature of threats posed by DoS attacks on large networks demands effective detection which could lead to its rapid obviation. Most of the detection approaches defines an attack as an abnormal and noticeable deviation of some statistic of the monitored network traffic workload. Given enough time most of the techniques detect DoS attack accurately but DoS detection in real time is a critical issue which need to be tackled before web services provided by server becomes inaccessible

Activity profiling is the approach which monitors network packet header information. ***Chi-square statistic*** is applied on parameter values extracted from packet header that determines deviation of observed frequencies from expected frequencies. Chi-square statistics uses nominal (categorical) or ordinal level data, thus instead of using mean and variance, this test uses frequencies. Along with Chi-Square analysis, algorithms for memory and CPU consumption continuously calculate and compare runtime memory and CPU values with specified thresholds. When the values exceed specified thresholds, an alarm is generated notifying that the server is under DoS threat. The main objective is to detect the DoS anomaly in runtime, with reduces false alarm rate.

Therefore **Anomaly based DoS Detection System** monitors chi-square statistic as well as keep track of resource consumption of server. This approach would circumvent large processing overheads and relieve network administrators from time consuming task of scanning network traffic.

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of any other award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose unflinching support and unstinting cooperation, a work of this magnitude would not have been possible

ACKNOWLEDGMENTS

We are eternally grateful to Almighty Allah for bestowing us with the strength and resolve to undertake and complete the project.

We gratefully recognize the continuous supervision and motivation provided to us by our Project Supervisor, Head of EE Department (MCS-NUST), Lt. Col. Mofassir-ul-Haque. No words how rich would do justice to the contribution of Lec. Ahmed Raza Cheema (IS Department). Our gratitude goes to Maj. Ather Mohsin for his guidance in making best use of software designing and engineering tools and helping us out in other related problems. Lt. Col. Naveed Sarfraz Khattak (Head of CS Department MCS-NUST) for his help regarding DoS attack generation tools. Special thanks to Mr. Kaleem and Mr. Asif (CS Dept.) for their support through every stage of our project. We are deeply thankful to the staff of EE and CS department for being very tolerant and helpful with our work. We are grateful to the unparallel support and tolerance that we received from our friends for their useful suggestions that helped us in completion of this project. We are also deeply obliged to our families for their never ending patience and support for our mental peace and to our parents for the strength that they gave us through their prayers.

A word of thanks to the Military College of Signals as it had been our foundation and has made us capable to undertake the project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations

ABBREVIATIONS	DESCRIPTION
DDoS	Distributed Denial of Service
DoS	Denial of Service
IDS	Intrusion Detection system

Chapter 1

1 Introduction

1.1 Introduction

This chapter highlights the significance of DoS detection system. Different types of DoS attack, their consequences, project scope, goals, objectives, specification and work breakdown structure have been briefly discussed.. The problem statement is defined in this chapter.

1.2 Preface

The internet was designed and developed for the best delivery of any information whether malicious or not and also for minimal processing overheads. This architecture provide unfettered network path to victims. In 2001, a quantitative estimate of worldwide DoS attack frequency was 12,000 attacks over a three week period. DoS attack was listed as the most financially expensive security incidents in CSI/FBI Computer Crime and Security 2004. [1]

Distributed DoS attacks have been seen in late June and early July of 1999.The first well documented DDoS attack was on University of Minnesota in August 1999 which floods a single computer and this system was knocked off the air for more than two days. This attack occurred in August 1999 by a DDoS attack tool Trinoo which was deployed in at least 227 systems, of which at least 114 were on Internet2.

The malicious DoS attack traffic consumes network buffers, CPU processing cycles, and link bandwidth. System performance is degraded when any of these resources form a bottleneck which lead to either slow or no response to legitimate users. [1]

According to CSI/FBI computer crime and security survey in 2006 [2] DoS attack is one of the top five reasons that cause lose of billions of dollars every year as shown in Figure 1-1.

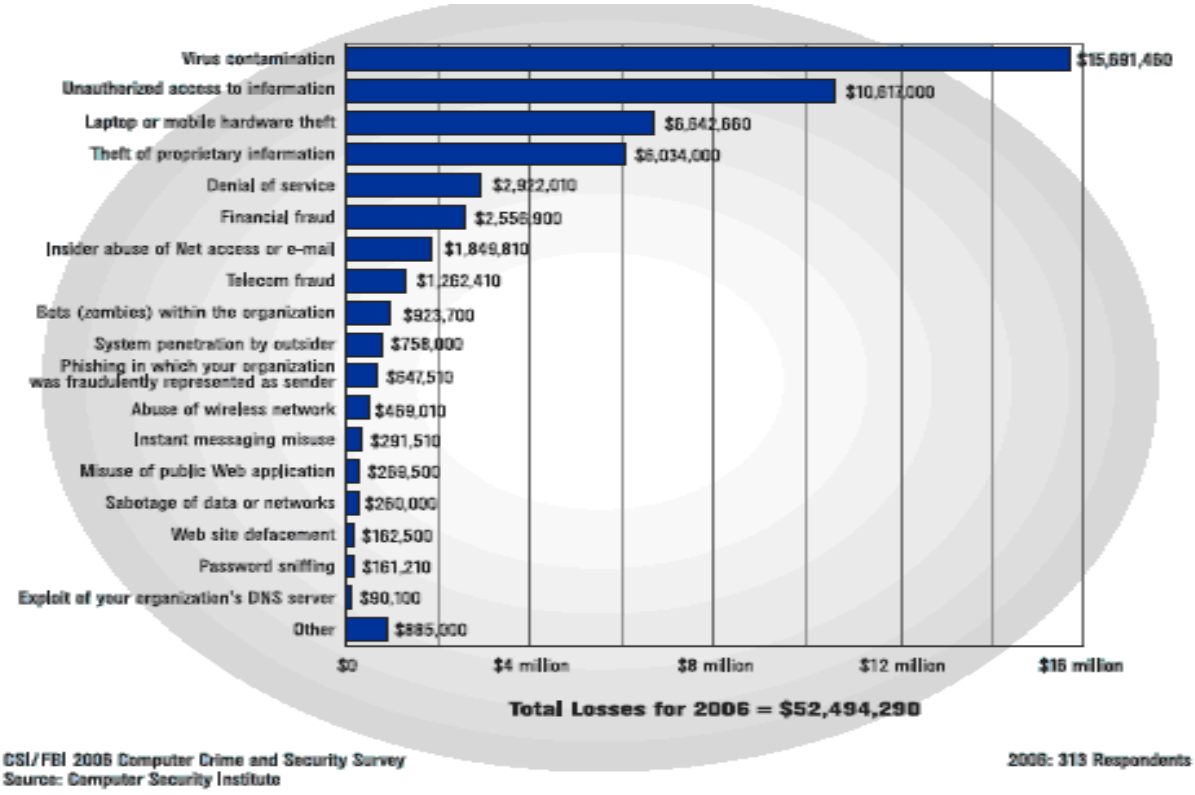


Figure 1-1: Dollar Amount Losses by Computer Crimes

This direct researchers and developers to propose techniques for detecting DoS attacks before the system crashes. Techniques that detect DoS also apply to DDoS. But real time DoS detection is a critical issue. Most of the techniques detect one type of DoS attack and do not cater for other types. This lead to development of more enhanced, efficient and reliable DoS attack detection technique. [1]

1.3 Denial-of-service (DoS) Synopsis

DoS are network attacks designed to bring the network to its knees by flooding it with useless traffic**.** Limitations in the **TCP/IP** protocols are exploited by many DoS attacks, such as the Ping of Death and Teardrop attack. The damage caused by DoS attacks can be limited by software fixes installed by system administrators. But new DoS attacks are constantly being developed by hackers. DDoS attack uses multiple compromised systems to target a single system causing **a** Denial of Service (DoS) attack. Victims of a DDoS attack include both the end targeted system and all systems maliciously used and controlled by the attacker in the distributed attack. The attacking packets come from tens or hundreds of addresses in DDoS attack, so if a defense strategy is based on monitoring packets from single address or single network, it will fail since attack comes from all over. For example, Instead of receiving a thousand gigantic Pings per second from an attacking site, the victim might receive one Ping per second from 1000 attacking sites. Almost all of DoS attacks employ TCP/IP in some non-standard ways.DoS attacks based on the exploited weakness can be generalized into two types **Vulnerability attacks** and **Flooding attacks** [1]. In Vulnerability attacks malformed packets interacts with some network protocol at the victim. Examples are Land attack, SYN Flood, Ping o'death.

An attacker sends the victim a large, occasionally continuous, amount of network traffic load in flooding attack as a result; legitimate workloads can become congested and lost at bottleneck locations. Examples include UDP flood attack, Smurf attack etc.

1.4 Problem

Over the years a number of efforts have been done and plans been executed in order to protect network systems from intrusive actions. But the major loop hole of all these efforts was that they were based on any one of the parameters like number of connections, memory or source addresses etc. Therefore a part of the problem was always left out and a complete and all in all solution could never be brought out. Now keeping that in view we have utilized multiple parameters to have an improved solution for this

multifaceted problem. The main challenges include study of traffic generators and attack tools, familiarization with Networking Terms and study of chi-square statistic

1.5 Concept Evolution

The project has been evolved in a series of steps as the search started early in the fifth semester about a year and a half back. There were many triggering events and motivating factors that led us to select this project of great caliber. In search of such a project we went to various organizations like PIEAS, NIIT, International Islamic University and CARE. After extensive research we ended up in selecting this project. We were convinced that we should do our degree project in the field of Network Security. Thus we were enthused to take up this uphill task. After rigorous meetings and interactions with DS and domain expert it as then decided to develop a new system which detects the DoS attacks. Each day leads us to new venues of exploration and learning and gradually the system idea became clear.

1.6 Project Scope

The project has a stupendous scope for the time to come. The proposed system will form a landmark in the field of Network Security. It is to develop a complete all purpose system that monitors the run time network traffic and detect the DoS attacks at run time. The system's goal is to provide a successful operational solution to present network problems caused by a variety of attack tools available today and also attempts by attackers to evade detection.

The project scope includes its colossal market value and great research potential. The product cannot only be installed in critical systems but it also invites new courses of undergraduate and even graduate level for research and development. The system places a new concept for the detection of DoS attack. This system can be deployed by any organization related to Internet E-Commerce community. It also has an important application in sensitive military areas. This smart software provides a completely automated solution to the problems caused by current as well as future growing security threats.

1.7 Objectives

The project had many multi dimensional objectives which were achieved with the course of time by Allah's Grace. These objectives are delineated as follows:

a) Immediate Objective

Immediate objective of the project is to provide a generalized solution to the problem by taking under consideration the specific system requirements. The detection method should be effective against a variety of attack tools available today and also robust against attempts by attackers to evade detection. GUI should be user friendly showing the analysis of the incoming and outgoing traffic. The detection and response techniques should be adaptable to a wide range of network environments, preferably without significant manual tuning. In order to achieve these objectives all the engineering skills, hardware knowledge and software expertise should be applied in the development of this system.

b) Future / Subsequent Objectives

The future objectives of developing the system are to install system at the large networks. After validation and thorough experimentation and deploy the system at the monitoring nodes and routers for large networks.

1.8 Project Beneficiaries

The project beneficiaries include for which the system has been developed i.e. Apart from that this project can benefit not only those who want to take up the field of network security but also the ones who are interested in Intrusion Detection Systems Development.

1.9 Project Title

Business Title: Anomaly Based DOS DetS

Technical Title: Anomaly Based Denial of Service Detection System

1.10 Project Logo

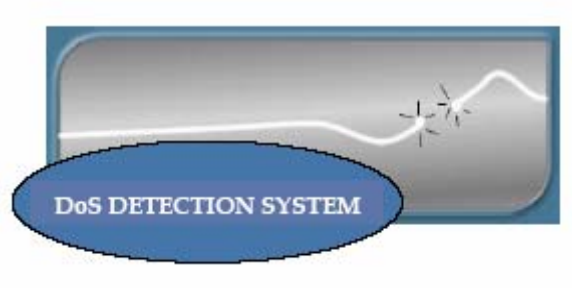


Figure 1-2: Project Logo

1.11 Project Specifications

Many approaches and techniques have been deployed to detect the DoS attacks. Two major concerns are detection **accuracy** and **real time detection**

Most of the approaches adopted, work well when given enough time but the DoS detection is a real time critical issue and it must be detected before it seriously affects the system.

Thus the project aims at developing an efficient and accurate system to detect Denial of Service attacks in real time .Thus implementing modified and enhanced chi-square statistics that determine deviation of observed frequencies from expected frequencies. In addition to this, it also incorporates continuous CPU and memory monitoring and measuring number of connections at the Server. This system also presents graphical views of Network, Transport and Application Layer revealing valuable information.

Limitations / Constraints

The software was tested and analyze thoroughly and the limits and constraints of the system are determined. In order to get accurate results, Anomaly Based DoS Detection system must be trained. Threshold must be adjusted according to normal traffic rates of the server. System must fulfill the distributed client server architecture requirements. There is need to provide interface(s) between the server and client applications.

1.12 Work Breakdown Structure

For the successful completion of the project, the project was divided into main modules and structures. It was ensured that each task being assigned was carried out appropriately and on time. Table 1-1 illustrates the requirement engineering steps followed for successful completion of project.

TABLE 1-1: Requirements Engineering

Development Process	**Description**
Requirements Elicitation	Interaction with Domain Expert and understanding basis of networking and study related security problems
Developing Problem Statement	SRS Preparation
Problem validation	Analyze various options available (i.e. platform compatibility, language) and propose the best approach to problem solution
Assignment and Planning	Assignment of tasks to the syndicate members , preparation of Gantt Charts and TimeLine charts

Literature Review

1. DoS/DDoS attacks Research Work
2. Intrusion Detection Systems Research Work
3. Existing Detection Techniques Research Work Study

Documentation

1. Preparation of System Manual
2. Detailed Thesis
3. Research Papers and Publications
4. User Manuals

1.13 Project Goals and Objectives

The objectives of the project were to design and implement effective detection algorithms in java and compare its efficiency and accuracy with other available DoS Detection techniques

1.14 Deliverables

Deliverables of the project are:

(a) EXE and code of Detection System

(b) Stored results .txt files for reusability

(c) Comparison Graphs

1.15 Conclusion

There is a need to develop an efficient DoS detection system overcoming the limitations of already developed systems. The specifications of the project have been given and the objectives, goals and scope of the project have been defined precisely.

2 Literature Review

2.1 Introduction

This chapter discusses DoS attack, describing different types of Intrusion detection systems, and highlighting pros and cons of each. Various approaches for DoS attack detection and their limitations have been discussed and overview of selected technique is presented.

2.2 Background

Internet has undergone rapid development all over the world because of widely deployed wide-area computer and communication networks and it has become indispensable in our daily lives. However due to unauthorized access, Internet has caused many security problems and financial loss. In 1986 Cliff Stoll identified the first international well-publicized security incident of the ARPANET. The first automated network security attack, referred to as the Morris worm was experienced by ARPANET in 1988.As the network capability grows faster , network security has become an important issue from both theoretical point of view as well as engineering applications.[3]

The motivation for DoS attacks is not to break into a system. Instead, it is to deny the legitimate use of the system or network to others who need its services. This will typically happen through crashing the system, deny communication between systems, bring the network or the system down or have it operate at a reduced speed which affects productivity and hang the system, which is more dangerous than crashing since there is no automatic reboot. Productivity can be disrupted indefinitely. [4]

A DoS attack can be perpetrated in following number of ways such as consumption of computational resources, such as bandwidth, disk space, or CPU time, disruption of configuration information, such as routing information, unsolicited resetting of TCP sessions and disruption of physical network components. Another way is obstructing the communication media between the intended users and the victim so that they can no longer communicate adequately.

This lead to the development of more powerful network attacks. Some of the crucial attacks include denial of service (DoS), worm, trojan horse etc each of which causes serious threats to normal business operations. A DoS attack can be defined as an explicit attempt of attackers to prevent legitimate users from gaining a normal network service. There is intensive research in this area because of serious consequences of DoS attacks. DoS attacks can be classified into different types but the most commonly known type is packet flooding attack. Attackers consume the basic and limited resources (process control blocks and the maximum allowed connections) of a victim by flooding the network with large volume of data. DoS attacks may also disrupt the normal operation of various physical components in the network such as hub, switches, routers etc. Moreover victim can be flooded with a barrage of attack packets send by multiple coordinated hosts which is usually referred to as distributed denial of service (DDoS) attacks. A reflection attack, which is a special case of DDoS attacks, hides the identity of the attackers or amplifies an attack by using compromised hosts as reflectors. Therefore damages caused by DDoS are more severe as compared to other DDoS attacks. [3]

2.3 Intrusion Detection System

Intrusion detection systems (IDS) are becoming increasingly important in helping to maintain proper network security. IDS monitors network traffic and detects network behavior pattern that matches known attack signatures. When a close match occurs, the IDS issues alarms or alerts and take actions accordingly. Firewalls and other similar boundary devices lack this degree of intelligence**.** IDS systems resemble antivirus software packages. They monitor and inspect the contents of network traffic to look for possible attacks, similar to antivirus which inspect contents of incoming files or email attachments etc.

IDSs' are classified into three main types, namely network-based, host-based, and application-based IDS, depending upon the kinds of activities, traffic, transactions, or systems they monitor. Network-based IDSs monitor an entire network with only a few nodes or devices without significant overhead and interference with network operations whereas host-based IDSs operate on hosts, defend and monitor the operating and file systems for signs of intrusion. Application-based IDSs monitor only specific applications such as database management systems, content management systems, accounting systems etc. [5]

The technique deployed can be anomaly based and signature based IDS systems. Signature based IDS operate in much the same way as virus scanners. These systems search for a known identity or signature based on specific intrusion event. It is very efficient at detecting known DoS attacks and also attack signatures keeps on updating with variation in hackers techniques. There are two problems that arise with signature based. First signatures can be easily fooled by changing the ways of attempting an attack. Signatures stored in databases give hackers ideal opportunities to gain access to network. [6]

An anomaly based IDS indicate attack by examining ongoing traffic, activity, or behavior for anomalies on networks or systems .According to underlying principle "attack behavior" differ significantly from "normal user behavior". Anomaly based IDS detects when current behavior deviates statistically from normal behavior by creating baselines of normal behavior. This makes anomaly based IDS systems capable of detecting new attacks even those for which signatures have not been defined. [5]

2.4 Detection of DDoS Attack:

Many approaches and techniques have been deployed to detect the DoS attacks. Two major concerns are the detection accuracy and real time detection [7].All techniques have the same goal, which is to reduce false positive rate and detect the anomalies accurately. Various techniques which give accurate results do not work well with the real time network traffic. Ingress/filtering, SYN cache, SYN cookie are some of the early mechanisms adopted to detect the DoS SYN Flood attacks which comprise 90% of all

DoS attacks [8].Machine Learning knowledge like ANN is also used to tackle the problem [9] [10] [11]. Some statistical approaches [7] also have been deployed to detect the intrusions. Most of the approaches work well and give optimized results when given enough time but the DDoS detection is a real time critical issue. It must be detected at the time of its occurrence or before it has caused the system to crash. Few researchers are pondering on real time detection. Self-Organizing Maps [12] is one of the approaches used but the computation overheads involved during training huge datasets prohibits its extensive use. The project presents implementation of DoS detection statistical technique, Chi-Square [13] along with the new idea of system resource monitoring.

2.5 Technique Overview:

Most of the detection approaches defines an attack as an abnormal and noticeable deviation of some statistic of the monitored network traffic workload. This makes choice of statistic critically important. The system developed is anomaly based that indicates attack by examining ongoing traffic, activity, or behavior for anomalies on network. ***Activity profiling*** is the selected approach which monitors network packet header information such as source address, port, protocol etc. [1]

Activity profiling uses ***Chi-square statistic*** that determine deviation of observed frequencies from expected frequencies. The chi-square (chi, the Greek letter pronounced "kye") statistic is a nonparametric statistical technique used to determine if a distribution of observed frequencies differs from the theoretical expected frequencies. Chi-square statistics test uses nominal (categorical) or ordinal level data, thus instead of using mean and variance, this test uses frequencies.

Chi Square statistics provide the basis for comparing the distributions involving discrete values [13]. The deviation of the distribution can be marked by using the "goodness of fit test" by comparing the observed frequencies with theoretical or expected frequencies derived under specified probabilities distributions or hypothesis. Since in network traffic, there are a lot many packets so concept of binning needs to be introduced. Binning is combining a set or range of possible values, on the basis of some criteria, and treating them as one unit. It can be done on the basis of port numbers, frequently incoming ip's

etc. Chi-Square test provides a good measure of the deviation of the observed_profile from the base_profile. The attribute taken for analysis is the source addresses and arrival time of the packet.

In DoS attack, the invader uses multitude of compromised systems (zombies or bots) to exhausts the resources (*CPU cycles, memory, disk space, network bandwidth etc*) of victim server, rendering it useless [14].The resource monitoring algorithm continuously compute the runtime CPU usage, memory consumption and number of connections at the Server machine. The outcomes are compared with the thresholds specified for the Server machine. If a considerable deviation is found in the comparison, this leads to generation of alarm thus depicting detection of attack. The algorithms are implemented by making use of .dll libraries of CPUUSAGE and java native libraries.

2.6 Conclusion

After discussing several approaches for DoS detection, activity profiling is selected which is based on analyzing packet's header information. The detection technique is based on monitoring several parameters and chi-square statistic is used for observing deviation of network traffic from normal behavior.

3 Design and Implementation of Algorithms

3.1 Introduction

In this chapter the quality parameters for determining system's accuracy have been specified. The adopted technique, system's architecture and sequence of activities in the system have been explained. This chapter gives description of Algorithms and system implementation as well.

3.2 Quality Parameters:

In order to evaluate the efficiency of the proposed approach we use two measures: the detection rate and the false alarm rate that has been defined in [13], as follows:

$$\text{Detection Rate} = \frac{TP}{TP+FN} \tag{3.1}$$

$$\text{False Alarm Rate} = \frac{FP}{TN+FP} \tag{3.2}$$

Where TP is the number of true positives (attack logs classified as attacks), TN the number of true negatives (normal logs classified as normal), and FP the number of false positives (normal logs classified as attacks) and FN the number of false negatives (attack logs classified as normal).

The most effective approach should reduce as much as possible the *False alarm rate* and at the same time increase the *Detection rate.*

For the evaluation of proposed approach, DARPA 1998 dataset are used that are widely used in order to compare intrusion detection approaches as a benchmarking dataset. Although the subsets used are randomly selected and different from subsets used in previous approaches, the results are extremely promising. [15]

3.3 Process used in all implemented algorithms:

In this thesis, we use the same procedure when testing Chi-Square and Resource algorithms.

Step 1: The packets are captured from network card

Step 2: A preprocessing is carried out in filtering process and packet header information is extracted

Step 3: The analysis is performed using Chi-Square Statistic that characterizes its behavior. For resource detection algorithms, parameters values are taken at runtime directly from the system and then analysis is carried out.

Step 4: Outputs of the algorithms are displayed continuously in the form of graphs and alarm status (on, off).

All algorithms are implemented in JAVA.

3.4 Techniques for analysis:

Activity profiling is the selected approach which monitors network packet header information such as source address, port, protocol etc. In activity profiling to avoid high-dimensionality issues, individual flows with similar characteristics are clustered. Each cluster's activity level is the summation of constituent flows. On the basis of this abstraction, an attack is indicated by an increase in activity levels among clusters, which indicate a few attacking agents increasing their attack-generation rate or increase in the overall number of distinct clusters, which can represent many distributed attacking agents (as in a DDoS). [1]

3.5 System Architecture:

The DoS detection system is deployed at the server end which provides service as a website to the clients. The clients are generally the normal users and may send malicious traffic acting as an attacker. The devised system architecture for the implemented technique is shown in Figure 3-1.

The DoS system is composed of Sniffing unit, Statistical Detector, Resource Detector and Visualization unit. Statistical Detector makes use of Chi-Square statistics. Memory Usage, CPU Consumption and Number of connections are part of Resource Detector. Visualization unit consists of Graphs and Alarms.

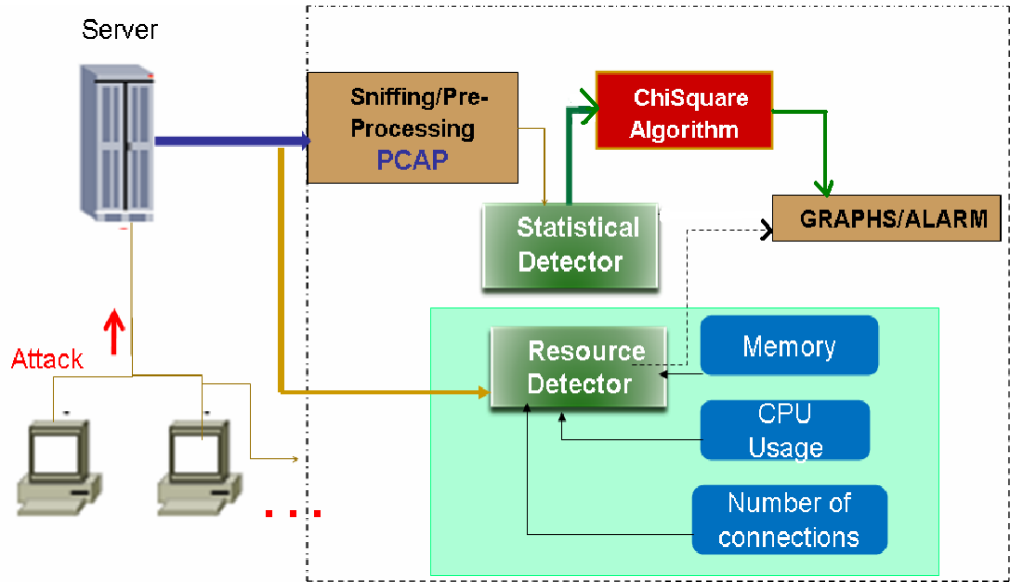


Figure 3-1: System Architecture

The system has two major detection modules. First, Statistic Detector using Chi-square algorithm and second, Resource Detector based on resource parameters algorithms to perceive the DoS attacks.

3.6 Flow Chart:

Figure 3-2 illustrates the flow of activities in Anomaly Based DoS Detection System

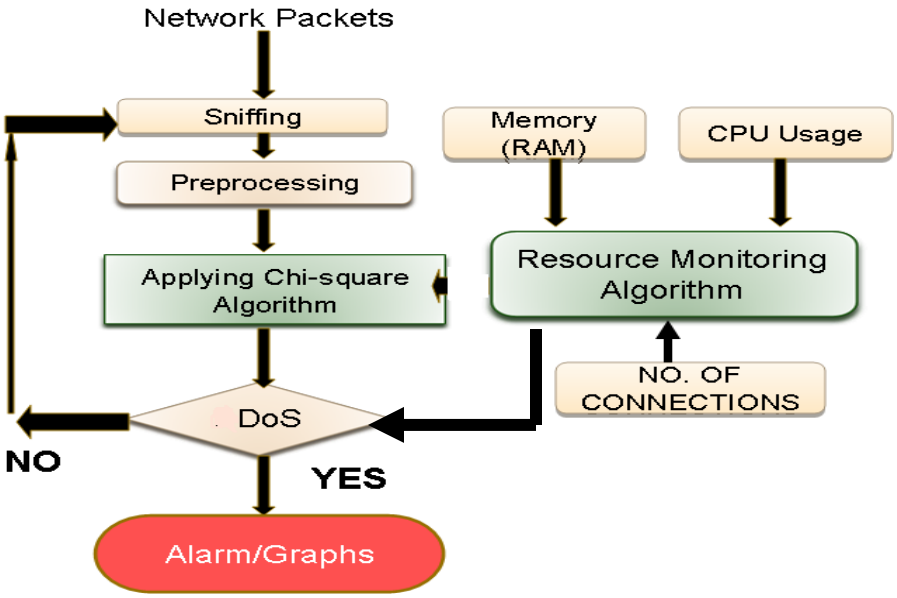


Figure 3-2: Flow Chart

Sniffing:

Sniffing refers to packet capturing

Preprocessing:

Preprocessing involves filtering the network traffic from broadcast addresses and extracting source IP address of packet. The packet details in each of the layer can be seen graphically. The continuous information is presented through line graphs while the cumulative layer information is viewed in pie charts.

Chi-Square Algorithm:

The algorithm is applied on the extracted source addresses from preprocessed network traffic and analyses the traffic distribution on the basis of frequency.

Resource based Algorithms:

Algorithms are applied for monitoring random access memory (RAM), CPU Usage and number of connections and then comparing against the threshold specified.

If DoS attack is identified by Chi-square or Resource based algorithms system continues preprocessing incoming traffic as this is real time critical system and there is need to monitor the traffic all the time. Here multiple parameters are used for decision making, enhancing accuracy and reducing false alarm rate for DoS detection.

Show Graphs:

The incoming traffic is tested with the DoS detection system .Chi-Square and free Memory graphs are displayed and CPU usage at each instant is shown.

3.7 Class Diagrams:

The class diagram is shown in Figure 3-3

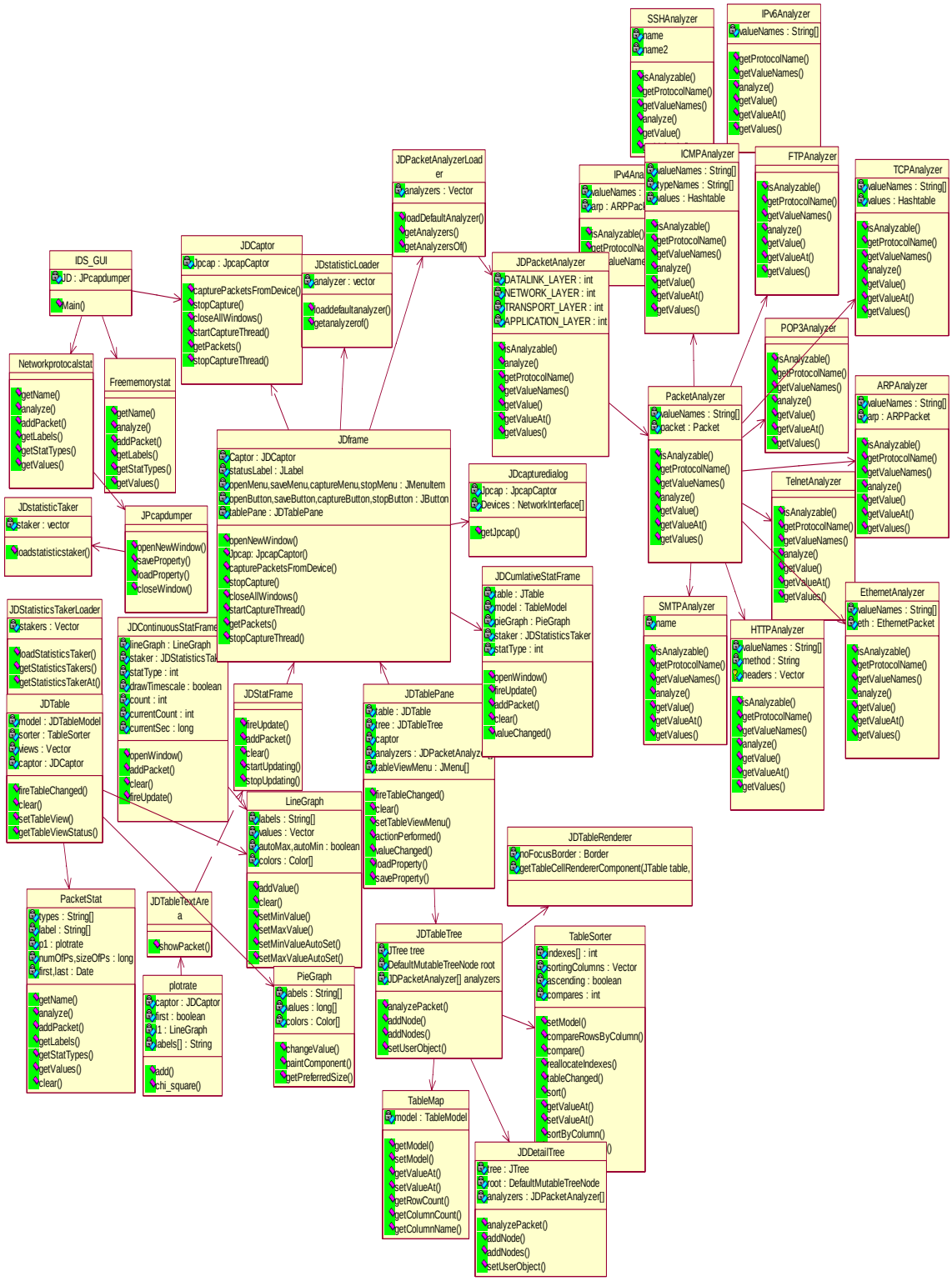


Figure 3-3: Class Diagram

3.8 Chi-Square Algorithm

The chi-square (chi, the Greek letter pronounced "kye") statistic is a nonparametric statistical technique used to determine if a distribution of observed frequencies differs from the theoretical expected frequencies. The value of the chi-square statistic is given by

$$X^2 = \sum (O-E)^2 / E \qquad 3.3$$

Where X^2 is the chi-square statistic, O is the observed frequency and E is the expected frequency.

In this approach packet arrival rate is calculated per second. Since the packet rate is being monitored every second, the DoS attack will be detected in the beginning. Early detection will aware the administrator of the DoS attack in order to take defensive measures as soon as possible.

After capturing 5 hour normal traffic at the server, the maximum packet arrival rate for normal traffic is calculated. Chi-square is calculated for every two samples each of 1 second. When a comparison is made between one sample and another, a simple rule is that the degrees of freedom equals number of samples minus one .For our data this gives (2-1) = 1.

First the null and alternative hypotheses are established.

Null Hypothesis: The observed packet rate is normal.

Alternative Hypothesis: The observed packet rate is not normal.

Step 1: For packets arriving at the server, calculate the packet rate Ro (observed rate) every second.

Step 2: Compare the observed rate Ro with the expected rate Re.

Step 3: If Ro is higher than Re, then determine the deviation of observed rate Ro from the maximum normal rate Re, using chi-square formula:

$$X^2 = (Ro\text{-}Re)^2 / Re \quad 3.4$$

If Ro is smaller than Re, then rate Re is considered normal within the range up to the maximum expected rate. In this case deviation needs not to be calculated.

Step 4: After obtaining two sample values for chi-square as in above step take the summation:

$$X^2 = \sum_{i=1}^{n} \frac{(R_o - R_e)^2}{R_e} \quad 3.5$$

Step 5: Compare it with the chi-square threshold. The chi square threshold is selected from the standard table defined for chi-square statistic while keeping level of significance .001 and degrees of freedom 1.

$$\text{Chi-square threshold} = 10.83 \quad 3.6$$

Step 6: If X^2 is lesser than to the chi-square threshold, then the rate is considered to be normal the null hypothesis is verified. On the other hand if X^2 is equal to or greater than the chi-square threshold, the null hypothesis is rejected and alternative hypothesis is accepted. The alarms are generated as an indication of a DoS attack.

The maximum expected packet rate should be determined with accuracy. A much higher expected rate may increase the false negatives (attack traffic detected as normal), decreasing the detection rate. Whereas very lower expected packet rate would increase the false positives (normal traffic detected as attack).Normal traffic of much larger time period would give a better approximation of maximum expected packet rate.

3.9 Resource Algorithm:

Every Operating system has resources (CPU Usage, runtime memory and protocol queue length) associated with it for its efficient and reliable working. A limited

sized backlog queue is associated for each network protocol. Backlog queue holds the number of connections that are made to the system. The size of the backlog queue serves as the threshold for the number of connections parameter. When the connections made to the system exceed the threshold, it means no other connections can be handled and access to the legitimate users is not provided by the system. This creates scenario for the DoS attack. Algorithm is written in Java to first get the number of connections made to the system and then compare it against the threshold. If value exceeds the threshold, it indicates DoS attack by raising alarm. Some DoS attacks consume CPU cycles causing the system to crash. The CPU consumption if remains 100% for 2 seconds, the system crashes. This serves as a threshold for the CPU usage parameter. The algorithm keeps acquiring CPU usage at runtime and comparing it against the threshold specified. As the parameter value exceeds the threshold specified attack is identified.

3.10 Implementation:

3.10.1 Sniffer Module:

The sniffer module is implemented in Java using Jpcap API. It captures the network traffic arriving at the interface .Figure 3-4 illustrates network packets captured by the sniffer and the packet header information extracted out and displayed.

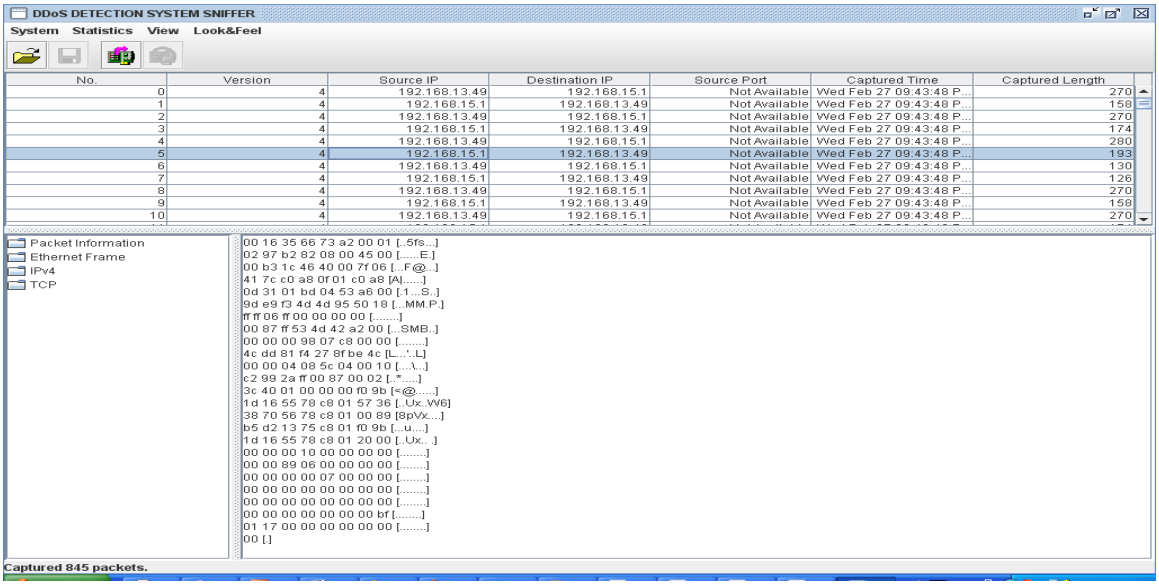


Figure 3-4: Packet Sniffer

Packet sniffer captures the packets and displays the information at each layer. A graphical view helps to monitor the nature of incoming packets in terms of Internet Model.

In Continuous view, the protocol ratios of the layer are shown with linear graphs. In Cumulative view, in addition to the protocol ratios view as pie graphs, statistics as total number of packets, percentage of packets of each protocol, total packet size and percentage of packets size are also displayed.

3.10.1.1 Network Layer:

The network layer is responsible for the source-to destination delivery of packets across multiple networks.IPv4 and IPv6 are the protocols of this layer. The graph in Figure 3-5 shows continuous(with respect to time) Network Layer protocol ratio with blue line depicting IPv4 packets, green illustrating IPv6 packets and red showing other packets.

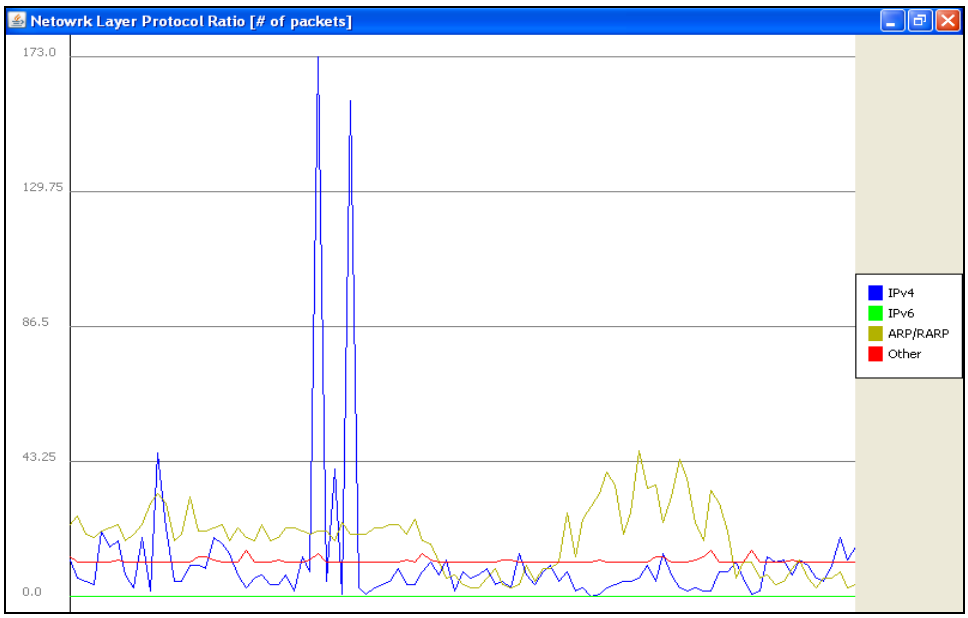


Figure 3-5: Continuous Network Layer View

Figure 3-6 shows cumulative network layer protocol ratio in the Pie graph along with statistical information of number of packets, percentage of packets, size of packets and percentage of size of packets.

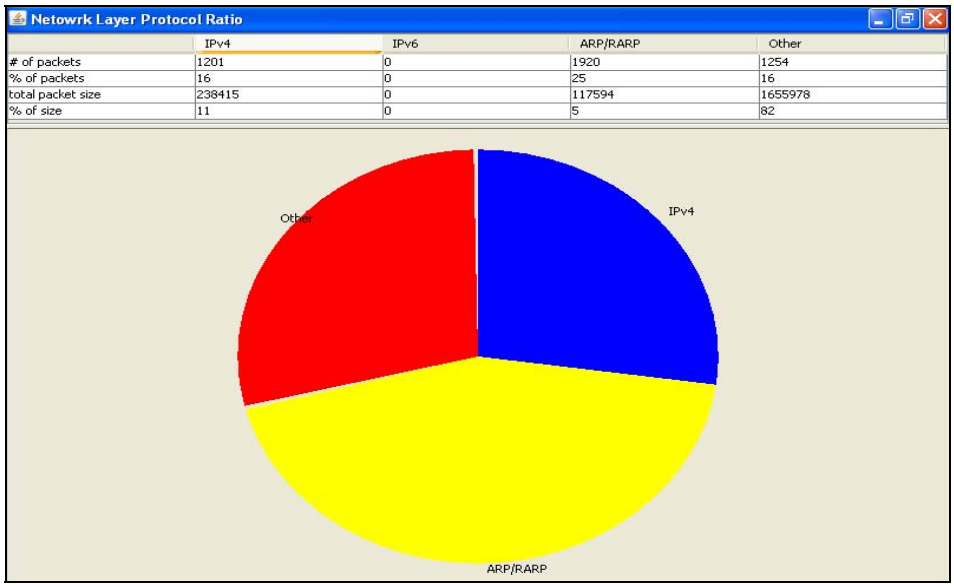


Figure 3-6: Cumulative Network Layer View

3.10.1.2 Transport Layer:

The transport layer is responsible for process to process delivery of entire message. The protocols of the layers are User Datagram protocol (UDP) and Transmission control protocol (TCP). Figure 3-7 shows continuous Transport Layer protocol ratio with blue line depicting TCP packets, green illustrating UDP packets, brown ICMP and red showing other packets.

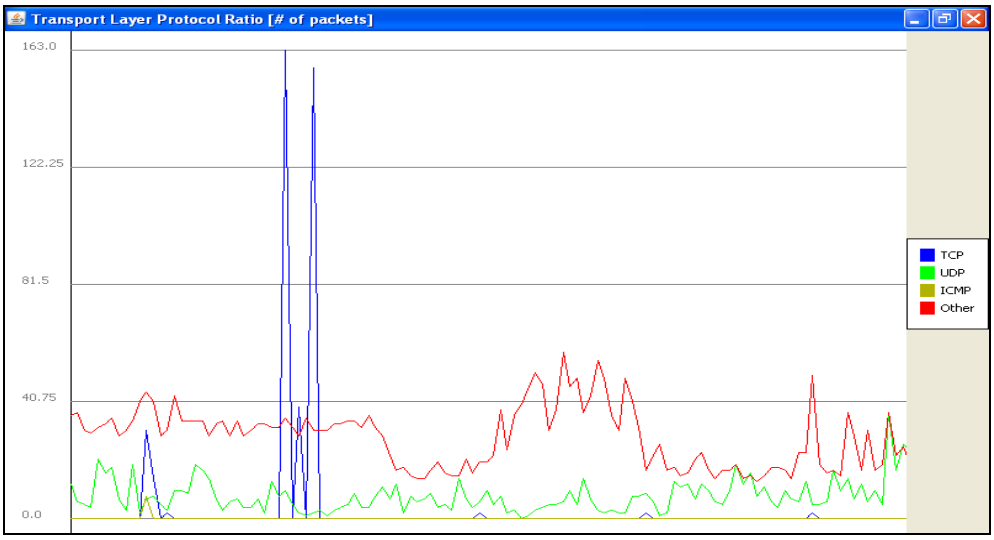


Figure 3-7: Continuous Transport Layer View

Figure 3-8 shows cumulative Transport layer protocol ratio in the Pie graph along with statistical information of number of packets, percentage of packets, size of packets and percentage of size of packets.

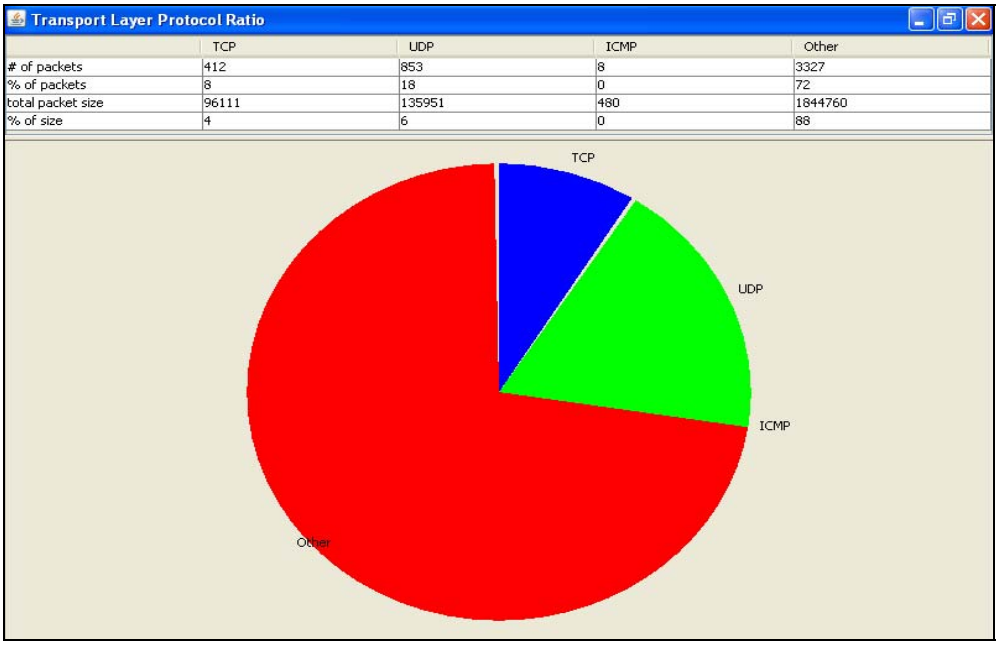


Figure 3-8: Cumulative Transport Layer View

3.10.1.3 Application Layer:

The layer which enables the user to access the network is the application layer. The protocols are HTTP, FTP, Telnet, SSH, SMTP and POP3 correspondingly depicted with blue, green brown, red, light blue, pink and orange color. Figure 3-9 shows continuous Application layer protocol ratio.

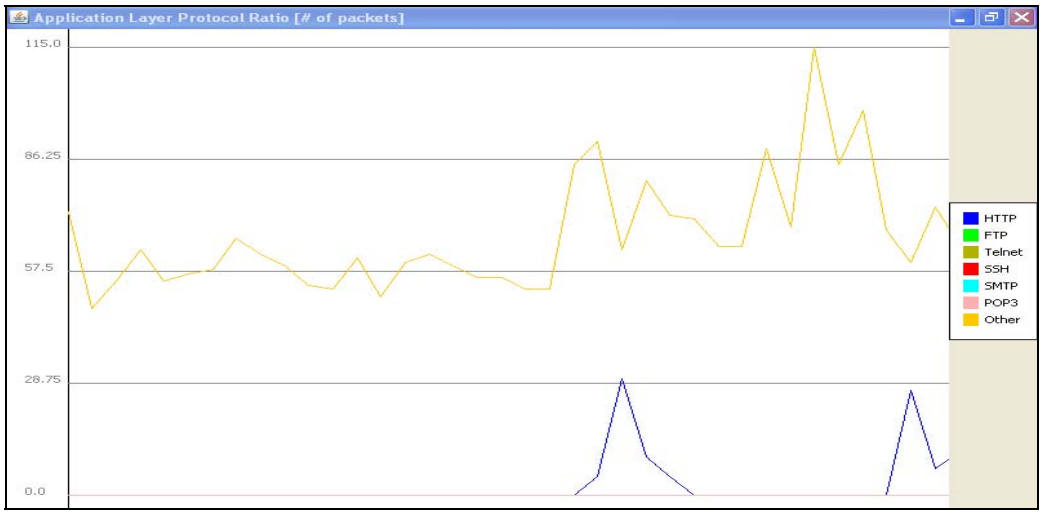
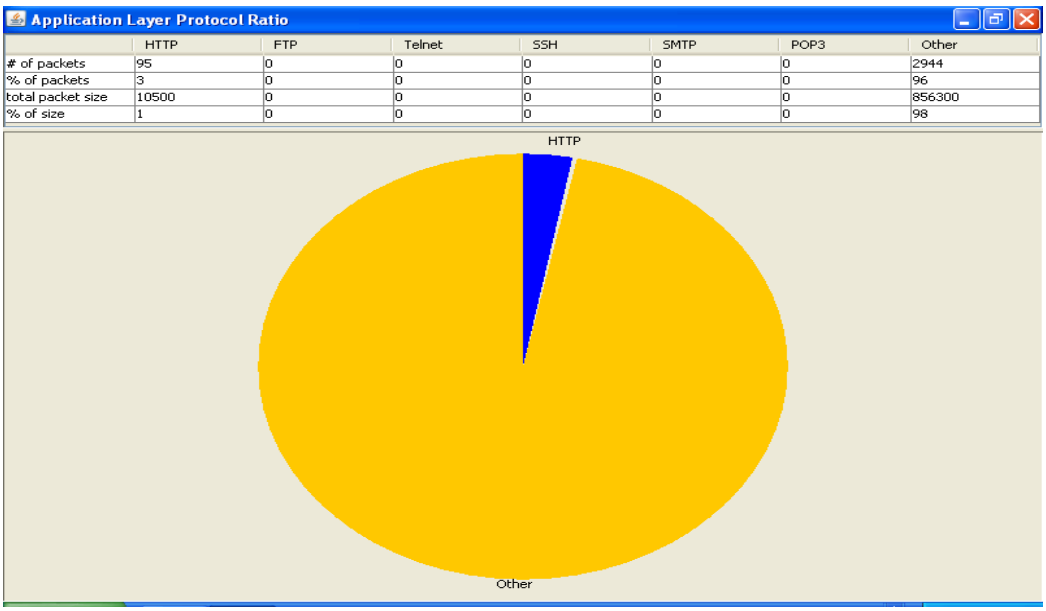


Figure 3-9: Continuous Application Layer View

Figure 3-10 illustrates cumulative Application layer protocol ratio.



	HTTP	FTP	Telnet	SSH	SMTP	POP3	Other
# of packets	95	0	0	0	0	0	2944
% of packets	3	0	0	0	0	0	96
total packet size	10500	0	0	0	0	0	856300
% of size	1	0	0	0	0	0	98

Figure 3-10: Cumulative Application layer View

3.10.2 Chi-Square Detection Module:

The module has been built in JAVA. The source IPs and frequency extracted out by the sniffer is applied as input to the Chi-Square algorithm to get two output graphs Packet count versus packet rate and Packet count versus Chi-Square values

3.10.3 Resource Detection Module:

Since there was no API found in JAVA for efficient CPU measurement a .dll file has been created in Visual C++ to calculate cup usage of the system. Windows API was used to measure the number of connections .The resource detection algorithm has been solely implemented in JAVA. The resource detector displays Pie graph illustrating runtime memory every second, a bar chart showing number of connections and a bar chart depicting CPU usage per second of time.

3.11 Conclusion

The system is implemented in JAVA. The complete system's design is provided. The developed algorithms are given. Each system module has been fully described along with the screen shots shown. Parameters for determining system's accuracy have been defined.

4 Testing and Results

4.1 Introduction

The chapter describes system testing in two modes that is, active and passive mode. Attack scenario is presented and results and graphs for normal as well as attack cases are shown. In the end the overall system accuracy in terms of detection rate and false alarm rate is determined.

4.2 Active Mode

To test the system in real time, Nemesis tool has been used for generating DoS attack. Nemesis is a command-line network packet crafting and injection utility for UNIX-like and WINDOWS systems. Nemesis is an appropriate tool for testing Network Intrusion Detection Systems, firewalls, IP stacks and a variety of other tasks. Nemesis can natively craft and inject ARP, DNS, ETHERNET, ICMP, IGMP, IP, OSPF, RIP, TCP and UDP packets. [16]

4.2.1 TCP Connection:

A **three-way handshake** is used to establish a TCP connection. Before a client attempts to connect with a server it should do passive open i-e the server first bind to a port to open it up for connections. In response to passive open, a client may initiate an active open. The active open is performed by the client sending a SYN to the server.In response, the server replies with a SYN-ACK. Finally the client sends an ACK back to the server [17].

Figure 4-1 illustrates three way hand shake protocol [18].

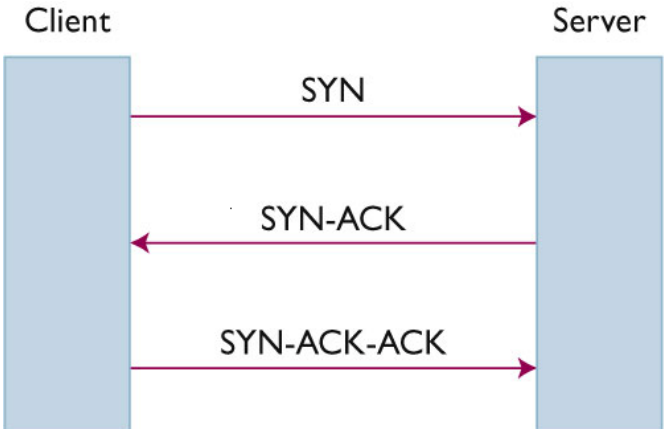


Figure 4-1: TCP's three-way handshake

4.2.2 SYN Flood Attack

The **SYN flood** attack sends TCP connections requests more rapidly than a machine can process them. It comprises 90% of DoS attacks. [19] In SYN flood attack attacker creates a random source address for each packet and set SYN flag in each packet to open a new connection to the server from the spoofed IP address. The victim (server) responds to spoofed IP address, and then waits for affirmation that never arrives .Victim's connection backlog queue fills up waiting for replies and all new connections are ignored .Legitimate users are ignored as well, and cannot access the server

4.2.3 Attack Scenario:

Nemesis1.4 tool has been used to craft TCP packets with spoofed source IPs to generate TCP Syn Flood attack. Infinite loop of 1024 crafted packets with distinct spoofed IPs have been used to inject huge number of packets to the server.

To run nemesis 1.4, Windows system requires libnetNT-1.0.2g and Winpcap-3.0.

On the server, apache 2.2.6 is running to provide service to the clients at port 80. A website can be accessed by the clients when the user is a legitimate client and the system

has not been attacked by the malicious network traffic. Under normal conditions, site is easily accessed by the client.

Under attack conditions, the resources consumption over exceeds the threshold and attack is detected. A remarkable deviation is seen in the Chi-Square and rate graph.

4.2.4 Results and Graphs:

The resource and memory graphs are illustrated below for normal and under attack configuration.

4.2.4.1 Normal Configuration

4.2.4.1.1 Packet Rate Graph:

When the system is under DoS attack, Packet Rate (no of packets per second) increases abruptly. However under ordinary situation Packet Rate is normal. This has been illustrated in Figure 4-2.

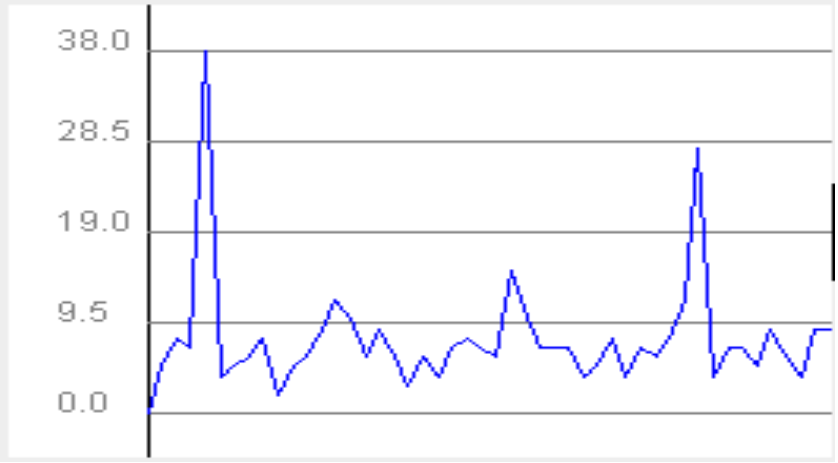


Figure 4-2: Normal Packet Rate

4.2.4.1.2 Connections Graph:

No of half open TCP connection that can be open per second is limited. In Windows NT, this threshold is defined to be 200.when system is not under DoS threat, no of connections per second are below this threshold as shown in Figure 4-3.

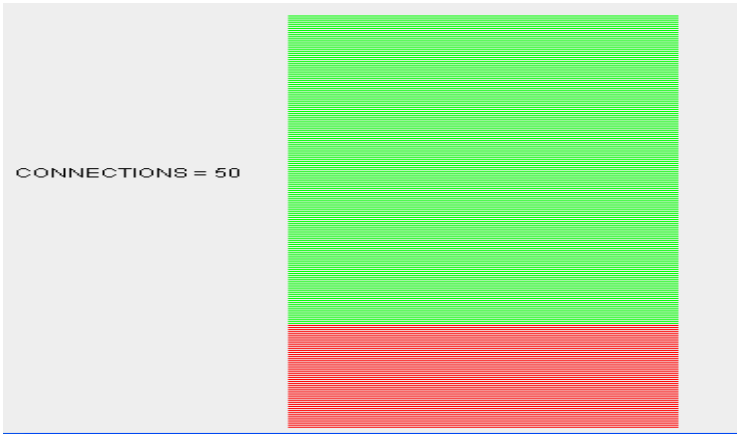


Figure 4-3: Normal TCP Connection Graph

4.2.4.1.3 Memory Graph:

Certain DoS attacks consume runtime memory of server so that server is not able to provide its services to legitimate users. Figure 4-4 shows that in normal situation run time usage does not exceed the specified limit.

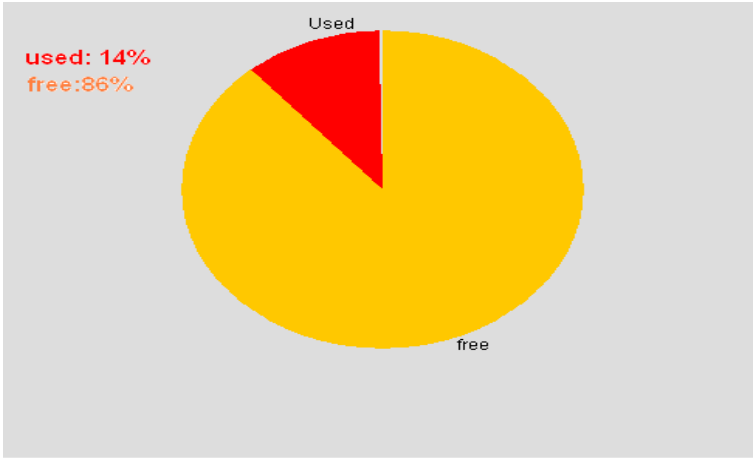


Figure 4-4: Normal Memory Usage Graph

4.2.4.1.4 CPU Consumption Graph:

There are certain DoS attacks in which crafted packets are send to server with illegal string. Server processor becomes busy in processing that string. Figure 4-5 shows that CPU usage is within limit when system is not under DoS attack.

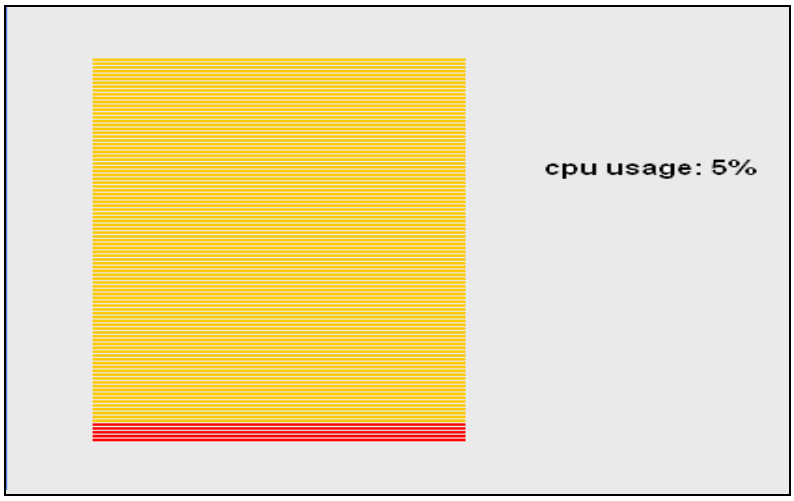


Figure 4-5: Normal CPU Usage Graph

4.2.4.2 Attack Configuration

4.2.4.2.1 Chi-Square and Rate Graph:

When the system is under DoS attack, attacker floods the server by sending large number of packets per second as shown in Figure 4-6.

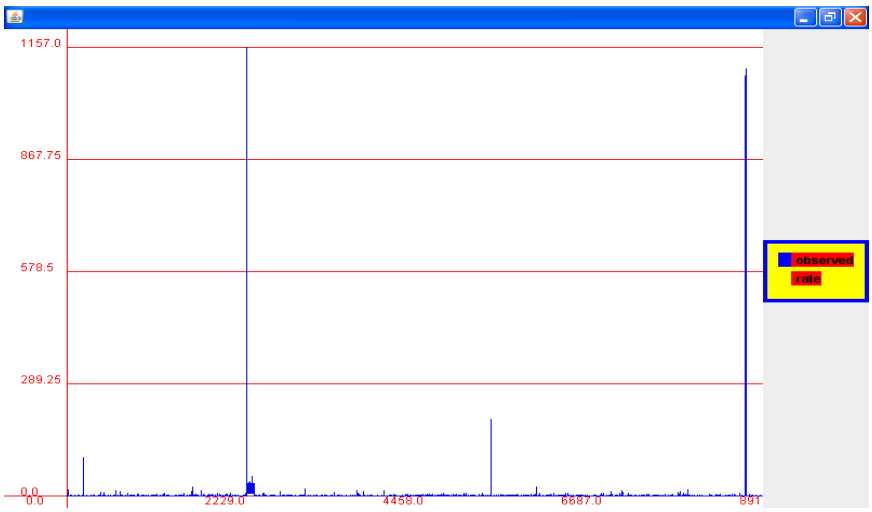


Figure 4-6: Packet Rate (No of packets per second under DoS attack)

But under certain situations, normal traffic rate is so high that no of packets per second increases unexpectedly. Therefore in order to reduce false alarm rate, chi-square values are calculated for each packet count as shown in Figure 4-7.When chi-square value exceed the specified threshold ,an alarm is generated.

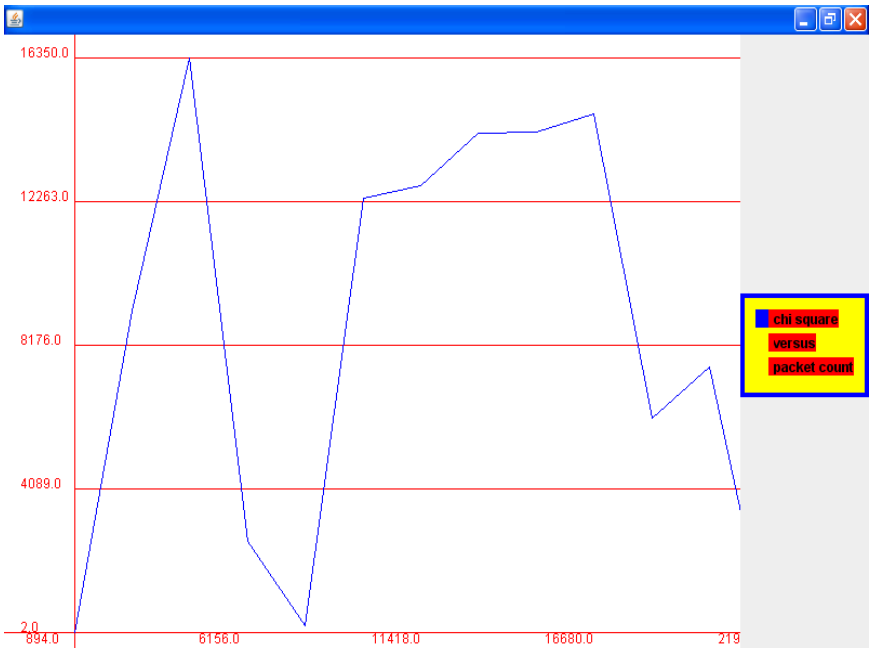


Figure 4-7: Chi square versus packet count under DoS attack

4.2.4.2.2 TCP Connection Graph:

Figure 4-8 illustrate the status of TCP connections when the system is under TCP SYN attack. Connections are exceeding the specified threshold.

Figure 4-8: Connection status under SYN DoS attack

4.3 Passive Mode:

The functionality of the Detection System has been evaluated by exhaustively testing the files from Database.The data was taken from the Lincoln Laboratory Massachusetts Institute of Technology data bank, DARPA datasets 98.One week tcpdump.list files, each containing packets in range from 20,000 to 1, 30,000 were used for testing the system. These log files contains simulated teardrop, smurf, ping of death and Neptune DoS attack packets.

Algorithm Results:

The individual data files have shown the results attached in Appendix A.

Overall Results:

Table 4-1: Overall Result Table

Detection rate %	**False alarm rate**
97%	0-0.86%

4.4 Conclusion

The system has been tested in the passive mode using DARPA MIT 1998 datasets and has given very promising results of 97.0% detection rate and 0.008 false alarm rate. Various graphs demonstrating changes in several parameters when attack traffic is encountered in active mode are presented.

5 Conclusion and Future Work

This chapter concludes the overall system giving its advantages and limitations if any.

5.1 Conclusion:

The implemented technique makes use of multiple parameters to detect DoS attack. Many approaches adopted earlier can detect only a particular type of attack. The technique to monitor different detection parameters such as no of connections, CPU usage, memory consumption and determining their deviation from the thresholds in conjunction with observing the deviation of observed packet rate from peak expected packet rate (chi-square statistic) leads to the development of more efficient, reliable and accurate DoS detection approach.

Higher rate DoS attacks with a short duration can be detected by the chi-square algorithm in early stages (within 1 or 2 seconds after it starts). The packet arrival rate based chi-square algorithm has been tested with DARPA 1998 datasets and has successfully detected SMURF, POD, NEPTUNE and TEARDROP (where the attack rate is higher) attacks.

5.2 Future Work:

The system lacks the ability of detecting slow rate DoS attacks in the initial stages but can detect them afterwards. To improve detection of slow rate attacks more research may be done.

APPENDIX A

RESULTS

TABLE 1: DATA SETS USED FOR EVALUATION

Dataset	Total packets	Normal packets	Attack packets	DoS attacks
DS_1	40815	40367	448	Pod(450)
DS_2	34642	34193	450	Pod(450)
DS_3	83453	46668	36785	Teardrop(200) Smurf(36385) Neptune(200)
DS_4	51144	50694	450	Pod(450)
DS_5	132168	18995	113173	Pod(448) Smurf(112725)
DS_6	46668	46129	539	Pod(0) Smurf(524) Neptune(15) Teardrop(0)
DS_7	37301	36851	450	Pod(450)

TABLE 2: EVALUATION RESULTS

Data set	Detection rate	False alarm rate
DS_1	0.995	0
DS_2	1.0	0
DS_3	0.85	0.06
DS_4	0.995	0
DS_5	0.998	0
DS_6	0.97	0
DS_7	1.0	0

APPENDIX B

SYSTEM REQUIREMENT SPECIFICATION

Anomaly Based Denial of Service Detection System

System Requirement Specification

Version (1.0)

ANOMALY BASED DoS DETECTION SYSTEM

Project Supervisor

- Lt. Col Mufassir-ul-Haque
  - Head of EE Department Military College of Signals

    (National University of Sciences and Technology)

Project Team

- Aamna Saeed (Project Leader)
- Rozina Nisa
- Nighat Perveen
- Rubina Shaheen

EXECUTIVE SUMMARY

Denial of Service (DoS) attacks is one of the greatest problems faced by the Network security resulting in loss of billions of dollars each year. Denial of Services is an attack on a computer system that causes a loss of service to users, typically the loss of network connectivity and services through the consumption of bandwidth of the victim network, or overloading the computational resources of the victim system. In a DoS attack a malicious user blocks legitimate users from accessing server services by exhausting or depleting the resources of the victim server. DoS attackers try to congest the network by raising huge amount of traffic rather than stealing the information from the victim server.

In DDoS attack, the victim server is attacked by numerous malicious users at a time, thus blocking legitimate users from accessing network services. Resources that are typically consumed are the network bandwidth, CPU cycles of server and data structures of particular protocols on server end.

The nature of the threats posed by DDoS attacks on large networks, such as the Internet, demands effective detection. Different approaches have been presented for detection of DoS attacks, but the major drawback of all these efforts were that they focused on any one of the parameters of network traffic thus having less accuracy and a large false alarm rate. Most of the approaches when given enough time can give good results but real-time DoS attack detection is a critical challenge that needs to be tackled before the attack succeeds.

In this project, analysis is done on the basis of more than one parameter, including memory usage, CPU usage, and number of connections and also making use of statistical analysis i- e computing Chi-Square statistics on distributions of network traffic applying test on specific attributes of packet. Accurate and sensitive DoS Detection system is required to help the servers to detect the malicious users and avoid giving them access.

The project is undertaken as a degree project hence no cost factor is involved or needs to be negotiated.

1. Introduction

1.1 Purpose

The purpose of this SRS is to clarify the system's requirements using stake holder's feedback. . This document is intended to outline System Requirements for the desired system and define them. The targeted audiences of this document are all the stakeholders including the user, designers and developers. The document is aimed at providing them an overview of the proposed system.

1.2 Scope

The project has a stupendous scope for the present time and the time to come. The proposed system will form a landmark in the field of Network Security. It is to develop a complete all purpose system that captures the run time network traffic and is able to automatically monitor and detect the DoS attacks at run time. The system's goal is to provide a successful operational solution to present network problems caused by a variety of attack tools available today and also attempts by attackers to evade detection. The project scope includes its colossal market value and great research potential. This system can be deployed on Web Servers and even on local computers which are providing access to other computers in a small network. The product cannot only be installed in critical systems but it also invites new courses of undergraduate and even graduate level for research and development.

1.2.1 Definition of System

"To design and develop an efficient, accurate and sensitive system in order to have necessary measures for detection of DoS attacks."

1.2.2 Implicit Requirements of the Defined System

Calls for a comprehensive Research Work

Since much work has been done earlier in this field but sill there is no solution to the existing problem. This shall serve the ends .Firstly, clarification, improvement and translation of the conceptual schema of the project into technical terms through successive and progressive experimentation. Secondly to provide a base/referential blueprint for the development works on the system.

Need for Extensive Experimentation

The research and development involves active and progressive experimentation. This experimentation is challenging due to a number of reasons such as concept evolution experimentation at various stages of design, validation of the developed / researched concepts and mathematical modeling and final testing and deployment of the system as working model.

1.2.3 Challenges

Study of Core of Networking

The design and development of the Detection system involves an in-depth understanding of phenomena such as the study of Internet and OSI (Open System Interconnection Model), thorough understanding of the packet header information and different protocols and thorough knowledge to determine how to compute parameters for algorithm evaluation.

1.3 References

All phases of development of this project will be done within the CS Dept, Communication Research lab, EE Dept, MCS. The proposal, requirements, design, implementation and other phases for the accomplishment of system will require the knowledge, help and expertise of the college resources.

1.4 Overview

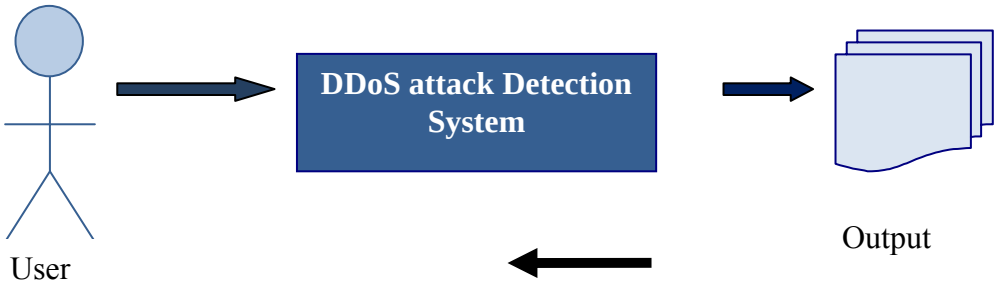
This document describes the functional features and details of Anomaly Based DoS Detection System. The SRS includes a brief product perspective and a summary of the functions the software will provide. It provides a description of the project for the stakeholders. User characteristics are discussed and any general constraints or assumptions and dependencies are listed.

2. The Overall Description

Anomaly Based DoS Detection System will be incorporated as an operational standalone application at Server end in a network. This system will shield the Server from the malicious traffic being burgled by the attacker. The network traffic itself will be able to provide runtime data which will then be used to calculate the required statistics for decision making. On the basis of calculated results and comparing them against the defined thresholds for the system, alarm will be generated if system is found under threat of DoS attack.

2.1 Product Perspective

Anomaly Based DoS Detection System is an independent system. The users of the system are external actors to it and can communicate with it via system's interfaces. Input is taken as the network traffic itself on runtime on the Server by capturing the packets. It uses captured packets to evaluate and generate results. Outputs are given as reports, graphs and alarms.



2.1.1 System Interfaces

The system, as proposed, interacts with the incoming network traffic information. The network traffic is handled in terms of packets (Network layer). For this interaction, the system needs interfaces to support information maintenance including packets capturing and sniffing. The main information source that the system interfaces with is the traffic that generates run time data to be operated upon.

2.1.2 Interfaces

The system will provide user interfaces to support its operations. It will provide a generic interface for the server side to check the computer status. It will show the packets being captured along with its header information, the memory status, number of processes running on the system and the graphs showing the statistics results. The header information of the packets captured at the network card will be taken as input and given to the system for operations. It will also give consolidated information about the Server status, as in normal or under threat of attack, and mark any possible vulnerability and unusual operations. More interfaces can be developed and integrated if required by the user/system.

2.1.3 Hardware Interfaces

Presently, Intel (R) PRO/100 VE Network Connection (Microsoft Packet Scheduler) Device and Computers to create Client Server Environment are needed but more can be required if advance level support/functionality is desired of the system.

2.1.4 Software Interfaces

The system will be developed to interface with WinPcap Driver (4.1 Version), Jpcap 6.0 and JVM (compatible with jdk1.6) .

It will provide client server architecture for user interaction and follow the distributed systems protocols. More software interfaces can be added at the time of design and/or development.

2.1.5 Operations

The system will remain operational all the time without any effects of operational issues. The mode of operation will be manual where user interaction is involved and automated for system's internal process(es) involving automated traffic capturing, analysis, generation of graphs and ringing of alarm at the server.

2.2 Product Functions

Software shall provide complete HELP for each and every aspect of the Software, acquire runtime network traffic, analysis of network traffic(packets captured) using a specific algorithm (probability statistics) and using other algorithms (memory consumption, CPU Usage), false alarm rate calculation, parameters results computation and show results through graphs and Pie-Charts

2.3 User Characteristics

Users that interact with the system are System Administrator, Engineers, Technicians, Network administrators and Server Administrator.

Based on above type of users and their skills following precautions should be taken

1. The interfaces shall be designed keeping novice in mind.
2. Error messages shall be used wherever required.
3. Users shall be consulted throughout design
4. Help option will be provided to all users.
5. Interfaces should be user friendly.

2.4 Constraints

The system is subjected a few operational constraints for its design and development. This can be refined and extended if the needed. Develop a system that fulfils the distributed client server architecture requirements. Provide interface(s) between the server and client applications.

3. Specific Requirements

3.1 General Overview

System Inputs

Network Traffic

Runtime network traffic serves as input to Anomaly Based DoS Detection System.

System Outputs

All parameters results and Alarm Status

3.2 Functions

The system is required to perform the following orderly operations:

i. Packets Sniffing At the Sever Machine
ii. Preprocessing
iii. Application of Algorithms
iv. Graphical User Interface (GUI)
v. Main Server Interface
vi. Cumulative Packet Information according to Network Layers
vii. Continuous Packets Information display (runtime Graphs)
viii. Active Mode
ix. Passive mode
x. Results View

3.3 Design Constraints

There are no design constraints for the system. It is subject to the developer's choice. All the hardware/software support required for system's development is currently available.

3.3 Software System Attributes

There are a number of attributes of the system that serve as nonfunctional requirements. These required attributes are highlighted here so that their achievement can be objectively verified.

3.3.1 Reliability

The system's reliability is important attribute for its functional working. It shall be reliable enough to work well with huge amount of network traffic and handle service request send to the server. Its reliability is most important in case of its output, in generations of alarms, where accurate results and comparisons are required to reduce false alarm rate.

3.3.2 Availability

The system is expected to be available to the server all the time to monitor the network traffic. Any chances of unavailability or malfunctioning should be minimized to get the optimized results.

4. Change Management Process

The change management process to be applied after reviews and considerations to update the SRS document .This is to effectively reflect changes in requirements/ scope of the project.

APPENDIX C

SOFTWARE DEVELOPMENT PLAN DOCUMENT

Anomaly Based Denial of Service Detection System

Software Development Plan

Version (1.0)

Preface

In this document the comprehensive development and management plan for Anomaly Based DoS Detection System is described. This plan will be strictly followed over a course of eight months. Any amendment applied to this plan will be incorporated in the next versions of this document.

The document includes the details of the software to be delivered; major activities, major deliverables, major milestones, required resources, and top-level schedule.

1. Introduction

This section contains the details of the project and the software product to be built. In this section a brief overview of the project is given.

1.1 Project Overview

Anomaly based DoS Detection System is a complete solution for monitoring network traffic, revealing information at different layers of Internet Model and leading to accurate DoS detection.

1.1.1 Product Functions

Anomaly Based DoS Detection System is a multipurpose software kit, which performs various important functions; it provides the capability to perform DoS detection. It is capable to divulge layers information (Application Layer, Transport Layer, and Network Layer). It has an integrated database of DARPA datasets.

1.1.2 Minimum Requirements

The minimum requirement for the software to be operational requires Microsoft WINDOWS XP, JDK 1.6, and Jpcap-0.6 with WinPcap_ 4-01 Driver.

1.1.3 Major milestones

The key milestones of the project are

- Completion of Requirement Analysis and Project Specifications phase.
- Completion of Design phase
- Implementation and Integration of components.
- Product testing and installation

The minutiae of the software to be delivered, major activities, major deliverables, major milestones, required resources, and top-level schedule will be followed in the sections to come.

1.2 Project Deliverables

This section delineates the major items to be delivered to the customer. Following are the details:

1.2.1 List of Project Deliverables

List of project deliverable is as follows:

Deliverable Name	Due Date
Project Definition and List of team Members	15th MAY 2007
Project Plan	1st June 2007
Requirements Description	15th June 2007
Requirement Analysis	1st July 2007
Project Specifications	26th August 2007
Testing Plan	15th December 2007
Detail Design Document	20th December 2007
Fully Functional Product Model	1st March 2008
Project Report and Review	7th March 2008

Figure 1: List of Project Deliverables

Further deliverables will be provided upon request.

1.2.2 Details of the Deliverables

Following are brief details of the project deliverables

Project Definition and List of team Members

This includes defining the project and list of team members.

Project Plan

This document describes how the team is to be structured and administered for the duration of the project.

Requirements Description

The requirements of the proposed software system, on which design and coding is based, is outlined in this document.

Requirement Analysis

The document includes analysis of user requirements based on requirements document.

Project Specifications

This document will describe detailed specification for Anomaly Based DoS Detection defining parameters and interface.

Testing Plan

A complete testing plan describing how the testing will be carried out.

Detail Design Document

This document builds on the high-level design, incorporates the Design Model made in Rational Rose Software and further details the specifics of implementation and key algorithms as required.

Fully Functional Product Model

A fully functional product model will be demonstrated. The full source code will be given if requested.

Project Report and Review

This will include complete project report explaining what was achieved during the project. Also further enhancement possible will also be described.

1.3 Reference Materials

Refer Appendix A

1.4 Definitions and Acronyms

Refer Appendix E

2. Project Organization

This section describes the development structure of the project which includes the process model, the organizational structure and responsibilities of individuals on the project.

2.1 Process Model

2.1.1 Project Life Cycle Model

The system will be developed using spiral life-cycle model. The model is shown in Figure 2:

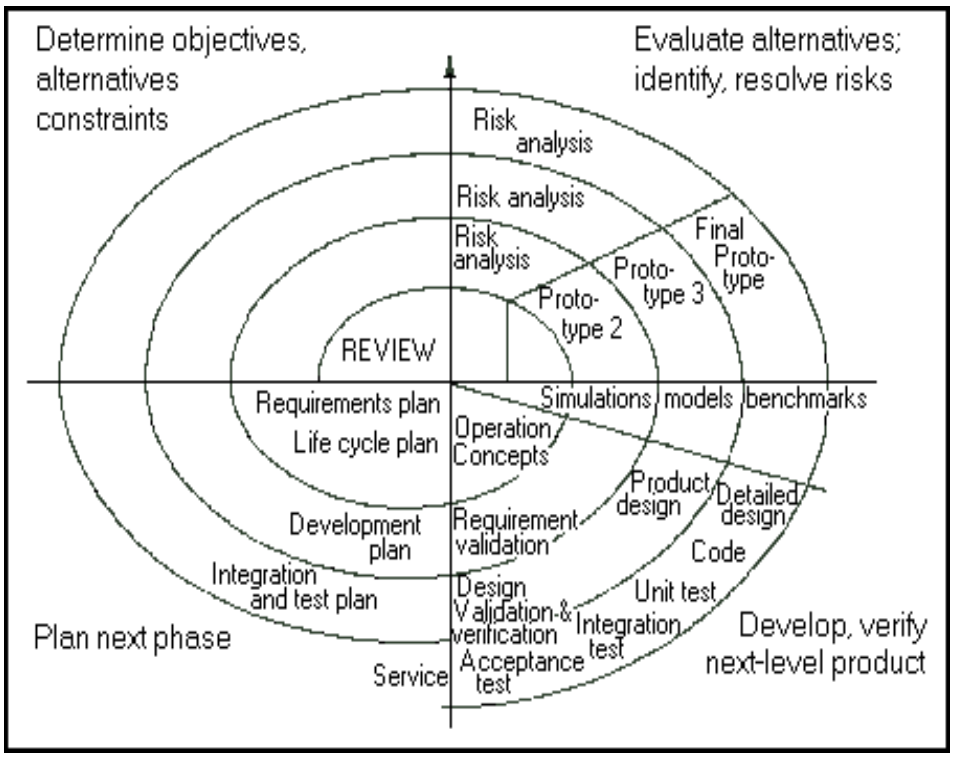


Figure 2: Spiral Model

2.1.2 Significance

- Incremental Model is best suited for Anomaly Based DoS DS as:
- The Spiral model allows for risk management and is better from waterfall which places too much emphasis on project management.
- There are only four team members. So each and every member will have to participate in all the phases and deliverables.
- Allows developer to determine and to counter to risks at each evolutionary level
- Direct consideration of risks at all levels greatly reduces problems

2.1.3 Project Work Products

Following work products are listed for completion. Accompanying relevant details of due dates and concerned personnel is also listed:

WORK PRODUCT NAME	Placed Under Change Control?	Deliverable to Customer?	People Who Must Sign Off on the Work Product
REQUIREMENTS DESCRIPTION The requirements of the proposed software system, on which design and coding is based, is outlined in this document.	Yes	Yes	Project Manager, CEO, Requirements Engineer
REQUIREMENT ANALYSIS The document includes analysis of user requirements based on requirements document.	Yes	Yes	Project Manager, CEO, Requirements Engineer
DETAIL DESIGN DOCUMENT This document builds on the high-level design, incorporates the Design Model made in Rational	Yes	No	Project Manager, CEO , Designer

Rose Software and further details the specifics of implementation and key algorithms as required. .			
Fully Functional Product Model A fully functional product model will be demonstrated. The full source code will be given if requested.	Yes	Yes	Project Manager, CEO
PROJECT REPORT AND REVIEW This will include complete project report explaining what was achieved during the project. Also further enhancement possible will also be described.	Yes	Yes	Project Manager, CEO

2.2 Organizational Structure

2.2.1 Parent Organization

The personnel involved in this project are as follows:

- amna Saeed (Chief Executive Officer, Quality Assurance Manager)
- Rozina Nisa (Chief Technical Officer, Customer Analyst,)
- Nighat Perveen (Chief Operational Officer, User Interface Prototyper)
- Rubina Shaheen(Chief Project Officer, Designer)

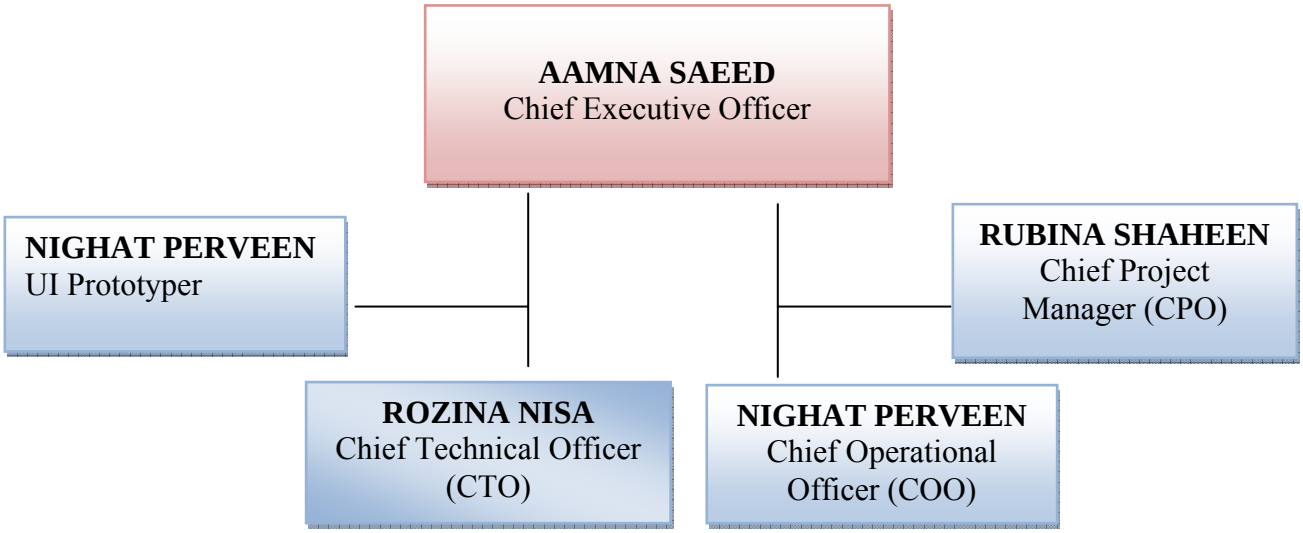


Figure 3: Organization Chart of the Executive Staff

Each member is equipped with significant Computer Science, Mathematics and other skills acquired during education tenure to accomplish the project.

All syndicate members are expected to contribute approximately equal amounts of work to the project on a weekly basis. This workload needs to be balanced with other

courses team members are undertaking. It is expected that a weekly workload for this project should be a minimum of approximately 30 hours/week.

2.3 Organizational Boundaries and Interfaces

- Regular meetings with client will be held. Approval of the project progress is taken by the client at least once a week.
- The project documentation will not be revealed to ant party other than the client by the team members.
- Every member will have to attend team meetings unless he/she has a valid reason. Team members will meet at least twice a week.
- Informal Communication will also be used extensively in this project due to its quick and efficient information transferable properties. The team makes use of personal communication and email communication as much as possible. However, given the academic restrictions, email communication is used as the primary form of interacting with other team members.
- Project Manager will be responsible for communicating with the upper management.
- Direct communication will be held with the Customer organization.
- The Project manager will be responsible for communication with the customer organization.
- There are no subcontracting organization(s) associated with project.

2.4 Project Responsibilities

The following chart illustrates the persons and their respective responsibilities concerning this project. Any alterations in the given structure will be incorporated in the future versions of the document.

Responsibility	***Persons Responsible***
Overall Project Manager	Aamna Saeed
Quality Assurance Manager	Aamna Saeed
End-User Documentation Manager	Rozina Nisa
Requirements Engineer	Nighat Perveen
Software Architecture	Rubina Shaheen
Technical Self-Reviews	Nighat Perveen

Figure 4: Project Responsibilities Chart

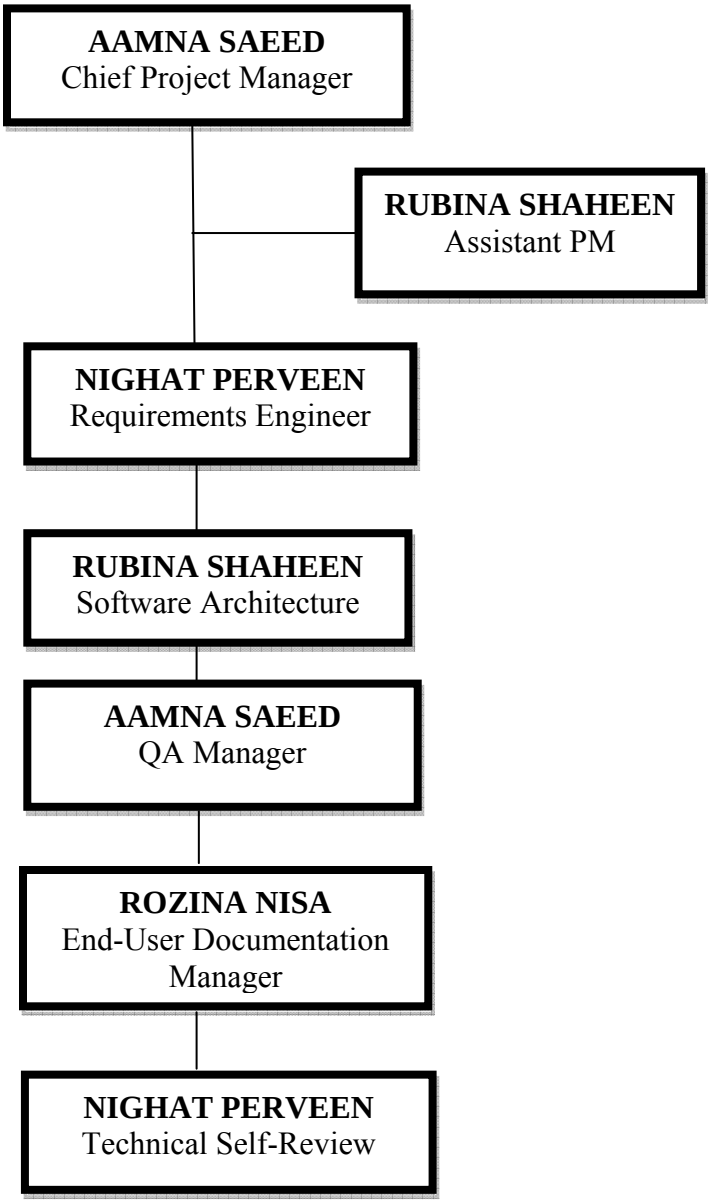


Figure 5: Organizational Chart for Project Responsibilities

3. Managerial Process

In this section we describe strategy that will be adopted to manage the project**.** This includes management objectives, priorities, project assumptions, dependencies, constraints, risk management techniques, monitoring and controlling mechanisms, and the staffing plan.

3.1 Management Objectives and Priorities

The major objective is to ensure that time constraints are strictly followed. The quality of work is also given priority. During the earlier stages of the project main concern will be following schedule.

Cost will not be as much a major factor in this project. Though this is a commercial project but it does not fall into the category of high budget projects, so cost management is not the primary goal. Due to work load priority will be, to meet time constraints and to make sure work is completed on schedule.

More priority will be given to functionality as well. The main aim of the project will be to achieve the maximum possible efficient functions of the project in the minimum possible time. The project team will attempt to strike maximum possible balance between time and functional priority.

3.2 Assumptions, Dependencies and Constraints

It is assumed that the schedule will not clash with the academic schedule of the team members. The team will able to setup the project to test the product in real time.

3.3 Risk Management

The project plan has been designed to address the major risks of the project. It also describes how risks will be tracked and monitored.

3.3.1 Major Risks

The major risks that can impinge on the project are

- Clash of project schedule with other activities of team members
- Shortage of time
- Availability of testing software
    - Attack tools

- Test Datasets

- Lack of software engineering experience and unfamiliarity with tools, compatibility of different versions of supporting softwares with hardware.

3.3.2 Risk Tracking and Monitoring

To avoid work overload the tasks will be distributed among team members in such a way that every personnel is able to give enough time. Every phase will have an informal sub schedule. Main schedule is made flexible enough to accommodate mishaps and undesirable circumstances

It will be made sure before integration phase or during the test plan definition phase that desired hardware and software is acquired.

Every team member will be given time to get familiar to the tools and different software. Help materials will be acquired for this purpose. Also necessary in time help will be requested from supervisor.

Any risk factor, suggested by a team member or client will be included in this section for next versions. The possibility of risks emerging during the project is very high so a risk list along with the avoidance mechanism will be kept and updated accordingly.

The possibility of risks emerging after the project is very high as DoS attacks are of numerous types and to cater for as many as possible attack types, issue is to be addressed which has to be incorporated for an accurate DoS detection. In case any risk cannot be overcome the priority will be given to the quality of the product as mentioned earlier that cost is not a factor for this project.

3.4 Progress Monitoring and Controlling Mechanism

3.4.1 Monitoring and Controlling Mechanisms

Project cost, schedule, quality, and functionality will be tracked throughout the project. In a weekly meeting the progress will be reviewed and analyzed by Project Manager and Quality Assurance Manager. Problems encountered by any team member will be discussed and resolved accordingly.

3.4.1.1 Report contents

Separate reports shall be primed for monitoring the functional, technical, quality and cost scrutinize of the project. Progress reports will also be maintained in due course of time. The status reports will contain the following details:

- Status of the current phase activities
- Estimated time of completion of the current phase
- Milestone deliverables at current phase
- Schedule for the next phase activities

The status reports will be accompanied by Gantt Charts. Various other graphs will also be presented to support the facts laid down in the status reports.

3.4.1.2 Reporting structure and frequency

The reporting structure will be developed and circulated to all concerned parties. All reports will be developed on a weekly basis.

3.5 Staffing Plan

Considering the complexity of project at hand and fixed number of personnel, the minimum skill levels are not defined at this stage. The entire duration of the project will be headed and dealt with the same time, each individual acquiring different role during different phases.

There will be no extra personnel acquired during the course of the project besides those that are currently involved in the project.

Type of Personnel's	Number of personnel	Required Skill Level / Qualification(s)
Requirement Engineer (s)	2	Experience in Requirements Engineering
Software Designer (s)	2	Software Engineering
Project Manager	1	Software Engineering
Coder (s)	3	Excellent Coding in C++ and JAVA.
QA Manager	1	Experience as QA Officer
Test Officer (s)	2	Experience as Test Officer

Figure 6: Staffing Plan Chart

4. Technical Process

This section explains the top-level technical processes used on the project including the technical methods, tools, and techniques; major software documents; and supporting activities such as configuration management and quality assurance.

4.1 Methods, Tools and Techniques

4.1.1 Operating Environment

The operating environment will be that of Microsoft Windows. Installation of JDK 1.6, Jpcap 0.6 along with WinPcap_ 4-01 Driver is mandatory.

4.1.2 Hardware

- Stand alone IBM PCs.
- Intel (R) PRO/100 VE Network Connection (Microsoft Packet Scheduler) Device

4.1.3 Software tools

- Compiler or IDE : JDK 1.6
- Programming language : JAVA and C++
- Coding standards : IEEE
- Documentation standards : IEEE

Remarks

Software will have object-oriented reusable structure. Documentation will be clear and explanatory. Microsoft Project software will be used to aid in management.

4.2 Software Documentation

The following documents will be developed for the project, including milestones, reviews, and sign offs for each document.

- Project Plan
- Requirements Document
- Requirement Analysis
- Specifications
- Detailed Design
- Testing Plan
- User Interface Plan
- Product Review

4.3 Project Support Functions

The project is supported by following other documents. These documents describe the plans for functions that support the software development effort.

- Configuration Documentation
- End - user documentation (USER MANUAL)

The plans for these supporting functions will be developed in due time and should be referred to as the need arises. Any other supporting documents that need to be included on the list of the above documents will be added in the later versions of this document.

5. Work Packages, Schedule, and Budget

5.1 Work Packages

The work packages defined for the software development lifecycle are as below. These including the sub-packages and tasks that must be completed in order to complete the software.

1. Software Project Initiation
2. Software Concept Development
3. Software Requirements Development
4. Software Architectural Design
5. Software Development
6. Software Implementation and Integration
7. Software Testing and Configuration Management

5.2 Dependencies

All the work packages discussed above are extensively interdependent. Every package is dependent upon all the packages illustrated before it.

5.3 Resource Requirements

The following resources will be needed.

Hardware Tools:Stand alone IBM PCs, Intel (R) PRO/100 VE Network Connection (Microsoft Packet Scheduler) Device

Software Tools: Programming tools (JAVA and C++)

Additional resources will be identified in the later versions of this document.

5.4 Budget and Resource Allocation

Budget Allocation

Following is the amount of Budget allocated to the Software processes.

Software Requirements Management	:	15 %
Software Design	:	15 %
Software Implementation	:	25 %
Software Testing and Integration	:	23%
Documentation	:	7%
Software Project Management	:	15 %

5.5 Project Schedule

During the development process, the following schedule will be followed. The schedule is based on project deliverables. As the project development progresses, the schedule given below is subject to change. Gantt charts also correspond to this schedule.

PROJECT SCHEDULE FOR ANOMALY BASED DoS DETECTION SYSTEM

Based on Project Deliverables, following Chart illustrates the Project Schedule.

Deliverable Name	Due Date
Project Definition and List of team Members	15th MAY 2007
Project Plan	1st June 2007
Requirements Description	15th June 2007
Requirement Analysis	1st July 2007
Project Specifications	26th August 2007
Testing Plan	15th December 2007
Detail Design Document	20th December 2007
Fully Functional Product Model	1th March 2008
Project Report and Review	7th March 2008

Figure 7-Chart for Anomaly Based DoS Detection Project Development Schedule

6. Additional Components

This section defines the additional components needed to manage this project.

- Training plans
- Installation plans

Details of these areas will be make available in separate documents in due time.

APPENDIX D

TEST PLAN DOCUMENT

Anomaly Based Denial of Service Detection System

Test Plan Document

Version (1.0)

Document & Version Control

Document Information

Document Title	Test Plan for Anomaly based DoS Detection System
Filename	TestPlan_CIS_v1.0.doc
Creation Date	February 15, 2008.
Authors	Aamna Saeed,Rozina Nisa,Nighat Perveen,Rubina Shaheen
Version	1.0
Distribution List	1. Nighat Perveen 2. Rozina Nisa
Reviewers	1. Aamna Saeed 2. Rubina Shaheen
Passing Review Date	1st March, 2008.

Related Documents

Input Documents	System Requirements Specifications (Annex A-1)
Output Documents	Test Case Document

Document History

S. No.	Document Version	Major Changes	Author/s	Creation Date	Review Date	Reviewer/s
1.	1.0	Initial Draft	Aamna Saeed Rozina Nisa Rubina Shaheen Nighat Perveen	February. 15, 2008.	March 1, 2008.	Aamna Saeed Rubina Shaheen

Project Information

Project Release Information

Project Name	Anomaly Based DoS Detection System
Project Code	Anomaly Based DoSDS 08
Project Release Number	1.0
Project Release Date	February 15, 2008.
Testing Dates	February 15, 2008 – March 1, 2008.
Testing Iteration Number	1.0
Project Modules Information	Software

Project Team Information

Group Leader	Aamna Saeed
Developer Name(s)	1. Rozina Nisa 2. Rubina Sheheen

	3. Nighat Perveen
Quality Analyst Name	Rozina Nisa
Tester Name(s)	1. Rozina Nisa 2. Nighat Perveen

Remarks

Introduction

Purpose of Document

This document is the master test plan for **Anomaly Based DoS Detection System**. It serves as guide to conduct test plans for this project. This document defines the scope, approach, schedule, risks/mitigations, and entry/exit criteria that are required as part of project planning prior to Execution.

The purpose of this Software Test Plan is to make sure that the requirements stated in the Software Requirement Specification (SRS) are fulfilled and software is functioning properly.

Product Scope

The product cannot only be installed in critical systems but it also invites new courses of undergraduate and even graduate level for research and development. This system can be deployed by any organization related to Internet E-Commerce community. It also has an important application in sensitive military areas. This smart software provides a completely automated solution to the problems caused by current as well as future growing security threats.

Intended Audiences

Intended audience includes test personnel who use the documented plan for the software/system test activities. Other stakeholders include Project DS, Group Leader and system developers.

Test Objectives

The tests included in this test plan are to be conducted to verify the software implementations for the Anomaly Based DoS Detection System. Major part of the test work includes software testing for Anomaly Based DoS Detection System components.

Test Case Matrix

List of all the test cases is attached.

Software

The following software /system will be used in the testing process:

- Resource Detector
- Statistical Detector
- Abnormality (DoS attack) Identification
- GUI

Testing Tools

Nemesis 1-4 is used to generate crafted packets and verify DoS Detection.

Recommendations:

Software for bug tracking system is highly recommended for future phases of this project.

Test Personnel

Parties	Contact Person	Role and Responsibilities
DS	Lt. Col. Moffasir- ul- Haq	• Overall Supervision • Work stream Management • Review of Test Plan and Test Cases • Monitor testing schedule and procedure.
Testing Team Lead	Aamna Saeed (aamna.saeed87@gmail.com)	• QA Team Lead • Development of Test Plan and Test Cases. • Managing and directing testing activity • To Ensure Testing activity comply with project and test plan • Conduct testing
Test Engineer	Rozina Nisa (rozina.nisa@gmail.com)	• Develop test cases and conduct testing • Submit Bug Reports

Testing Schedule

Task	Start Date	End Date	Days
Document Requirement	15-02-08	18-02-08	3
Document Design	19-02-08	19-02-08	1
Create Test Cases	20-02-08	21-02-08	2
Conduct walk-through of Test Plan and Test Cases	22-02-08	22-02-08	1
Test Case Execution	23-02-08	27-02-08	5
Retest incidents and Regression Testing	28-02-08	29-02-08	2
Sign off on Test	30-02-08	01-03-08	2

Risks, Dependencies and Assumptions

Since the development of the Anomaly Based DoS Detection System is almost completed and final, so unit and integration testing of the software is not possible. Only the feature testing for Software will be conducted.

The following is a list of risks and contingencies for this testing procedure.

- The source code for Anomaly Based DoS Detection System should be available by the start date of testing i.e. 15-02-2008.
- In case of some critical or major bug, time is very limited for consultation with developers. Also there should be some pre-defined procedure to contact with developers for defect management etc.
- The test plan and test schedule are based on the current Requirements Document. Any changes to the requirements could affect the test schedule.

Defect Management (DM) Process

Defects (bugs) found during test execution will be captured and recorded by the testers in a separate database. Development Manager, Project Manager and all concerned development team members will share these reports. The following steps are included in the bug report .

- Title or Summary
- Description of the Problem
- Software release version or phase etc.
- Priority
- Functional area: which software block/module is suspected to have problem
- Traces: attach any trace files/screen shots that are taken as part of the analysis.
- Currently assigned to: e.g. development manager, developer
- Status: Open or closed
- Remarks/comments

Bug Severity Levels

S. No.	Severity	Definition
1.	Critical	A major part of the system cannot be used, or system / data integrity is in doubt, which causes a halt to the testing. All available resources must be used to resolve the incident, examples are: • Application crashes forcing the tester off the system. • Output not generated.
2.	Major	Must be fixed before the final sign off the testing phase. Productivity is adversely affected e.g. Error condition is created which was not the

		expected result of the test, for which there is no recovery for that test, although the user can select other tasks and run other tests.
3.	Medium	Use of the relevant process can continue with a temporary workaround, but a fix must be scheduled before exit. Examples are: • Cosmetic errors. • Problems that do not cause easily noticeable faults.
4.	Minor	Errors that have a little or no impact on interim use or productivity. Fixing these bugs before exit is not mandatory.
5.	Enhancement	Not in specifications, but desirable according to tester.

Debugging Priority Levels

Priority	Definition
Critical	Resolve immediately
Major	Give High Attention
Medium	Normal Queue
Minor	Low Priority

Testing Approach

Entry Criteria

- Sign-off and baseline Requirement and Design specification.
- Completion of this test plan, test matrix and test cases.

- Completion of Unit Testing by Software Developers.
- Testers have been committed to testing and completed the sufficient training provided by developers.

Exit Criteria

- All Critical & High defects have been fixed and a clear strategy has been devised on how to handle any remaining Medium priority defects.
- Completion of all Testing activities according to the planned test cases and conditions.
- All problems/incidents have been resolved, fixed, retested successfully and closed.
- Full record of all problems / incidents and evidence of satisfactory retest maintained.
- All Testing Activities have been stated at Test Report.

Test Execution Procedure

Test Execution steps are follows:

- Ensure the code and the test environment is in place.
- Execute the test cases according to the pre-defined test sequence
    - Document the actual results
    - Compare the actual results against the expected results
- Document any discrepancies/bugs in the 'Bug Tracking Document' according to the Incident Tracking Procedures. Assign Bug tracking document to applicable owner. For this purpose an email will be sent to the respective development engineer.
- A brief report showing status for test cases will be issued periodically.
- Errors and incidents will be updated periodically.
- Re-test issues that are turned back for re-test.Regressions tests will be conducted after consulting with development team (if necessary), which may have been affected as a result of the fixes.

Features and flow testing for Software part of **Anomaly Based DoS Detection System** will be tested. In general following types of testing will be conducted:

- Startup/shutdown Testing
- Feature/function Testing
- Negative Testing
- Unit Testing
- Integration Testing

Testing Types/Functions Not Used

Any other functionality except Software modules will not be tested. In general the following type of testing will not be conducted:

- Installation/Migration Testing
- System level Regression Testing

Test Deliverables

- Test Plan
- Test Cases
- Test Cases summary/status Report
- Bug Report Tracking Documents
- Final Test Summary Report

Test Cases

Module Name: Software

Module Information

Module Name	Software Module
Module Release Number	1.0
Module Release Date	15-02-08
Testing Dates	16-02-06 to 28-02-08
Testing Iteration Number	1
Developer Name(s)	1. Aamna Saeed 2. Rozina Nisa 3. Rubina Shaheen 4. Nighat Perveen

Functional Requirements

Module Description	The Software comprises of various modules integrated together.
Functional Requirements	1. Main Menu 2. Active Mode 3. Passive Mode 4. Record Saving

Test Case Table

Test Case Name/Title: Main Menu	
Test Case Number	**FN-REQ-001-DoSDS0001**
Purpose	To see the Main Menu for Anomaly Based DoS Detection with options for active mode and passive mode.
Precondition	JDK 1.6, WinPcap 4.0, JPcap 0.6 must be installed. CPUUSAGE.dll must be placed in lib and bin of jdk 1.6 and jre 1.6. Start main_menu
Procedure	Initialize main_menu Test all Options • Open User Manual • Open Options o "What is Anomaly Based DoS" o "What is Chi-square" o DoS Detection Demo • Check Information About o NUST(MCS) o Data and Information Flow o Contact Information
Expected Result	All associated files open in their respective formats(.txt and .batch)
Actual Result	Files opened
Status	Pass
Bug ID	<In case the test fails, specify a bug id to track it's changes throughout debugging>
Tester	Nighat Perveen
Date	18-02-08
Remarks	<Additional remarks>

Test Case Name/Title: Active Mode	
Test Case Number	**FN-REQ-001-DoSDS0002**
Purpose	Perform real time DoS Detection
Precondition	Click on ACTIVE MODE button in main_menu
Procedure	1. Open ACTIVE MODE. 2. Resource monitoring is started automatically when ACTIVE MODE window is opened. 3. Start Sniffer by clicking on sniffer button on left. 4. Chi-square analyses on basis of no of packets per second (traffic rate).To shift to frequency based analysis click on button FREQUENCY BASED ANALYSIS.
Expected Result	Successful execution of all four algorithms o Chi-square packet rate based o Chi-square frequency based o CPUUSAGE algorithm o Memory Monitoring algorithm
Status	Pass
Bug ID	<In case the test fails, specify a bug id to track it's changes throughout debugging>
Tester	Rozina Nisa
Date	22-02-08
Remarks	<Additional remarks>
	Noise data results bugs reported to System Developer with tested values.

Test Case Name/Title: Passive Mode	
Test Case Number	**FN-REQ-001-DoSDS0003**
Purpose	Perform offline data analysis and detect DoS attack.
Precondition	Click on PASSIVE MODE button in main_menu
Procedure	1. Open PASSIVE MODE. 2. Choose the data file to be analyzed (in the left listbox or type directly into the edit window). 3. Press START button.
Expected Result	Successful execution of all algorithm o Chi-square rate based
Status	Pass
Bug ID	<In case the test fails, specify a bug id to track it's changes throughout debugging>
Tester	Aamna Saeed
Date	24-02-08
Remarks	<Additional remarks>
	Noise data results bugs reported to System Developer with tested values.

Test Case Name/Title: Record Saving	
Test Case Number	**FN-REQ-001-DoSDS0004**
Purpose	To save traffic captured by sniffer in a log file for further reference.
Precondition	Capture internet traffic from …….
Procedure	• Stop the sniffer. • Press YES/NO button to save internet traffic or not.
Expected Result	The files are correctly saved on PC in desired result directory.
Actual Result	The files are correctly stored in PC.
Status	Pass
Bug ID	<In case the test fails, specify a bug id to track it's changes throughout debugging>
Tester	Rubina Shaheen
Date	25-02-08
Remarks	<Additional remarks>

Test Case Name/Title: Results Display	
Test Case Number	**FN-REQ-001-DoSDS0005**
Purpose	To display the result graphs.
Precondition	Run the DoS System in either ACTIVE or PASS.
Procedure	1. When in Passive Mode select the desired data to be analyzed. 2. When in Active mode start the sniffer.
Expected Result	Accuracy must be close to desired values.
Actual Result	Improved Accuracy obtained
Status	Pass

Bug ID	<In case the test fails, specify a bug id to track it's changes throughout debugging>
Tester	Rozina Nisa
Date	26-02-08
Remarks	<Additional remarks>

Test Summary Report

The tests have shown the following results

- All software modules preliminary execution shows desired results
- All modules integration testing illustrate no problem
- Desired results of Algorithms demonstrate improved values of parameters. The mean values of Anomaly Based DoS Detection System are:

Detection rate %	**False alarm rate**
97%	0-0.86%

BILBIOGRAPHY:

[1] Glenn Carl, George Kesidis, Richard R. Brooks, and Suresh Rai, "Denial-of-Service Attack-Detection Techniques," IEEE Internet Computing, vol. 10, no. 1, 2006, pp. 82-89.

[2] "BCS Cyber Crime" http://www.bcs.org/server.php.

[3] Yang Wang, Chuang Lin , Quan-Lin Li and Yuguang Fang "A queuing analysis for the denial of service (DoS) attacks in computer networks, " Computer Networks vol 51, Issue 12, 22 August 2007, pp. 3564-3573.

[4] Abhishek Singh "Demystifying Denial-Of-Service attacks, part one" www.security.focus.com/infocus/1853 , 14 December, 2005

[5] Rober J. Shimonski "What You Need to Know About Intrusion Detection Systems" http://www.windowsecurity.com/articles/ What You Need to Know About Intrusion Detection Systems.html, 18 November 2002.

[6] Arnt Brox "Signature-Based or Anomaly-Based Intrusion Detection: The Practice and Pitfalls"http://www.scmagazine.com/asia/news/article/419779/signature-based-anomaly-based-intrusion-detection-practice-pitfalls/ , 1 May 2002.

[7] Anazida Zainal, Mohd Aizaini Maarof , Siti Mariyam Shamsuddin "Research issues in Adaptive Intrusion Detection". Proceeding of the 2nd Postgraduate Annual Research Seminar, PARS'06, Faculty of Computer Science & Information Systems, Universiti Teknologi Malaysia, 24 - 25 May, 2006.

[8] Yuichi Ohsita, Shingo Ata, Masayuki Murata (2004). "Detecting Distributed Denial of Service Attacks by analyzing TCP SYN packets statistically".IEICE Transaction on Communication 2006.pp 2868-2877.

[9] Li, J., Zhang, G. Y. and GuG. C. "The Research and Implementation of Intelligent Intrusion Detection System Based on Artificial Neural Network." IEEE Proceedings of the 3rd International Conference on Machine Learning and Cybernetics. pp. 3178-3182.

[10] Chen, W. H., Hsu, S. H. and Shen, H. P. (2005a). "Application of SVM and ANN for Intrusion Detection" Journal of Computers & Operations Research vol. 32. pp. 2617-2634.

[11] Pan, Y., Chen, D., Guo, M and Cao, J., Dongarra, J. J. (2005). "A Hybrid Neural Network Approach to the Classification of Novel Attacks for Intrusion Detection" In Proceedings of Parallel and Distributed Processing and Applications Third International Symposium, ISPA 2005, Nanjing, China. vol. 3758.

[12] Aikaterini Mitrokosta and Chistos Douligeris(2005). "Detecting Denial of Service Attacks Using Emergent Self-Organizing Maps". Proceedings of the Fifth IEEE International Symposium on Volume, Issue, 18-21 Dec. 2005 pp. 375 – 380.

[13] Aura Feinstein, Dan Schnackenberg, Ravindra Balupari and Darrell Kindred"Statistical Approaches to DDoS Attack Detection and Response", IEEE Proceedings of the DARPA Information Survivability Conference and Exposition 2003.

[14] Vicky Laurens, Abdulmotaleb El Saddik and PulakDharVineet Srivasta, "Detecting Distributed Denial of Service Attack Traffic at the Agent Machines". Electrical and Computer Engineering, 2007. CCECE 2007.

[15] B. Rhodes, J. Mahaffey and J. Cannady, "Multiple Detection", Proceedings of the NISSC 2000 conference.

[16] Jeff Nathan nemesis-1.4 (Build 26) **(**DSA sig**)** (includes LibnetNT).
[17] “TCP Connection States and Netstat output”http://support.microsoft.com/kb/137984.
[18] Gregor Hohpe, "Let's Have a Conversation," IEEE Internet Computing, vol. 11, no. 3, 2007, pp. 78-81
[19] Yuichi Ohsita, Shingo Ata, Masayuki Murata (2004). “Detecting Distributed Denial of Service Attacks by analyzing TCP SYN packets statistically”. Revised in IEICE Transactions on Communications 2006.