IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Integrated And Constrained Based Approach For Network Intrusion Detection System

Kamna Singh
Assistant Professor
Computer Science & Engineering Department
Ajay Kumar Garg Engineering College, Ghaziabad,India

Abstract: Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. Intrusion prevention is the process of performing intrusion detection and attempting to stop detected possible incidents. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, attempting to stop them, and reporting them to security administrators. In addition, organizations use NIDSs for other purposes, such as identifying problems with security policies, documenting existing threats, and deterring individuals from violating security policies. NIDSs have become a necessary addition to the security infrastructure of nearly every organization.

Keywords: NIDS, IDPS

I.Introduction: Intrusion detection systems (IDSs) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for signs of security problems. As network attacks have increased in number and severity over the past few years, intrusion detection systems have become a necessary addition to the security infrastructure of most organizations. Security is a big issue for all networks in today's enterprise environment. Hackers and intruders have made many successful attempts to bring down high-profile company networks and web services. Many methods have been developed to secure the network infrastructure and communication over the Internet, among them the use of firewalls, encryption, and virtual private networks. Intrusion detection is a relatively new addition to such techniques. Intrusion detection methods started appearing in the last few years. Using intrusion detection methods, one can collect and use information from known types of attacks and find out if someone is trying to attack one's network or particular hosts. The information collected this way can be used to harden network security, as well as for legal purposes.

II.Literature Review:

This research paper has discussed the various issues that needs to be taken care off during networks intrusion detection. Intrusion detection systems (IDSs) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analysing them for signs of security problems. As network attacks have increased in number and severity over the past few years, intrusion detection systems have become a necessary addition to the security infrastructure of most organizations. This guidance document is intended as a primer in intrusion detection, developed for those who need to understand what security goals intrusion detection mechanisms serve, how to select and configure intrusion detection systems for their specific system and network environments, how to manage the output of intrusion detection systems, and how to integrate intrusion detection functions with the rest of the organizational security infrastructure.

III.Proposed Solution:

Intrusion detection allows organizations to protect their systems from the threats that come with increasing network connectivity and reliance on information systems. Given the level and nature of modern network security threats, the question for security professionals should not be whether to use intrusion detection, but which intrusion detection features and capabilities to use. IDSs have gained acceptance as a necessary addition to every organization's security infrastructure. Despite the documented contributions intrusion detection technologies make to system security, in many organizations one must still justify the acquisition of IDSs. There are several compelling reasons to acquire and use IDSs.

3.1Features of the proposed System:

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

There are different types of intrusion detection systems.

i. Host Based intrusion detection system
ii. Network Based intrusion detection system
iii. Protocol Based Intrusion Detection System

3.2 Host based vs. network Intrusion and Prevention System

HIPS can handle all types of encrypted networks and can analyze all code. NIPS does not use processors and memory on computer hosts but uses its own CPU and Memory.

3.3 Intrusion Detection and Prevention Systems (IDPS)

This paper discusses about Intrusion detection and prevention methods. This is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. Incidents have many causes, such as malware (e.g., worms, spyware), attackers gaining unauthorized access to systems from the Internet, and authorized users of systems who misuse their privileges or attempt to gain additional privileges for which they are not authorized. Although many incidents are malicious in nature, many others are not; for example, a person might mistype the address of a computer and accidentally attempt to connect to a different system without authorization. An intrusion detection system (IDS) is software that automates the intrusion detection process. An intrusion prevention system (IPS) is software that has all the capabilities of an intrusion detection system and can also attempt to stop possible incidents. This section provides an overview of IDS and IPS technologies as a foundation for the rest of the publication. It first explains how IDS and IPS technologies can be used. Next, it describes the key functions that IDS and IPS technologies perform and the detection methodologies that they use. Finally, it provides an overview of the major classes of IDS and IPS technologies.

3.4 Automatic Analysis of Firewall and Network Intrusion Detection System Configurations

Given a network that deploys multiple firewalls and network intrusion detection systems (NIDSs), ensuring that these security components are correctly configured is a challenging problem. Although models have been developed to reason independently about the effectiveness of firewalls and NIDSs, there is no common framework to analyze their interaction. This paper presents an integrated, constraint-based approach for modeling and reasoning about these configurations. Our approach considers the dependencies among the two types of components and can reason automatically about their combined behaviour. We have developed a tool for the specification and verification of networks that include multiple firewalls and NIDSs, based on this approach. This tool can also be used to automatically generate NIDS configurations that are optimal relative to a given cost function.

3.5 Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems

This article presents a survey of denial of service attacks and the methods that have been proposed for defense against these attacks. In this survey, we analyze the design decisions in the Internet that have created the potential for denial of service attacks.We review the state-of-art mechanisms for defending against denial of service attacks, compare the strengths and weaknesses of each proposal, and discuss potential countermeasures against each defense mechanism.We conclude by highlighting opportunities for an integrated solution to solve the problem of distributed denial of service attacks.

3.7 Network Based Intrusion Detection:

Network based intrusion detection systems are those that monitor traffic on the entire network segment. A network interface card (NIC) can operate in one of two modes, these being:

3.7.1 **Normal mode**, where packets which are destined for the computer (as determined by the Ethernet or MAC address of the packet) are relayed through to the host system.

3.7.2**Promiscuous mode**, where all packets that are seen on the Ethernet are relayed to the host system. A network card can normally be switched from normal mode to promiscuous mode, and vice-versa, by using a low-level function of the operating system to talk directly to the network card to make that change. Network based intrusion detection systems normally require that a network interface card is in promiscuous mode.

3.8Packet Sniffers and Network Monitors

Packet Sniffers and Network Monitors were originally designed to aid in the process of monitoring the traffic on an Ethernet network. These basically capture all packets that they see on the network. Once the packets are captured, a number of possibilities arise:

Packets can be counted. Counting the packets that come past, and adding together their total size over a period of time (including overheads such as packet headers) gives a pretty good indication of how heavily loaded the network is.

•Packets can be examined in detail. For example, one might want to capture a set of packets arriving at a web server to diagnose some problem with the server.

3.8.1 Packet Sniffing and Promiscuous Mode

All packet sniffers will require that a network interface is in promiscuous mode. Only in promiscuous mode will every packet received by the NIC be passed up to the packet sniffer itself. The packet sniffer normally requires administrative privileges on the machine being used as a packet sniffer, so that the hardware of the network card can be manipulated to be in promiscuous mode.

Here is an example of some of the types of intrusion detection that the Network Intrusion Detection System can perform:

i. Examine packets that pass through the network.

ii.For legitimate packets, allow them to pass (perhaps recording them for future analysis).

iii.Where a packet endangers the security or integrity of a target system, stop transmission of the packet by sending TCP "connection closed" or ICMP "port unreachable" messages to both the target system and the system sending the packet. When unwanted activity is detected, network based intrusion detection can take action, including interfering with future traffic from the intruder, or reconfiguring a nearby firewall to block all traffic coming from the intruder's computer or network.

.8.2. Host Based Intrusion Detection

Once a network packet has arrived at the host that it was intended for, there is still available a third line of defence behind the firewall and network monitor. This is called "host based intrusion detection"

The two main types of host based intrusion detection are:

3.8.2.1 Network monitors. These monitor incoming network connections to the host, and attempt to determine whether any of these connections represent a threat. Network connections that represent some kind of intrusion attempt are acted on. Note that this is different to network based intrusion detection, as it only looks at network traffic coming to the host it is running on, and not all traffic passing the network. For this reason it does not require promiscuous mode on the network interface.

• Host monitors. These monitor files, file systems, logs, or other parts of the host itself to look for particular types of suspicious activity that might represent an intrusion attempt (or a successful intrusion). Systems administration staff can then be notified about any problems that are found.

3.8.3 Monitoring Incoming Connections

It is possible on most hosts to monitor packets that attempt to access the host before those packets are passed onto the networking layer of the host itself. This mechanism attempts to protect a host by intercepting packets that arrive for the host before they can do any damage. Some of the actions that can be taken include:

• Detect incoming connection attempts to TCP or UDP ports that are unauthorized, such as attempts to connect to ports where there are no services. This is often indicative of a possible cracker who tries to find weaknesses of the system.

• Detect incoming port scans. Detecting the scans and making a log of "How may packets in how much time", gives a useful indication about network threats.

Detect different DoS attacks like Syn Flooding and Ping of Death .If there is an attempt to bombard the system using these attacks, proper action like switching off the network services or switching off the port on which the attack is detected can be taken.

• Keeps a check on packets coming from a particular host.

Proposed Architecture:

The system has been logically divided into multiple components. These components work together to detect particular attacks and to generate output in a required format from the detection system. It consists of the following major components:

• Rule Parser
• Packet Capture Module
• Packet Analyzer
• Packet Logger

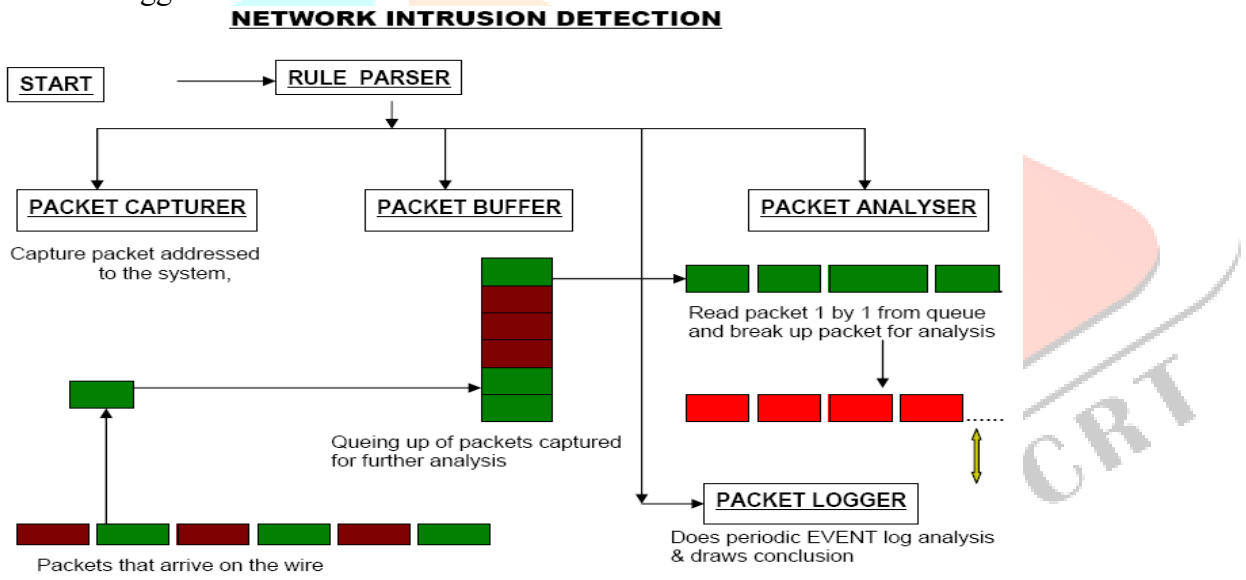

Figure :1

Figure above shows how these components are arranged. Any data packet coming from the Internet is given to the main module which sends it to a corresponding thread which is based on packet information. On its way towards the output modules, it is either dropped, logged or an alert is generated. Detailed explanation of each module follows.

3.9 Rule Parser

Like viruses, most intruder activity has some sort of signature. Information about these signatures has been used to create rules. These known attacks are also used as signatures to find out if someone is trying to exploit them. These signatures may be present in the header parts of a packet or in the payload. This detection system is based on rules. These rules in turn are based on intruder signatures. The rules can be used to check various parts of a data packet. A rule may be used to generate an alert message, log a message.

Rules are written in an easy to understand syntax. Most of the rules are written in a single line. However one can also extend rules to multiple lines. Here is an example rule.

alert ip any any -> any any (msg: "IP Packet detected";)

The rule will generate an alert message for *every* captured IP packet.

Below is a brief explanation of different words used in this rule:

• The word "alert" shows that this rule will generate an alert message when the criteria are met for a captured packet. The criteria are defined by the words that follow.

• The "ip" part shows that this rule will be applied on all IP packets.

• The first “any” is used for source IP address and shows that the rule will be applied to all packets. Instead of any one can give any IP e.g 172.19.1.33.
• The second “any” is used for the port number. Since port numbers are irrelevant at the IP layer, the rule will be applied to all packets. Instead of any one can give any port no e.g. 80
• The -> sign shows the direction of the packet.
• The third “any” is used for destination IP address and shows that the rule will be applied to all packets irrespective of destination IP address.
• The fourth “any” is used for destination port. Again it is irrelevant because this rule is for IP packets and port numbers are irrelevant.
The last part is the rule options and contains a message that will be logged along with the alert.
A second example of rule can be:
alert icmp any any -> any any (msg: "ICMP Packet found";)
It generates alerts for all captured ICMP packets.

Structure of a Rule:
Basic structure of the rules is:

Rule Header	Rule options

Fig-2 Structure of rule

The rule header contains information about what action a rule takes. It also contains criteria for matching a rule against data packets. The options part usually contains an alert message and information about which part of the packet should be used to generate the alert message. The options part contains additional criteria for matching a rule against data packets. A rule may detect one type or multiple types of intrusion activity.
The general structure of a rule is:

Action	Protocol	Address	Port	Direction

Figure: General structure of a rule

The action part of the rule determines the type of action taken when criteria are met and a rule is exactly matched against a data packet. Typical actions are generating an alert or log message or invoking another rule. The protocol part is used to apply the rule on packets for a particular protocol only. This is the first criterion mentioned in the rule. Some examples of protocols used are IP, ICMP, UDP etc. The address parts define source and destination addresses. Addresses may be a single host, multiple hosts or network addresses. One can also use these parts to exclude some addresses from a complete network. Note that there are two address fields in the rule. Source and destination addresses are determined based on direction field. As an example, if the direction field is “->”, the Address on the left side is source and the Address on the right side is destination. In case of TCP or UDP protocol, the port parts determine the source and destination ports of a packet on which the rule is applied. In case of network layer protocols like IP and ICMP, port numbers have no significance. The direction part of the rule actually determines which address and port number is used as source and which as destination.

3.9.1Using Variables in Rules
In the configuration file, one can use variables. This is a very convenient way of creating rules. For example, one can define a variable HOME_NET in the configuration file.
HOME_NET 172.19.1.0/24
Later on this variable HOME_NET can be used in the rules, for e.g.
alert ip any any -> $HOME_NET any (msg: “Loose source routing attempt”;)
Using variables makes it very convenient to adapt the configuration file and rules to any environment. For example, there is no need to modify all rules while copying rules from one network to another; only modification of a single variable is required.
Using the any Keyword The any keyword can also be a variable. It matches to everything, just as it does in rules (such as addresses and port numbers). For example, if there is a need to catch packets regardless of their source, a variable EXTERNAL_NET can be defined . EXTERNAL_NET any
There are many variables defined in the variables.conf file that can be modified and new variables added to this file. After writing the rules in the above fashion the work of rule parser is to convert these rules into a usable format i.e take out the important information out of the rules and fill the structures used in different programs to make the capturing and analyzing the packet easier. The rule parser divides the rules in filterstrings on the basis of the main part of the rules.

For e.g for the given rules:
detectscan tcp $EXTERNAL_NET any -> $LO_NET any (flags:F!PU;)
detectscan tcp $EXTERNAL_NET any -> $LO_NET any (flags:!FSPURA;)
Rule parser parses the above rules and convert into just 1 filterstring :
tcp $EXTERNAL_NET any -> $LO_NET any
Also the keywords are replaced by proper values from the variables.conf file .

3.9.2 Packet Capture Module:

A precursor to this module is the main program (main.c) which makes children on thebasis of filterstrings generated by the Rule Parser. Number of children generated is equal to the number of filterstrings created. According to these filterstrings the Packet capture module which is a child of main program captures the packets of a particular type depending on the filter string.

Let's understand why there is a need to make children to the main program. Basic aim of NIDS is to capture and analyze packets efficiently. Since, the packets come with a trememndous speed, of the order of hundreds in a second. There is an urgent need to classify these packets into different groups based on the information they come with. This information has been classified into filterstrings in this NIDS. A child is generated for each filterstring such that all children work as in a parallel processing architecture and reduce the load from the main program .A shared memory has been allotted to all the children in the form of a queue. Each child is catching packets of a particular signature , picking up the packets one by one and sending it to the packet analysis phase .The process id of all the children are stored in a file "child.txt " which may be used for future references . e.g. for Killing a child in order to keep the buffer free and stop the system from crashing down weighing on the load because of the thousands of packets coming every now and then.

3.9.3Packet Analyzer

This phase works in tandem with the packet capturing phase. This program comes into action when the packet capture program gives it the information about the captured packet. Once it gets the packet, the packet is matched for all the rules and if any rule matches with the information of the packet then the proper action as indicated by the rule is taken. Action can be alert or log or anything else. Shared memory that is provided in the packet capturing phase comes in handy in this phase too. The process id's of the children are sent to the process packet module with the help of which it picks up the corresponding packet. Now , all the packet information like IP, Protocol, TTL, length of packets, flags(SYN,FIN,RST,URG …) etc are matched against the rules which are present in the rules file . The parser module provides a structure of information regarding different rules corresponding to a filter string. This structure is matched with the packet information and if any match found then the corresponding action present in the rule is taken .Suppose that the NIDS wants to detect a "SYN scan", this is the phase which compares the "SYN" flag of the incoming packet with the options provided in the rule, if a rule has been added then a match will occur and the corresponding action "detectscan" is taken.

Rule that was used is: detectscan tcp $EXTERNAL_NET any -> $LO_NET any (flags:S;)

3.9.4 Logging and Alerting System

Depending upon what the detection engine finds inside a packet, the packet may be used to log the activity or generate an alert. Logs are kept in simple text files, tcpdump – style files or some other form. All of the log files are stored under /LOGS/ folder by default. One can use command line options to modify the location of generating logs and alerts. The logging of packets is done to enhance the further usage of the information stored in the packets. For e.g in order to detect a synscan one can log the packets and according to their timestamps one can check the formula by which scan can be detected. A sample formula is:

P packets come in T time. If the packets logged follow this criteria and have SYN flag on then one can be sure that it was a "SYN SCAN" attempt. All the information including packet header and message is kept in the log file .A Sample Log is as below:

Packet 1 :(in file 172.19.1.33)
Packet captured length: 54 received at Wed Mar 21 01:03:59 2004
Ethernet type hex:800 is an IP packet 0:5:5d:4a:37:26->
0:8:a1:50:d1:4c 172.19.1.34-->172.19.1.33 IP Header Length: 5 Version:4
Packet Length:40 Offset:0 TTL:64 Protocol:6 Chksum:37596 A R Port: 16 -
> 55577 Seq : 0 Ack -1139256936 Window Size : 0 Tcp Header Length : 20
ALERT : message is coming
Packet 2: (in main.log file)
74 Thu Mar 19 18:20:10 2004
IP 0:8:a1:50:d1:4c-> 0:5:5d:4a:37:26 172.19.1.33-->172.19.1.34 ICMP

Echo Request ALERT : PING OF DEATH ATTEMPT
Logs are made on the basis of of IP they are generated from .A separate file is maintained for all IP's in order to keep track for the culprit amongst them .
IDS
The main task of intrusion detection systems is defense of a computer system by detecting an attack and possibly repelling it. Detecting hostile attacks depends on the number and type of appropriate actions . Intrusion prevention requires a well-selected combination of "baiting and trapping" aimed at both investigations of threats. Diverting the intruder's attention from protected resources is another task. Both the real system and a possible trap system are constantly monitored. Data generated by intrusion detection systems is carefully examined (this is the main task of each IDS) for detection of possible attacks (intrusions).

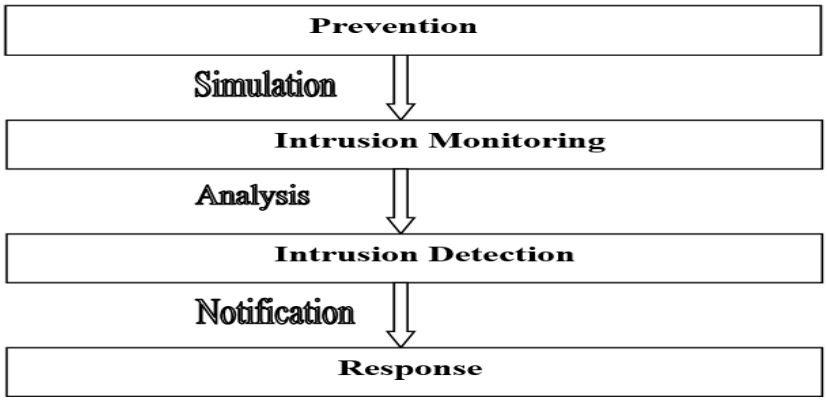

Figure 3: Intrusion system activities

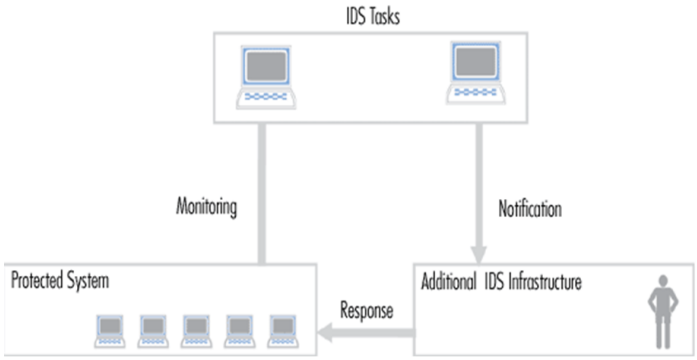

Fig 4: Intrusion detection System Infrastructure

Once an intrusion has been detected, IDS issues alerts notifying administrators of this fact. The next step is undertaken either by the administrators or the IDS itself, by taking advantage of additional countermeasures (specific block functions to terminate sessions, backup systems, routing connections to a system trap, legal infrastructure etc.) – following the organization's security policy . An IDS is an element of the security policy. An intrusion detection systems always has its core element - a sensor (an analysis engine) that is responsible for detecting intrusions. This sensor contains decision-making mechanisms regarding intrusions. Sensors receive raw data from three major information sources own IDS knowledge base, syslog and audit trails. The syslog may include, for example, configuration of file system, user authorizations etc. This information creates the basis for a further decision-making process.

Conclusion: The spaghetti of network attacks, viruses, hacking, cracking is luring enough for pursuing a project in the area. The thought of intruding into others' system and at the same time preventing incursions on one's system, is quite enchanting. Coupled with this is the fact that the paper helps one acquire sound network programming skills and experience of working on large applications in C language.. The paper helped gain valuable experience of developing large applications as a team. In particular the vast capabilities Sockets API, the Packet Capture library and 'QT' were used for efficient implementation. After being duly tested, one can conclude that this paper can be used for networks with continuous attacks or virus threats but with proper rules incorporation.

IV.**Future Scope:** Implementation of some other port scanning techniques - Fragmentation Scanning , TCP reverse ident scanning, Vanilla scan .
2. Implementation of Counter Attack Mechanism.
3. Implementation of some other attacking techniques which will make the checking of NIDS more rigorous – ICMP Destination Unreachable, brkill attack, session hijacking.
4. In case of large traffic, NIDS may take long time to detect the attacks, so optimization to such a level that NIDS is independent of bandwidth is desired.
5. To analyze encrypted information which is not yet a feature of any NIDS

References:

[1] Syed Masum Emran, Nong Ye , 'Network intrusions: A System Architecture for Computer Intrusion Detection , August 2001 Information-Knowledge-Systems Management, Volume 2 Issue 3 Publisher: IOS Press

[2] Vinod Yegneswaran, Paul Barford, Johannes Ullrich,' Network Security: intrusion detection And Prevention Systems ', June 2003 Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '03, Volume 31 Issue 1 Publisher: ACM Press

[3] Marco de Vivo, Eddy Carrasco, Germinal Isern, Gabriela O. de Vivo ,' A review of port scanning techniques', April 1999, ACM SIGCOMM Computer Communication Review, Volume 29 Issue 2 ,Publisher: ACM Press

[4] Tomás E. Uribe, Steven Cheung," Security & analysis II: Automatic analysis of firewall and network intrusion detection system configurations ",October 2004, Proceedings of the 2004 ACM workshop on Formal methods in security engineering FMSE '04 Publisher: ACM Press

[5] Tao Peng, Christopher Leckie, And Kotagiri Ramamohanarao,'Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems' ,Department of Computer Science and Software Engineering, The University of Melbourne, Australia