Java based Network Intrusion Detection Systems using Object Oriented Methodology

Nareshkumar D. Harale* and Dr. B. B. Meshram**
*Department of Computer Science & Engineering,
Sant Gadge Baba Amravati University, Amravati, Maharashtra, India.
nareshkumar.d.harale@gmail.com
**Department of Computer Engineering and Information Technology,
VJTI, Matunga, Mumbai (Maharashtra), India.
bbmeshram@vjti.ac.in

Abstract: Malicious attacks are getting smarter, more widespread and increasingly difficult to detect, and dozens more are added to the menagerie each day. Identifying and classifying the type of malicious program spreading across global networks is a crucial step in developing strategies to contain and eradicate it. A firewall and other preventive security controls are an essential and important part of network security but they don't have the ability to detect hostile intent. Unlike a firewall, an intrusion detection system has the ability to weigh introverted packets and generate an alarm if it detects a packet with hostile potential. This research paper has proposed a novel idea for design and development of Network IDS using object oriented techniques, JPCapDump Packet capturing library and JAVA Language. Unlike Snort Network IDS software; this system is developed using Java language and tested in the typical enterprise environment to detect known attacks and protocol anomalies, it also run multiple OS platform. Through the use of open source tools and replacement hardware a Network IDS can be setup and tested with minimal financial burden. Packet Preprocessing, Packet Analyzer, Intrusion Detection and Alert Generations are the key components of the proposed system. This paper describes the most common attacks used these days to paralyze computer and network resources. It provides network traces of malicious traffic and strategies for providing better countermeasures. Data sets captured at the enterprise network of Calgary are analyzed to classify intrusion attempts and identify security holes in the network using developed Network IDSs.

Keywords: Object Oriented Model, Platform Independent, Network Attacks, Protocol Anomaly, Intrusion Detection Algorithms, Firewall Systems, and Network Security.

Introduction

Intrusion Detection is the process of monitoring computers or networks for unauthorized entrance, activity, or file modification. Network IDS can also be used to monitor new traffic, there by detecting if a system is being targeted by network attack such as denial of service attack. In short host-based IDS examine data held on individual computers that serves as host, while network-based IDS examines data exchanged between computers. An ID generally detects unwanted manipulations to computer systems, mainly through the Internet. The manipulations may take the form of attacks by crackers. An Intrusion Detection System is used to detect many types of malicious network traffic and computer usage that can't be detected by a conventional firewall. This includes network attacks against vulnerable services, data driven attacks on applications, host based attacks such as privilege escalation, unauthorized logins and access to sensitive files.

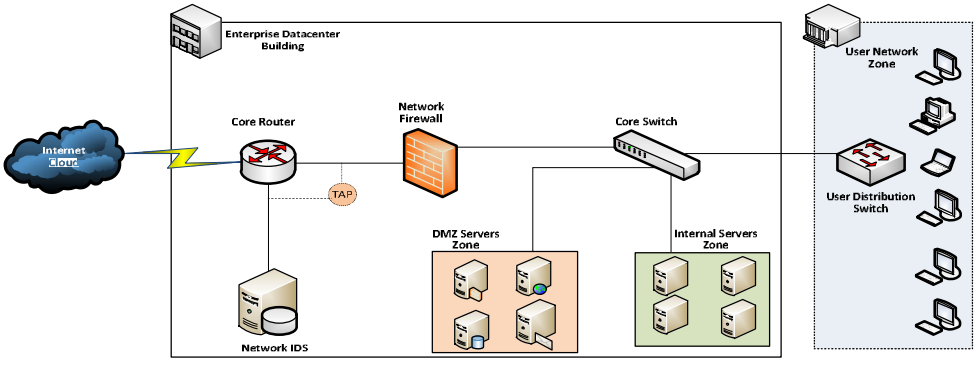


Figure 1: Network IDS Deployment Scenario used for Implementation

Network IDS serves three essential security functions: they monitor, detect, and respond to unauthorized or mutant activity performed by insiders and outsider intrusions. IDS uses policies to define certain event that, if detected issue an alert. In other words, if a particular event is considered to constitute a security incident, an alert will be issued if that event is detected. Certain intrusion detection systems have the capability of sending out alerts, so that the administrator of the Network IDS will receive the notification of possible security incident in the form of the page, email. Many intrusion detection systems not only recognize a particular incident and issue an appropriate alert, they also respond automatically to the event. Such a response might include logging off a user, disabling a user account, and launching of scripts.

Necessity of Network IDSs ***: There are several compelling reasons to acquire and use network IDSs:***

- To prevent problem behaviors by increasing the perceived risk of discovery and punishment for those who would attack or otherwise abuse the system.
- To detects attacks and other security violations those are not prevented by other security measures.
- To detect and deal with the preambles to attacks (commonly experienced as Network probes and other "doorknob ratting" activities).
- To document the existing threat to an organization.
- To act as quality control for security design and administration, especially of large and complex enterprises.
- To provide useful information about intrusions that do take place, allowing improved diagnosis, recovery, and correction of causative factors.

Functions of Network IDS Technologies : There are many types of Network IDS technologies, which are differentiated primarily by the types of events that they can recognize and the methodologies that they use to identify incidents. In addition to monitoring and analyzing events to identify undesirable activity, all types of Network IDS technologies typically perform the following functions:

Recording information related to observed events **:** Information is usually recorded locally, and might also be sent to separate systems such as centralized logging servers, security information and event management (SIEM) solutions, and enterprise management systems.

Notifying security administrators of important observed events: This notification, known as an alert, occurs through any of several methods, including the following: e-mails, pages, messages on the Network IDS user interface, Simple Network Management Protocol (SNMP) traps, syslog messages, and user-defined programs and scripts. A notification message typically includes only basic information regarding an event; administrators need to access the Network IDS for additional information.

Producing Actionable Reports: Reports summarize the monitored events or provide details on particular events of interest. Some Network IDSs are also able to change their security profile when a new threat is detected. For example, a Network IDS might be able to collect more detailed information for a particular session after malicious activity is detected within that session. A Network IDS might also alter the settings for when certain alerts are triggered or what priority should be assigned to subsequent alerts after a particular threat is detected.

Problem Statement

In the mid 1990s, commercial products surfaced for the masses. Two of the most popular Network IDS in the mid 1990s were Wheelgroup's Netranger and Internet Security Systems' RealSecure. Both of these companies started out with Network-based Network IDS. Until now numbers of Open Source Network IDSs are developed and are in use across small scale and enterprise organizations. Commonly used IDSs are Snort, BroIDS, AIDE, and Samhain. Working of all is different but the output is almost same i.e. to prevent the enterprise network or information systems. Different techniques considered while making Network IDSs are accurate capturing the packet stream, prioritizing the detection rules, matching the traffic patterns, generating intrusion alerts for administrator and reducing the false alarm rate. Use of computing network is increased tremendously. Even in an organization all employees may not have good background, there is possibility that someone has bad intensions, and later they may create misuse authorization to create problems hence there is need of security brought out major attentions by enterprises. We can implement security controls like Firewall, Antivirus, Proxy, Identity Management Systems, but if the intruder is the authorized user then what?? For that we need a meticulous strong security solution like Network Intrusion Detection System.

Proposed Systems Objective is to implement a Network based Intrusion Detection Systems which can run on any OS platform, provides fastest capture of packets and filters it according to intrusion or attack rules designed and maintained. System is also expected to generate intrusion alerts when actual network attacks are detected for network security administrator for further analysis and intrusion response in a timely manner.

The Proposed Network IDS mainly emphasized on the following modules as part of the research and development work.

Main Graphical User Interface: User friendly graphical interface will be developed by using JAVA technology for ease of use and intrusion management. It facilitates to start IDS function, stop IDS function, selecting Network Interface for capturing packets, and adding various fields to main window to monitor and alert them for timely response.

Data Capture and Formatting (Preprocessing): It contains following sub modules -

- **Packet capturer:** Capture packets (Network traffic) from specified Network Interface.
- **Protocol Analyzer:** Analyzes protocol, and extracts various data like source-destination address, port number, flags, protocol, type of packet, captured time, length, etc. And format data as per input required by detection engine.

Attack Detection and Alert Generation Module: This module implements various Attack Detection Module, and Alert Generation Module. It contains sub-modules as given below.

- **Attack Detection Module** - Detection module reads data from Data Capture and Formatting Module. It calculate aggressiveness of the source if aggressiveness is considerable than it allow that user otherwise drop that packet and block that IP. Also it try to match various attack signature with collected packet data if it matches, then it drops the packet else allow the packet. This includes modules like TCP packet Attack Detection,IP packet Attack Detection,ICMP packet Attack Detection ,ARP packet attack detection

In these module when a packet arrive then first system checks that is it new packet or not? If packet comes first time then we try to find attack type by matching attack signature with packets captured data.

- **Alerts Message Generator -** When receiving warning messages from Detection Engine, it shows the related information to alert the users**.**

Proposed System Analysis and Design

The analysis and design of the proposed JAVA based Network IDS is described in the below section and implementation is described in the following section. This explains how the system is analyzed to carry out the work for proposed systems. The determining of system requirements is typically an evolutionary process. Analysis increases their level of understanding of an existing system or process gradually; adding additional details by studying major sections independently of others sections. Initially requirement analysis is done for the system. Then the ER Diagram and various Data Flow Diagrams for the system are explained. This chapter deals with the requirement gathering, analysis and design of the Educational decision system.

Level-0 DFD - As shown in the 0th level Data Flow Diagram, the input is given to the Protocol Analyzer for IDS in form of packets which are collected by Jpcap. This input is formatted and stored. The output from the Protocol Analyzer for IDS is detailed analysis of packet and files.

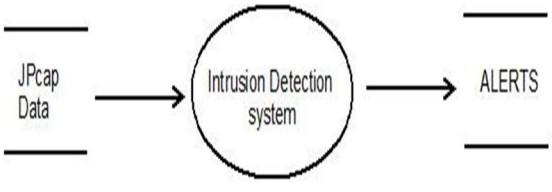


Figure 2: DFD Level-0

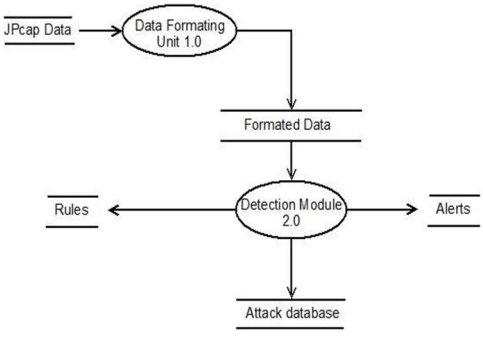


Figure 3: DFD Level-1

Level-1 DFD: As shown in the level-1 data flow diagram includes the following:

- **JPcap Data:** It contains information about packet captured from network.
- **Data Formatting:** The network based approached relies on the JPcapdump data as input, which gives packet header information that we used as the normal data. This data needs to be formatted as grouping records corresponding to one connection. Following this, Content-based, Time-based and Connection-based features were extracted from the data.
- **Formatted Data:** It contains formatted data. Data is extracted according to their protocols. This data is used for further analysis.
- **Detection Module:** Detection module is the heart of the IDS. It takes data from formatted data and checks if there is any intrusion. It uses different techniques for detection.
- **Rules:** There are some sets of known patterns (signatures) of attacks. Incoming packets are checked for similarity in pattern of captured packets header format and these rules.

- **Alerts:** If Detection engine finds some malicious behavior in packet header then it generates appropriate alert so that administrator can take necessary action.
- **Attack Database:** When Detection engine generates alert for presence of intrusion, simultaneously it makes entry in attack database so that administrator can use it for further reference.

Level-2 DFD – Level-2 DFD provides detailed process structure of learning module, path decision making module and acting module as shown below. Data formatting (Preprocessing) module (in DFD level-1) can be expanded into 3 modules as shown below.

- **Protocol matching**: In this module protocol used for the packet transfer is identified. It can be either of the following: TCP-IP, UDP, ARP, and ICMP. (E.g. TCP protocol can be identified by the WIN field in packet header captured by JPcapDump).
- **Field Identification**: Once the protocol is identified, values for the different fields for that protocol are extracted. Each protocol has some specific fields for that particular protocol only.
- **Data entry:** After the values of fields for protocol are identified, these are to be entered into the database in the tables constructed for that particular protocol.

Detection Module can be expanded into three sub modules as following:

- **Decision Making:** Formatted data is given as input to testing module. Firstly decision algorithm is applied to this by using it data is classified.
- **Rule matching:** Classified data is then given to the rule matching block where rules based on predefined signature of attacks are applied to the data.

Attack detection: Those data items which show similarities with attack signature are considered as intrusion. Attack detection module then generates appropriate alert and make entry in attack database.

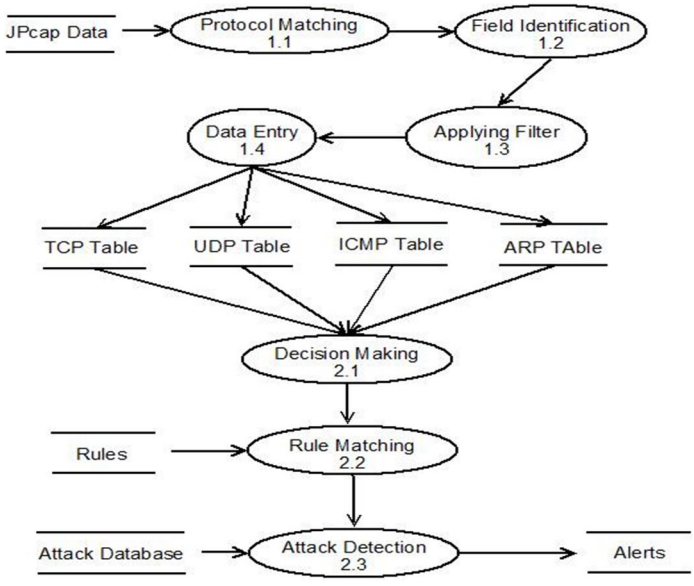


Figure 4: DFD Level-2

Use case Diagram: In software engineering, a use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Use case diagram describes the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The system use case diagram contains user and IDS as the main actor.

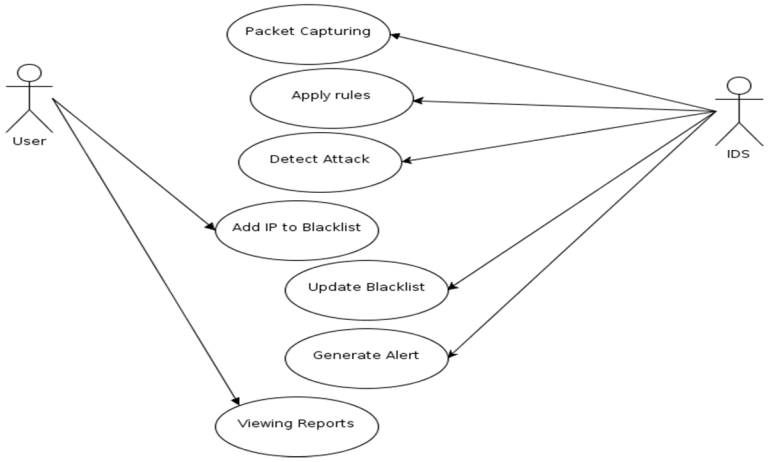


Figure 5: Use Case Diagrams

Class Diagram

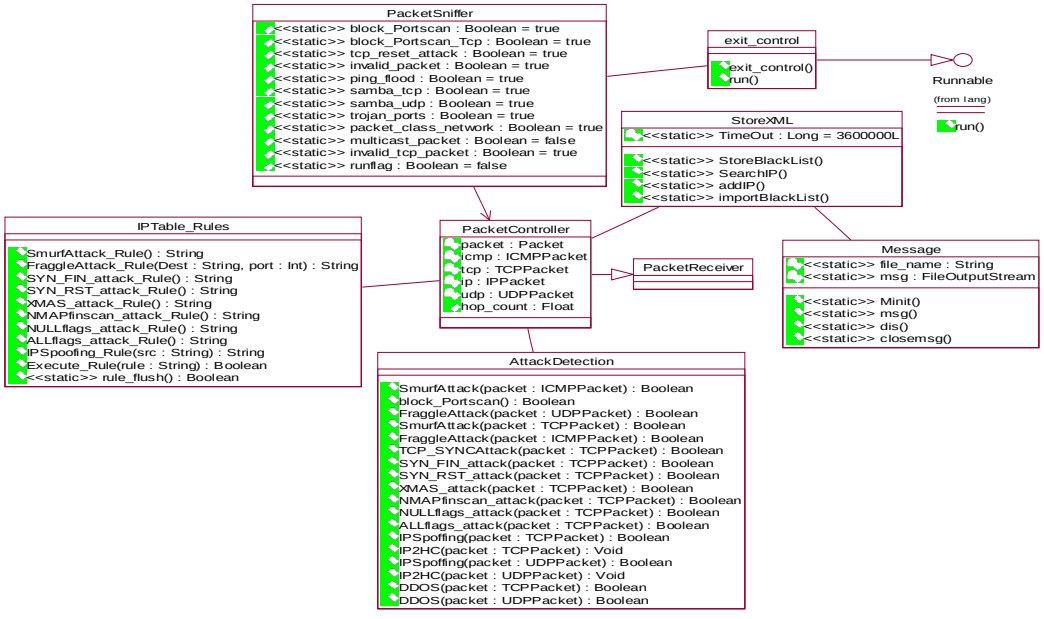


Figure 6: Proposed System's Class Diagram

Proposed Network IDS's Architecture

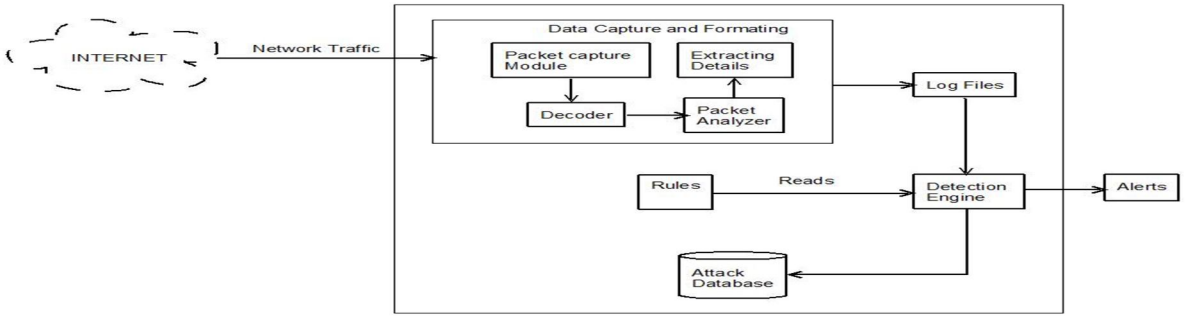


Figure 7: Proposed Network IDS Block Diagram

Figure shows deployment architecture diagram of proposed Network IDS. It consists of building blocks:

Data Capture and Essential Formatting: Packet Sniffer JpcapDump collects packet headers of network traffic data coming from Internet or Internal Networks Zone as shown in the Figure 3.1. Data captured is decoded and analyzed accurately. Captured data is analyzed according to protocol information. Various header field details are extracted from captured data for normal or abnormal values to classify the traffic as normal or intrusion.

Log Files: Extracted data (details) is classified into various fields and are stored into log files.

Intrusion Detection Engine: Many computer or network attacks have predefined attack signatures to which new captured data would be compared and classified. These attack signatures can be used to identify particular type of attacks before initiate the intrusion response activity. We used predefined intrusion signatures or rules and compared capture data packet header fields with them. If packet data pattern matches the IDS predefined signatures then system declares it as Intrusion and generates an alert to network security administrator.

Network Attack Database: Attack database also contains tables for different protocols. The entries from log which are declared as attack are stored in attack database. This attack database can be referred in future for drawing the conclusions.

The proposed block diagram is used for the design of software architecture of Network IDS as given below using Object Oriented Analysis and Design methodology:

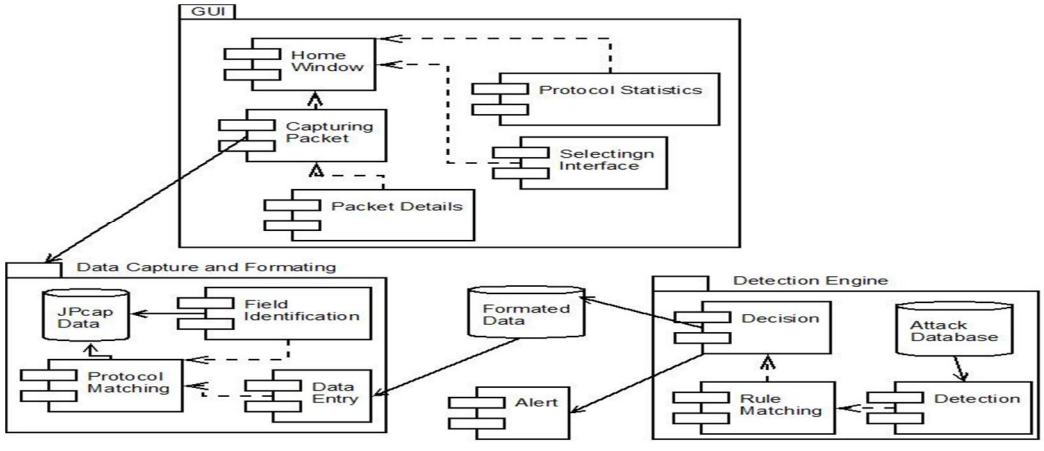


Figure 11: Proposed Network IDS – Software Architecture Diagram

Proposed Network Intrusion Detection System Algorithms

In this section, we have designed the attack detection algorithms for the implementation of the proposed Network IDS.

a. Smurf Attack Detection Algorithm *b.*	*B.Fragile Attack Detection Algorithm*
Input: Packet. **Output:** Smurf attack is detected or not. *Smurf attack (Packet P)* *IF packet type is ICMP {* *IF packet type is ECHO_REQUEST and destination address is BROADCAST ADDRESS {PRINT "smurf attack detected"* *RETURN TRUE }* *ELSE RETURN FALS }*	**Input:** Packet. **Output:** Fragile attack is detected or not. *Fragile attack(Packet P)* *IF packet type is UDP {* *IF DESTINATION_PORT is 7 or 19 and destination address is BROADCAST ADDRESS { PRINT "fragile attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE }*

C.Portsweep Attack Detection Algorithm	*D.Land Attack Detection Algorithm*
Input: Packet. **Output:** Portsweep attack is detected or not. *Portsweep attack(Packet P)* *IF packet type is TCP {* *IF FIN flag is set and ACK flag is reset{ PRINT "portsweep attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE}*	**Input:** Packet. **Output:** Land attack is detected or not. *Land attack (Packet P)* *IF packet type is TCP {* *IF SYN flag is set and source address is equal to destination address {PRINT "land attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE }*

e.SYN FIN Attack Detection Algorithm	*f.SYN RST Attack Detection Algorithm*
Input: Packet. **Output:** SYN FIN attack is detected or not. *SYN_FIN attack (Packet P)* *IF packet type is TCP {* *IF SYN flag and FIN flag is set {* *PRINT "tcpsyn_fin attack detected"* *RETURN TRUE}* *ELSE RETURN FALSE }*	**Input:** Packet. **Output:** SYN RST attack is detected or not. *SYN_RST attack(Packet P)* *IF packet type is TCP {* *IF SYN flag and RST flag is set {* *PRINT "tcpsyn_rst attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE }*

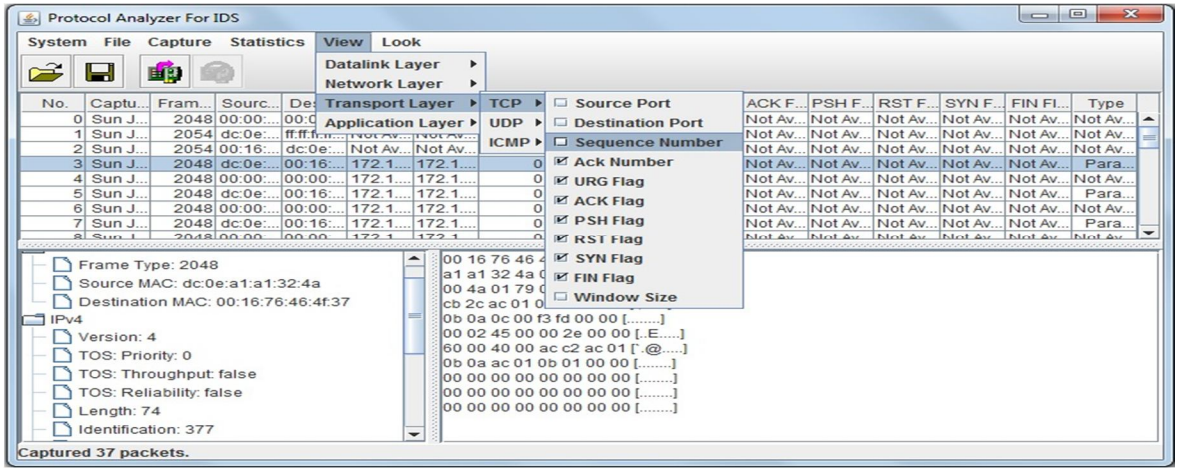
g.XMAS Attack Detection Algorithm	*h. NMAP fin scan Attack Detection Algorithm*
Input: Packet. **Output:** XMAS attack is detected or not. *XMAS attack(Packet P)* *IF packet type is TCP {* *IF SYN flag and RST flag and ACK flag are set and PSH flag and FIN flag and URG flag is reset { PRINT "tcpxmas attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE}*	**Input:** Packet. **Output:** NMAP FIN SCAN attack is detected or not. *NMAPfinScan attack(Packet P)* *IF packet type is TCP {* *IF FIN flag is set and RST flag and SYN flag and PSH flag and ACK flag and URG flag is reset {PRINT "tcpnmapfinscan attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE}*

i.NULL flags Attack Detection Algorithm	*j.ALL flags Attack Detection Algorithm*
Input: Packet. **Output:** NULL flags attack is detected or not. *NULL flags attack(Packet P)* *IF packet type is TCP {* *IF FIN flag and RST flag and SYN flag and PSH flag and ACK flag and URG flag is reset* *{ PRINT "tcpnullflags attack detected" RETURN TRUE } ELSE RETURN FALSE}*	**Input:** Packet. **Output:** ALL flags attack is detected or not. *ALL flags attack(Packet P)* *IF packet type is TCP {* *IF FIN flag and RST flag and SYN flag and PSH flag and ACK flag and URG flag is set { PRINT "tcp all flags attack detected" RETURN TRUE }* *ELSE RETURN FALSE} }*

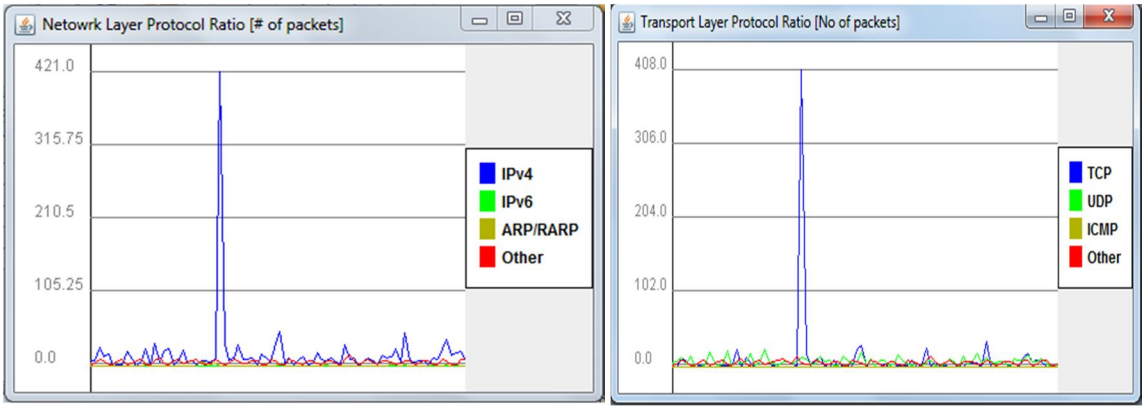
k. DDOS Attack Detection Algorithm	*l. Teardrop Attack Detection Algorithm*
Variable: P Pass test value of IP address, F Fail test Value of IP address. Threshold value of pass and fail test is set by user. R_N $_N$ew arrival rate of packet per minute from that IP address R_O Old arrival rate of packet per minute from that IP address **Input:** Packet. **Output:** DDOS attack is detected or not. *DDOS (Packet P) {Fetch pass test of IP (P)* *If (P > threshold value of pass test) then* *Send it for checking IP spoofing attack. Else Fetch fail test of IP (F)* *If (F > threshold value of fail test) then Drop that packet and store that packet to black list* *Else Check new rate R_N < old rate R_O/2 If yes then Send it for checking IP spoofing attack. Else Drop that packet and store that packet to black list }*	**Input:** Packet. **Output:** Teardrop attack is detected or not. *Teardrop attack(Packet P)* *IF packet type is IP {* *IF More fragment flag is set and fragmentation offset is zero { PRINT "ipteardrop attack detected"* *RETURN TRUE }* *ELSE RETURN FALSE}*

Proposed Implementation and Results

This section deals with the setting up the environment for implementing and testing of the proposed Network IDS system. Sample User Interface screen shows how the user will interact with the system and the data entry forms required gathering data for the system, it is demonstrated though limited screens due to paper pages .The result of operations from Protocol Analyzer for Network IDS is also shown in this section. *Software Runtime Screenshots***:** This section gives the screenshots of the system. Starting from the home window, we have given all the output screens also taken in different conditions.



Screen 3.3: Selecting protocol fields which we would want to be monitored



Screen 3.5: Network and Transport Protocol Layer Statistics

Conclusion

This paper has presented a research and development done for design and development of the Network IDS using OOPs approach. The paper covered the approach used intrusion detection techniques to secure the information systems present in the enterprise networks that are directly or indirectly connected to the Internet. It depends on the Network Administrator to make sure that his Network is out of danger. This software does not completely shield Network from Intruders or new attacks, but it helps the Network Security Administrator to trace down malicious guys on the Internet or Intranet whose purpose is to bring the network to a breach point and make it vulnerable to unauthorized or intended attacks. It facilitates blocking the traffic from certain IP address depending blacklist maintain by Network IDS. The Network IDS system is designed in such a way that it can be reused very easily. A platform is set very clearly in order that some known attacks can be identified and alerted. The Network IDS is developed completely in Java programming language hence proposed system is platform independent, yet it has been tested only on Windows7. Finally It can be employed and tested on various other OS platforms which would equally perform satisfy the security requirements and pre-requisites for the Network IDSs. We have also implemented intrusion detection based on protocol analysis to detect various well known attacks. Intrusion detection approach that we have used is Hybrid which contains Network-based Intrusion Detection and Protocol-based Intrusion Detection. So our system captures Network data using JPcapdump and detects attacks based on captured packets header information according to various protocols.

References

[1] ZongpuJia, Shufen Liu, Guowei Wang, *"Research and Design of NIDS Based on Linux Firewall"* 1st International Symposium on Pervasive Computing and Applications" , 2006.

[2] Yuebin Bai, Hidetsune Kobayashi, "Intrusion Detection System: Technology And Developement", Proceedings of the17 th International Conference on Advanced Information Networking and Applications (AINA'03),IEEE

[3] W. Li, "A Genetic Algorithm Approach to Network Intrusion Detection", SANS Institute, USA, 2004.

[4] Zhenwei Yu, Tsai, J.J.P. and Weigert, "An adaptive automatically tuning intrusion detection system", ACM Transactions on Autonomous and Adaptive Systems(TAAS), Vol3, Issue 3, August 2008
[5] Kapil Kumar Gupta, BaikunthNath, Kodagali Ramamohanarao, and Ashraf Kazi. Attacking Confidentiality: An Agent Based Approach. In Proceedings of IEEE International Conference on Intelligence and Security Informatics, pages 285–296. Lecture Notes in Computer Science, Springer Verlag, Vol (3975), 2006
[6] S. AXELSSON, the Base-Rate Fallacy and the Difficulty of Intrusion Detection, ACM Transactions on Information and System Security 3 (2000), no. 3, 186–205.
[7] Packet Pre-filtering for Network Intrusion Detection Ioannis Sourdis, Vasilis Dimopoulos, Computer Engineering Lab. Electrical Engineering Dept,TU Delft, The Netherlands, Dionisios Pnevmatikatos,Stamatis Vassiliadis Microprocessor and Hardware Laboratory Electronic and Computer Engineering Dept,
[8] Design and Implementation of IDS Using Snort, Entropy and Alert Ranking System Shiv Kumar, R.C.Joshi Department of Electronics & Computer Engineering, Indian Institute of Technology, Roorkee, India-247667
[9] Integrating Artificial Intelligence into Snort IDS Xianjin Fang, Lingbing Liu, School of Computer Science and Engineering, Anhui University of Science and Technology, Huainan, China.
[10] RAPID: Reputation based Approach for Improving Intrusion Detection Effectiveness Ashley Thomas SecureWorks, Inc Atlanta, USA.
[11] Evaluating Intrusion Detection Systems in High Speed Networks Faeiz Alserhani, Monis Akhlaq, Irfan U Awan, , John Mellor, Andrea J Cullen Informatics Research Institute, University of Bradford, Bradford, BD7 1DP, United Kingdom and Pravin Mirchandani Syphan Technologies.
[12] Efficient Snort Rule Generation using Evolutionary computing for Network Intrusion Detection by Raghavan Muthuregunathan, Siddharth S, Srivathsan R, Rajesh SR.
[13] Research and implementation on Snort-based Hybrid intrusion detection system Yu-Xin Ding, Min Xiao, Ai-Wu Liu Key Laboratory of Network Oriented Intelligent Computation Department of Computer Sciences and Technology, Harbin Institute of Technology Shenzhen Graduate School,China.
[14] Framework of Intrusion Detection System via Snort Application on Campus Network Environment Mohd Nazri Ismail Department of MIIT, University of Kuala Lumpur (UniKL), Malaysia mnazrii@miit.unikl.edu.my Mohd Taha Ismail Department of MIIT, University of Kuala Lumpur (UniKL), Malaysia.
[15] Christos Douligeries, Aikaterini Mitrokosta, "DDoSClassification and attacks and defense mechanisms: classification and state of the art" The International Journal of Computer and Telecommunications Networking, vol. 44, issue 5, pp:643-666, 2004.
[16] Improvement on Rules Matching Algorithm of Snort Based on Dynamic Adjustment Kuo Zhao, Jianfeng Chu, Xilong Che, Lin Lin, Liang Hu Department of Computer Science and Technology Jilin University Changchun 130012, China.
[17] Kumar, S., "Classification and detection of Computer Intrusion", PhD Thesis, 1995, Prude Univ., West Lafayette, IN.
[18] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN flooding attacks", Proceedings of IEEE INFOCOM.
[19] Sean Whalen, Matt Bishop, Sophie Engle, "Protocol Vulnerability Analysis (draft)".
[20] Oryspayuli O. D. (2006), "What intrusion detection approaches work well if only TCP/IP packet header information is available?" , Master thesis, University of Twente, The Netherlands.
[21] YacineDjemaiel, "Tracing, Detecting and Tolerating Attacks in Communication Networks",Ph.D. thesis, Communication Networks and Security Research Unit CNAS, SUP'COM
[22] K. Hwang, M. Cai, Y. Chen, and M. Qin, Hybrid intrusion detection with weighted signature generation over anomalous internet episodes, IEEE Transactions on Dependable and Secure Computing (2007), 41–55.S.
[23] SANS Institute - Intrusion Detection FAQ. Last accessed: November 30, 2008. http://www.sans.org/resources/idfaq/.
[24] Intrusion Detection & Prevention by Carl Endorf, Eugene Schultz and Jim Mellander McGraw -Hill© 2004.