NETWORK FORENSICS SYSTEM FOR ICMP ATTACKS USING REAL TIME APPROACH

Madhuri N. Turup and B. M. Patil
P.G.Dept., M.B.E.S's College of Engineering, Ambajogai, Maharashtra, India

ABSTRACT

Network forensics is sub-branch of Digital Forensics where network traces are collected from victim system for analysis and producing legal evidences which are helpful for prosecuting the attacker.[1] In this paper we are proposing network forensics system for collecting and analysing ICMP packets for recognizing an attack, at real time, instead of storing packets and analyzing them later on. The proposed system addresses challenges like storage of network traffic as well as time required for investigation of network traces so that attack is detected as early as possible. In this system we are referencing attacks on ICMP protocol. Fields of ICMP header are checked and if values are found varying than expected then packet is marked as suspicious. Such suspicious packets are checked for specific parameters and type of attack is decided at run time only. And after detection of an attack extracted fields and related information of packet is stored into database as a record, so that evidence can be generated easily. In this system we are almost eliminating storage of network traffic for analysis purpose and as attack recognition is done at real time further investigation can be done faster. Proposed model has been tested and results are found satisfactory.

GENERAL TERMS: *Network forensics.*

KEYWORDS: *WinPcap, Network forensics, ICMP Sweep, ICMP Smurf, Traceroute network mapping, etc.*

I. INTRODUCTION

Today's world is very fast and an internet is essential for faster communication, faster transaction as well as faster completion of task. But this internet is victim of cyber crimes especially economical transactions are main attraction of an attacker. So to find out attacks and attackers behind the cyber crime we require forensics techniques like "Network forensics".

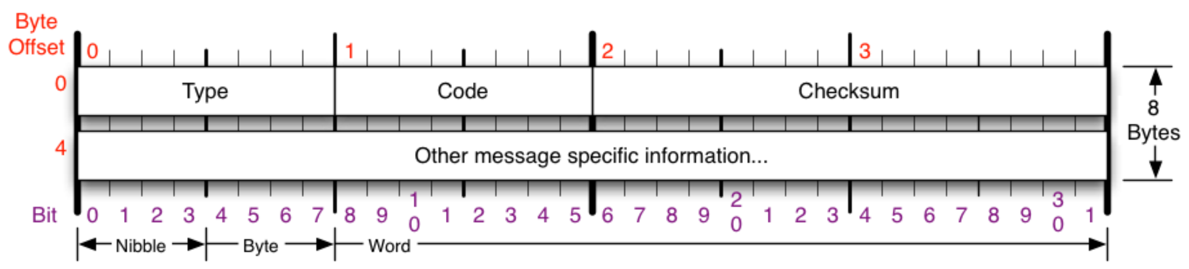
Network forensics *is a sub-branch of digital forensics related to the monitoring and analysis of network traffic for the purpose of generating legal evidence.*[2] Unlike other areas of digital forensics, network traffic inspections are concern with volatile and dynamic information as network traffic is transmitted and then lost, so network forensics is often a pro-active investigation. In network forensics systems network traces are gathered from victim system with the help of security tools like IDS or firewall logs and then NFATs (Network Forensics Analysis Tools) are used to analyze traces for identification of attack and finding out source of attack. Generally in network forensics systems network traffic is captured, then it is stored and further analysis is performed for generating legal evidence of an attack. In this process vast storage is required to store network traces as well as time required for analysis is more, in our proposed system we are analysing network traffic at real time so no need to store packets anywhere. And as recognition of an attack is also performed at run time investigation time is minimized than existing systems.

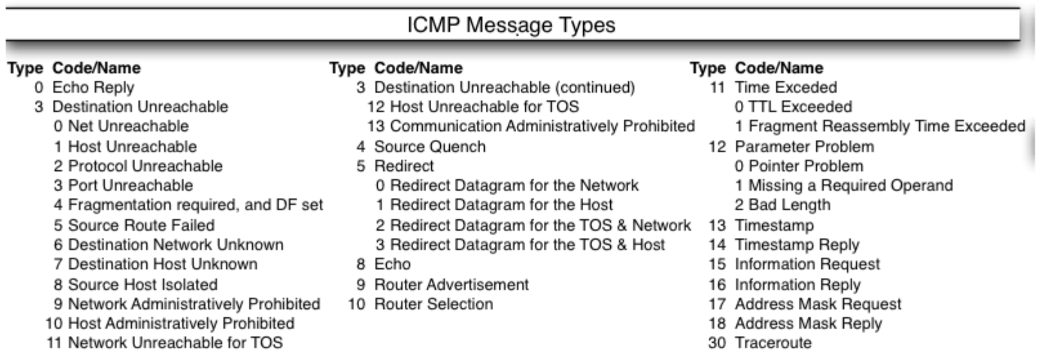
In our proposed system we are concentrating on attacks related with ICMP protocol because of following reasons.

- ICMP (Internet Control and Messaging protocol) is protocol among IP protocol family. ICMP is always transferred in the payload of IP packet.

- Within the Internet Protocol, data is carried by the TCP, UDP etc., whereas ICMP is for confirmation about whether source as well as destination are in working condition or not. Thus all that ICMP does is diagnostic tasks on the Internet, and is not used to carry any data.
- ICMP protocol is utilized to give either error handling or query messages, as well as it also provide facility to give one-way informational request and reply messages to host and inform source host about errors in packet flow up to destination via error messages.
- ICMP messages are sent in several situations: for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route, etc. [3].
- Actually ICMP is an integral part of IP, and it is implemented by every IP module for query messages, for error reporting or for request & reply messages [2]. But because of such nature of ICMP messages, these messages are generally ignored by security tools like firewall and IDS, etc. So most of the attackers are utilizing ICMP messages to launch an attack.

The ICMP header is generally as shown below in fig.1.





ICMP Message Types

Type	Code/Name
0	Echo Reply
3	Destination Unreachable
	0 Net Unreachable
	1 Host Unreachable
	2 Protocol Unreachable
	3 Port Unreachable
	4 Fragmentation required, and DF set
	5 Source Route Failed
	6 Destination Network Unknown
	7 Destination Host Unknown
	8 Source Host Isolated
	9 Network Administratively Prohibited
	10 Host Administratively Prohibited
	11 Network Unreachable for TOS
3	Destination Unreachable (continued)
	12 Host Unreachable for TOS
	13 Communication Administratively Prohibited
4	Source Quench
5	Redirect
	0 Redirect Datagram for the Network
	1 Redirect Datagram for the Host
	2 Redirect Datagram for the TOS & Network
	3 Redirect Datagram for the TOS & Host
8	Echo
9	Router Advertisement
10	Router Selection
11	Time Exceded
	0 TTL Exceeded
	1 Fragment Reassembly Time Exceeded
12	Parameter Problem
	0 Pointer Problem
	1 Missing a Required Operand
	2 Bad Length
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply
30	Traceroute

Fig.1. General ICMP Header with types and code

There are three important fields for any ICMP header as

1) Type - ICMP messages are sent using the basic IP header. The first octet of the data portion of the datagram is a ICMP type field; the value of this field determines the format of the remaining data. The message type identifies what sort of ICMP message it is (echo request for ping, router solicitation, redirect, etc.).
2) Code - Each ICMP message type also has a message code that lets you know the exact meaning of Type.
3) Checksum - The checksum is the 16-bit ones's complement of the one's complement sum of the ICMP message starting with the ICMP Type. For computing the checksum the checksum field should be zero.

For calculating checksum following formula is used:

Internet Header + 64 bits of Data Datagram [3]

This data is used by the host to match the message to the appropriate process. If a higher level protocol uses port numbers, they are assumed to be in the first 64 data bits of the original datagram's data.

ICMP messages are ignored by firewall, IDS like systems and here comes the reason of attacks done with ICMP protocol. There are number of attacks done on ICMP protocol but in our proposed system we will have only few of them as described below.

1.1. ICMP Attacks

1.1.1. ICMP Sweep Attack

In any typical attack scenario, the attacker will first do some scanning activities in order to
1. Better understand the environment of the target
2. Collect information about the target for planning attack approach
3. Apply the right techniques & tools for the subsequent attack phases

ICMP sweep is the attack used for such scanning purpose. ICMP Sweep is the most used technique for discovering the range of hosts which are alive in the target's environment. So, ICMP Sweep is not direct attack.

For ICMP Sweep an attacker follows following steps:

1. Send a series of ICMP request packets to the target network range.
2. Form list of ICMP replies.
3. Confirm alive hosts in the target's network range.[10]

Although the attack can be done manually via a very simple command **ping**, many automated scanning tools (E.g. nmap and Superscan) will speed up the entire scanning process by performing such a scan on all possible IP address range given a target network. Querying multiple hosts using ICMP ECHO is referred to as *ICMP Sweep* (or *Ping Sweep*)[4].

1.1.2. ICMP Smurf Attack

Whenever a type 8 ICMP eco request is sent, a type 0 ICMP eco reply is sent back. In a smurf attack, an attacker will spoof the source address of the ICMP packet and send a broadcast to all computers on that network. If networking devices do not filter this traffic, then they will be broadcasted to all computers in the network. The victim's network gets congested by this much traffic, which brings down the productivity of the entire network. The smurf attack is illustrated with steps given below:

1) Attacker finds some intermediary network that will respond to the network's broadcast address.
2) Attacker spoofs the IP address of the victim host and sends a great number of ICMP echo request packets to the broadcast address of the above intermediary networks.
3) Now all the hosts on that network will respond to that ICMP echo request with a corresponding ICMP reply request back to the spoofed IP address (the victim).
4) This will send a whole bunch of ICMP echo replies to the victim and its network thus causing network degradation or a total denial of service. [11]

1.1.3. Traceroute Network Mapping Attack

Traceroute attack gives the list of traversed routers in simple text format, together with timing information.

The **traceroute** command is used to discover the routes that packets actually take while travelling to their destination.

- The device (for example, a router or a PC) sends out a sequence of User Datagram Protocol (UDP) datagrams to an invalid port address at the remote host. Three datagrams are sent, each with a Time-To-Live (TTL) field value set to one. The TTL value of 1 causes the datagram to "timeout" as soon as it hits the first router in the path; this router then responds with an ICMP Time Exceeded Message (TEM) indicating that the datagram has expired.
- Another three UDP messages are now sent, each with the TTL value set to 2, which causes the second router to return ICMP TEMs. This process continues until the packets actually reach the other destination.
- Since these datagrams are trying to access an invalid port at the destination host, ICMP Port Unreachable Messages are returned, indicating an unreachable port; this event signals the Traceroute program that it is finished.
- This "traceroute" command will send out progressively a series of packets with an increasing TTL (Time to Live) value set. [4]

- When an intermediate router receives a forwarding packet, it will decrement the TTL value of the packet before forwarding it to the next router. At this time if the TTL value of the packet reaches zero, an ICMP "time exceeded" message will be send back to the originating host.
- By sending the packet with initial TTL value of 1 will allow the first router in the path of the packet to now send back an ICMP "time exceeded" message which will then allow the attacker to know the IP address of the first router.
- Subsequent packets are send by increasing the TTL value in the packet by 1 each time, thus the attacker will be able to know every hop between him and the target. With this attack, the attacker can trace the path of a packet to the destination as well as he can also get information of the topology of the target network. [12]

1.1.4. Inverse Mapping

Inverse Mapping is a technique for checking out internal networks or hosts of target system which are generally protected by filtering devices like firewall. An Inverse Mapping attack is illustrated below

1) Attacker sends an ICMP reply message to a range of IP addresses presumably behind a filtering device.
2) Upon receiving the series of ICMP reply messages, since the filtering device does not keep state of the list of ICMP requests, it will allow these packets to their destination.
3) If there is an internal router, the router will respond with a ICMP "Host Unreachable" for every host that it cannot reach, thus giving the attacker knowledge of all hosts which are present behind the filtering device.[3],[4].

II. LITERATURE SURVEY

According to Yasinsac and Manzano [5] computer and network forensics techniques are essential as cyber crimes are increasing day-by-day. They define Computer forensics as "Computer forensics is the art of discovering and retrieving information about a crime in such a way to make it admissible in court". They propose enterprise network and computer related policies that will deter computer crime and enhance recovery from attacks by facilitating computer and network forensics.

Meghanathan and Moore [6] presents different tools and techniques for conducting network forensics like eMailTrackerPro – to identify the physical location of an email sender; Web Historian – to find the duration of each visit and the files uploaded and downloaded from the visited website. As well as they also have a survey of different IP traceback techniques like packet marking that help a forensic investigator to identify the true sources of the attacking IP packets.

Pelaez and Fernandez [7] discuss a systematic approach to network forensic collection and analysis of data in converged networks. They also suggested a forensic model for Voice over IP (VoIP)-based network attacks.

Mukkamala and Sung [8] state that capturing network activity for forensic analysis is simple in theory, but relatively trivial in practice. They discuss about the use of artificial intelligent techniques for identifying features of network forensics analysis. They also represent study about two artificial intelligent techniques. Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs).

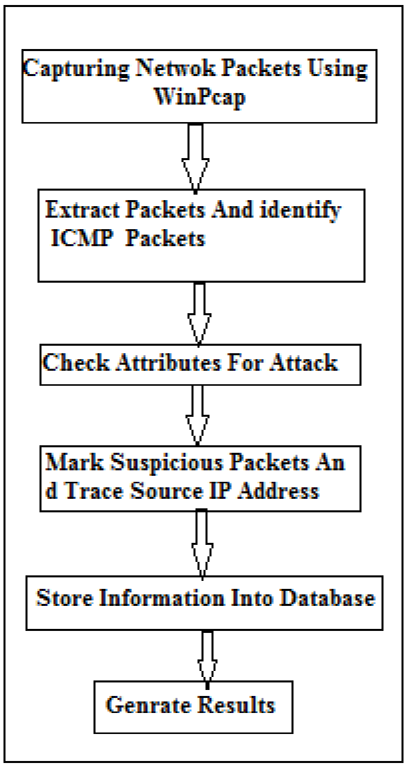
Jatinder Kaur et al [18] have implemented Linux based Network forensics system which is utilizing honey nets. In this system they are utilizing network forensics method to collect malware behaviours, in order to ensure the effectiveness of digital evidence and credibility of the evidence on judicial review.

Raftopoulos and Dimitropoulos [19] have investigated about 200 security incidents within an operational network manually. They have collected data from Snort alerts, reconnaissance and vulnerability scanners, blacklists, and a search engine to manually investigate these incidents. As a result of their experiment it is found that search engine is better among the above four traditional ways for providing evidence. Also they have enlisted 138 Snort signatures that will be effective in identifying validated malware without producing false positives.

Nazanin Borhan et al [20] put forth a Trusted Module Platform (TPM)-based solution using Secure Virtual Machines (SVM) for secure storage of forensic logs on computer system. Such logs are very useful during cyber forensics investigation.

III. Proposed System

We propose a model for real time Network Forensics system where network traffic is captured, features of packets are extracted and analysis is done for attacks at run time as well as immediate report is generated about type of attack. Here we are concerning attacks on ICMP protocol. We are using Perl language and WinPcap in this system. This model is addressing problem of network traffic storage as well as enhancing speed of inspection. In existing forensics systems network traces are captured and stored in pcap file which requires memory to store. And analysis is done later but in our proposed system we are not storing network traces anywhere, instead of that we are directly extracting packet features, checking for specific parameters and if values of parameters are violated marking that packet as suspicious .ICMP header is checked for TYPE and CODE attributes, attack done on the system is identified with the help of values of TYPE and CODE. Following fig.2 shows working of system.


Capturing Netwok Packets Using WinPcap
Extract Packets And identify ICMP Packets
Check Attributes For Attack
Mark Suspicious Packets An d Trace Source IP Address
Store Information Into Database
Genrate Results

.

Fig.2. Proposed System

In our proposed system following steps are used for analysis.

3.1 Capture packet

In this first step network packets are captured with WinPcap packet capture and extraction tool. WinPcap is the industry-standard tool for link-layer network access in Windows environments. It allows applications to capture and transmit network packets bypassing the protocol stack, and has additional useful features, including kernel-level packet filtering, a network statistics engine and support for remote packet capture. WinPcap tool provides library for easy access of the low-level network layers. The library also contains Windows version of the libpcap UNIX API.

3.2 Extraction and identification of packet

In this second step we are using inbuilt modules of Perl language. Here packets are received through WinPcap and with following inbuilt modules and functions of Perl language further operations are done.

3.2.1 Net::Pcap module

Following functions of module are utilized in this proposed system.

1) Net::Pcap::lookupdev(\$err)
The Net::Pcap::lookupdev(\$err) function returns the name of a network device which is used with Net::Pcap::open_live() function. When error occurred the $err parameter is filled with an error message otherwise it is undefined.
2) Net::Pcap::lookupnet($dev, \$net, \$mask, \$err)
This function is used to determine the netmask and network number for the device which is specified in $dev. It returns 0 when it is successful and sets the $mask and $net parameters with values whereas on failure this function returns -1 and the $err parameter is filled with an error message.
3) Net::Pcap::open_live($dev, $snaplen, $promisc, $to_ms, \$err)
For looking at packets on the network this function returns a packet capture descriptor, which network interface to capture packets from is specified by the first $dev parameter. The maximum number of bytes to capture from each packet, and whether to put the interface into promiscuous mode, respectively is specified by the $snaplen and $promisc parameters. The $to_ms parameter of this function specifies a read timeout in milliseconds. If an error occurs the packet descriptor will be undefined, and the $err parameter will be set with an appropriate error message.
4) Net::Pcap::compile($pcap_t, \$filter_t, $filter_str, $optimize, $netmask)
This function of Net::Pcap is used to compile the filter string contained in $filter_str and store it in $filter_t. The netmask of the network device must be specified in the $netmask parameter. The function returns 0 if the compilation was successful or -1 if there was a problem. The filter is optimized the filter if the $optimize variable is true.
5) Net::Pcap::setfilter($pcap_t, $filter_t)
This function associate the compiled filter stored in $filter_t with the packet capture descriptor $pcap_t.
6) Net::Pcap::loop($pcap_t, $cnt, \&callback_fn, $user_data)
This function is used to read the $cnt parameter from the packet capture descriptor $pcap_t and then call the perl function and callback_fn with an argument of $user_data. If $cnt is -ve, then the function loops infinite times otherwise up to when an error occurs. packet header information and packet data is passed by The callback function.
7) Net::Pcap::close($pcap_t)
This function close the packet capture device associated with descriptor $pcap_t.

3.2.2 Netpacket module

This module is classifying all packets under categories like TCP, UDP, IP, etc By using following functions of this module IP packets among the network packets are identified. Then from the payload of each IP packet ICMP header is extracted. After extraction of header, fields of header and their values are taken and checked. Following functions are used from this inbuilt module.

1) NetPacket::Ethernet->decode([RAW PACKET])
This method of NetPacket is used to decode the raw packet data which is given an object containing instance data and return an object containing instance data. This method will quite decode garbage input. For valid packet data transfer only the programmer is responsible.
2) NetPacket::IP->decode([RAW PACKET])
This method of NetPacket is used to decode the raw packet data which is given an object containing instance data and return an object containing instance data. This method will quite decode garbage input. For valid packet data transfer only the programmer is responsible.
3) NetPacket::IP->encode()
This method of NetPacket is used to return an IP packet encoded with the instance data specified. This will infer the total length of the packet automatically from the payload length and also adjust the checksum.
4) NetPacket::ICMP->decode([RAW PACKET])
This method of NetPacket is used to decode the raw packet data which is given an object containing instance data and return an object containing instance data. This method will quite decode garbage input. For valid packet data transfer only the programmer is responsible.
5) NetPacket::ICMP->encode()
Return an ICMP packet encoded with the instance data specified.

6) NetPacket::ICMP::strip([RAW PACKET])
This function return the encapsulated data (or payload) contained in the ICMP packet as given below
- **Type:** The ICMP message type of this packet.
- **Code:** The ICMP message code of this packet.
- **Cksum:** The checksum for this packet.
- **Data:** The encapsulated data (payload) for this packet.

3.3. Check attributes of an attack

ICMP fields like Type and Code plays important role while deciding type of attack. Following are values of type and code with which we can decide the type of attack [1]:
The packets having such violated values of attributes as given in table.1 are marked as suspicious and thus we can identify an ICMP attack on the system run-time and we can generate immediate report for an attack.

Table 1. Parameters for ICMP attacks. [1]

Attack on ICMP protocol	**Significant parameters**
ICMP Sweep Attack	Type=8, and Code=0.
ICMP Smurf attack	Type=0 and without sending type=8
Traceroute Network mapping attack	TTL =0 and Type = 8
Inverse mapping attack	Type=0 and without sending type=8.

3.4 Marking of suspicious packets

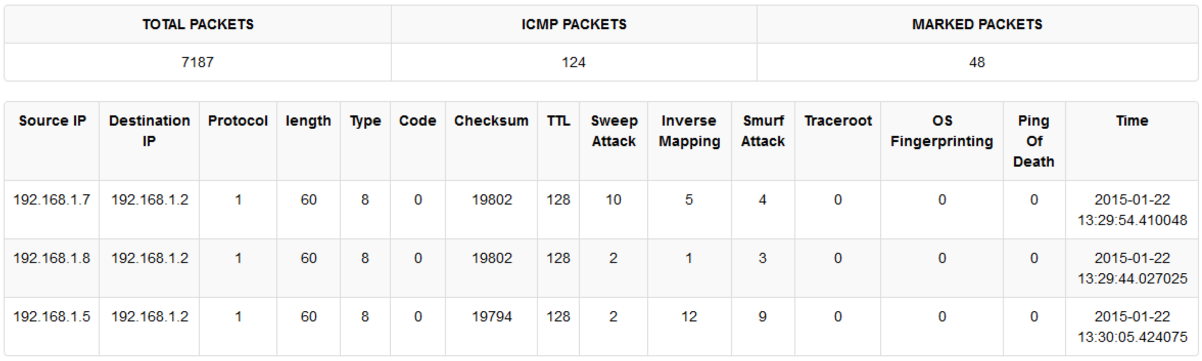
By using values of type and code packets are marked as suspicious.

3.5 Storing results in database

As shown in fig.3 we are storing fields of suspicious ICMP packets in database.

IV. Results And Discussion

In our system we are using ActivePerl 5.16.3 for perl language and its inbuilt packets, wampserver 2.4 as a framework for php, mysql, apache. And we are having WinPcap 4.3.1.Also we are using attack launching tools like sing and traceroute for launching attacks on system. With the help of WinPcap and its inbuilt libraries we are capturing network traffic. After that packets are extracted and ICMP packets are identified with the help of program in Perl language and its inbuilt packets. For detection of suspicious packets we are using significant network parameters like type and code as given in table.1 corresponding to various network attacks on ICMP protocol. All this analysis is done with real time dynamic approach. We are using mysql database to store attributes and values of attributes for marked ICMP packet. Also we are tracking IP address of source of attack and storing into database as a record. Following fig.3 will give a snapshot of records in database captured at an instant. And fig.4 shows a pie chart for same instant records.



TOTAL PACKETS	ICMP PACKETS	MARKED PACKETS
7187	124	48

Source IP	Destination IP	Protocol	length	Type	Code	Checksum	TTL	Sweep Attack	Inverse Mapping	Smurf Attack	Traceroot	OS Fingerprinting	Ping Of Death	Time
192.168.1.7	192.168.1.2	1	60	8	0	19802	128	10	5	4	0	0	0	2015-01-22 13:29:54.410048
192.168.1.8	192.168.1.2	1	60	8	0	19802	128	2	1	3	0	0	0	2015-01-22 13:29:44.027025
192.168.1.5	192.168.1.2	1	60	8	0	19794	128	2	12	9	0	0	0	2015-01-22 13:30:05.424075

Fig.3. Snapshot of records in database

According to above snapshot in fig.3, total packets captured at instance time were 7187 among that 124 were ICMP packets and 48 packets were found suspicious.
Following fig.4, will give the pie chart for the same instance. According to pie chart 96% packets among total packets are other than ICMP packets, 3% packets out of total packets are ICMP packets and out of these ICMP packets 2% packets are found suspicious and those packets are taken for further analysis. By analysing those marked packets we are finding out type of attack done on system as well as we are also tracing the source of attack.

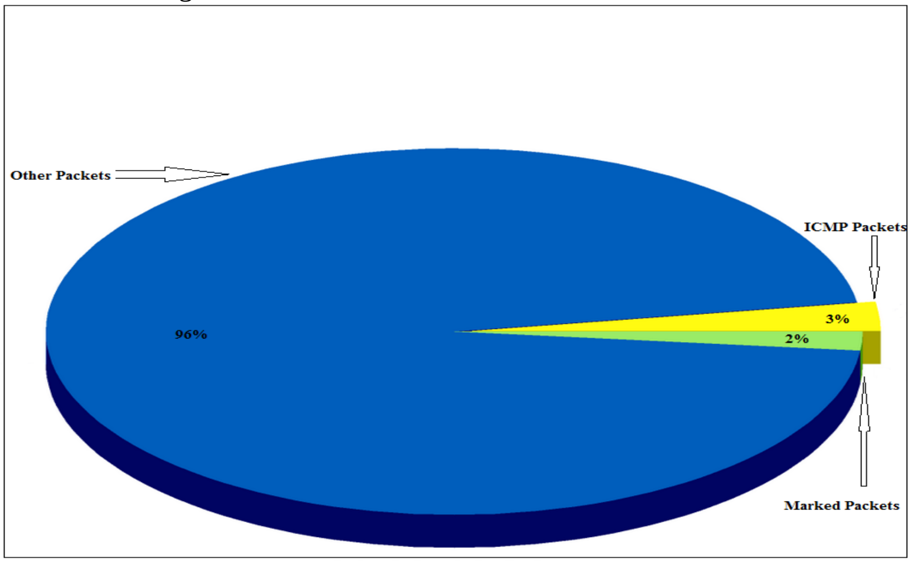


Fig.4.Pie Chart

V. Conclusion and Future Work

Any victim (person or system) is interested in quick investigation and results. In networking also investigation of attacked system and immediate reports about type of attack is very essential. Here with our system we are providing the same. We are concentrating on attacks done with ICMP protocol as ICMP protocol is used by large no. of attackers to launch attacks on network. With the help of WinPcap and Perl language we are capturing and analysing packets run-time which save a lot of time as well as we does not require to store information for any further investigation which saves the storage space. So, this system is providing advantage in time and space.
We have our focus on some specific ICMP attacks but we can extend this system for attacks on TCP protocol also in near future.

References

[1] A. Kaushik and R. Joshi, " Network Forensics Systems For ICMP Attacks", International Journal of Computer Applications vol. 2, no.3, May 2010
[2] E. Pilli, R. Joshi, R. Niyogi, "A Generic Framework for Network Forensics", International Journal of Computer Applications vol.1, no.11, 2010
[3] J. Postel, Internet Control Message Protocol, RFC 792. http:// tools.ietf.org/html/rfc0792
[4] SANS Institute Reading Room. ICMP attack illustrated http://www.sans.org/
[5] A. Yasinsac, and Y. Manzano, " Policies to Enhance Computer and Network Forensics", In IEEE Workshop On Information Assurance and Security 2001.
[6] N. Meghanathan, S. Reddy Allam and L. Moore, "Tools And Techniques For Network forensics", International Journal of Network Security & Its Applications ,vol.1, no.1, April 2009.
[7] J. Pelaez & E. Fernandez, "Network Forensics Models for Converged Architectures", International Journal on Advances in Security, vol.3 no 1 & 2, 2010.
[8] S. Mukkamala and A. Sung, "Identifying Significant Features for Network Forensic Analysis Using Artificial Intelligent Techniques", International Journal of Digital Evidence, 2003.
[9] http://www.winpcap.org/
[10] http://www.google.com/

[11] S. Devidoff, J. Ham, "Network Forensics – Tracking Hackers Through Cyberspace", Prentice Hall, 2012.
[12] S. Staniford, J. Hoagland, and J. McAlerney, "Practical automated detection of stealthy portscans", Journal of Computer Security 10, 2002.
[13] M. Bailey, E. Cooke, F. Jahanian, N. Provos, K. Rosaen and D.Watson, " Data reduction for the scalable automated analysis of distributed darknet traffic", In Proceedings of USENIX/ACM Internet Measurement Conference, 2005.
[14] A. Dracinschi, S. Fdida, "Congestion Avoidance for Unicast and Multicast Traffic", In Proceedings of 1st European Conference on Universal Multiservice Networks, 2000.
[15] http://resources.infosecinstitute.com/
[16] Karen Kent, Suzanne Chevalier, Tim Grance, Hung Dang, " Guide to Integrating Forensic Techniques into Incident Response", National Institute of Standards and Technology Special Publication 800-86,Aug. 2006.
[17] http://www.wikipedia.com
[18] Jatinder Kaur, Gurpal Singh, Manpreet Singh, "Design & Implementation of Linux based Network Forensic System using Honey net", International Journal of Advanced Research in Computer Engineering & Technology vol. 1, no. 4, June 2012.
[19] E. Raftopoulos, X. Dimitropoulos, "Understanding Network Forensics Analysis in an Operational Environment", IEEE Security and Privacy Workshops, 2013.
[20] N. Borhan, R. Mahmod, A. Dehghantanha, " A Framework of TPM, SVM and Boot Control for Securing Forensic Logs", International Journal of Computer Applications, vol. 50, no.13, July 2012.

AUTHORS

Madhuri N. Turup is currently working as a lecturer in Information Technology Department at JSPM's J.S.C.O.E., Pune, Maharashtra, India. She has completed her Bachelor's Degree in Computer Science Engineering from TPCT's College of Engineering, Osmanabad, Maharashtra,India under Dr. B.A.M. University, Aurangabad. She is pursuing her Master's Degree from the M.B.E.S's College of Engineering, Ambajogai under same university. Her areas of research interest include Computer Networks & Network Forensics.

B.M. Patil is currently working as a Professor in P.G. Computer Science & Engineering Department in M.B.E. Society's College of Engineering, Ambajogai, India. He received his Bachelor's degree in Computer Engineering from Gulbarga University in 1993, MTech Software Engineering from Mysore University in 1999, and PhD Degree from Indian Institute of Technology, Roorkee, 2011. He has authored several papers in various international journals and conferences of repute. His current research interests include data mining, medical decision support systems, intrusion detection, cloud computing, artificial intelligence, artificial neural network, wireless network and network security.