

# A Rule based Http Anomaly Classifier

Sandeep Sahu

Assistant Professor, SRIT Jabalpur (M.P.)

Amit Kumar Dewangan

M.E. Research Scholar, SRIT Jabalpur (M.P.)

## ABSTRACT

Ever since the inception of internet network security has been the prime important area of research for computer scientists. Network security using Honeypot presents a system that pretends to have one or more network vulnerabilities that a blackhat is looking for. Actually it does not have those vulnerabilities; it does so just to deceive the intruder by stealthily monitoring the network. Honeypots are emerging technology and have got lot of attention of late. In this research work, our system uses the advantages of Honeypot for implementing an intrusion detection system. There are two major part of our research work. First is data accumulator and second is data analyzer. For data accumulation, we have used honeypot. We have also used open source tool Honeyd which is available free of cost. Honeyd is a powerful tool which can simulate even complex networks very easily. Second part is data analyzer which analyzes data captured by the Honeyd. This part of the system is basically a java based Intrusion detection system which can work along honeyd system. This is a basic pattern based IDS which uses snort rule base to detect intrusion. It is a feature rich data analyzer which can detect intrusion. Function of the system is quite simple it reads the data logged by the honeypot system and looks for intrusion pattern of rule base into the packets. This analyzer can work in both offline and online mode; in online mode it reads data directly from the network interface card while in offline mode it reads data from binary files (tcpdump) which also gives it an advantage that it can analyze data from other resources as well.

## 1. INTRODUCTION

Now a days the idea of connecting the world together through the internet, came the danger of “internet terrorism”, hacking. Tools such as anti-viruses, firewalls, intrusion detection systems and soon have been developed in order to protect us from the computers’ greatest enemy, the hackers. We are design Intrusion Detection System (IDS). Basically, there are two main types of intrusion detection systems: signature-based (SBS) and anomaly-based (ABS). SBS systems (e.g. Snort) rely on pattern recognition techniques where they maintain the database of signatures of previously known attacks and compare them with analyzed data. An alarm is raised when the signatures are matched. On the other hand ABS systems (e.g. PAYL) build a statistical model describing the normal network traffic, and any abnormal behavior that deviates from the model is identified. Classifier is a way to categories the anomalies and normal data.

## 2. MEHODOLOGY

Honeyd can be classified under the category of low-interaction Honeypots designed for detection, and occasionally deception. Honeyd is software that we install on a machine on your network, with the purpose of detecting unauthorized activity within our organization. It does so by either monitoring all the unused IPs in our network, or

emulating a network topology that we define. In both cases, it will emulate effective IP addresses. Honeyd allows a single host to claim as many as 65536 IP addresses. Any attempted connection to an unused IP address is assumed to be unauthorized or malicious activity. After all, if there is no system using that IP, why is someone or something attempting to connect to it? For example, if our network has a class C address, it is unlikely that every one of those 254 IP addresses is being used. Any connection attempted to one of those unused IP addresses is most likely a probe, a scan, or a worm hitting our network. Honeyd is not based on any advanced algorithms, hence it is easy to set up and maintain, and requires minimal resources on our machine. Honeyd can detect (and log) any activity on any UDP or TCP port, as well as some ICMP activity [6].

## 3. EXPERIMENTAL WORKS

Honeyd is a classifier that places a very important role in network environment to detect unauthorized access. Honeyd is a trap which attracts users to through a virtual environment. In this research work we have capture and analyzed data packets in the network environment using rule based (snort). The Pseudo process of analyzing the packets in the network environment as shown below.

### Algorithm: Pseudo code of analyzing data packets

#### Input

1. RuleBase (R) = {r<sub>1</sub>, r<sub>2</sub>, r<sub>3</sub>,.....r<sub>n</sub>}

where n is number of rules,

r<sub>i</sub> is ith rule.

2. HttpUrl ( HU) is a string = hu1 hu2 hu3.....huk

where k is lenght of string

#### Output

if HttpUrl is malicious then

flag = true

otherwise

flag = false

Analysing (R, HU)

1. N <- length of R

2. K <- length of HU

3. i <- 1

4. While (i<=N)

5. if R is sub string of HU

6. then

7. flag <- true;

8. return falg

9. else

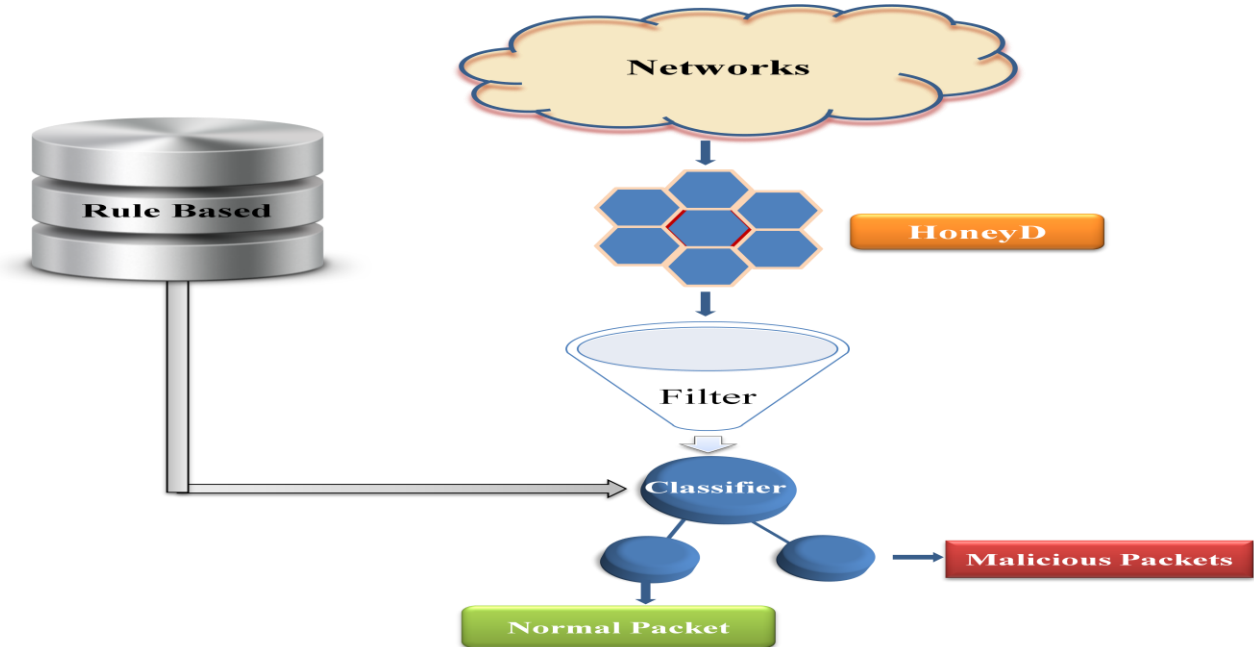
```

10.      flag <- false;
11.  end if
12.      i <- i+1
13.  end while
14. return flag <- false

```

### 3.1 Design and Implementation

The project setup a virtual network using low interaction Honeypot daemon Honeyd. The project has been tested for a complex network; one simple network having some primary features is shown below –



### 3.2 Component of Honeypot

Our proposed system has two main components

- I. Data Accumulator
- II. Data analyzer

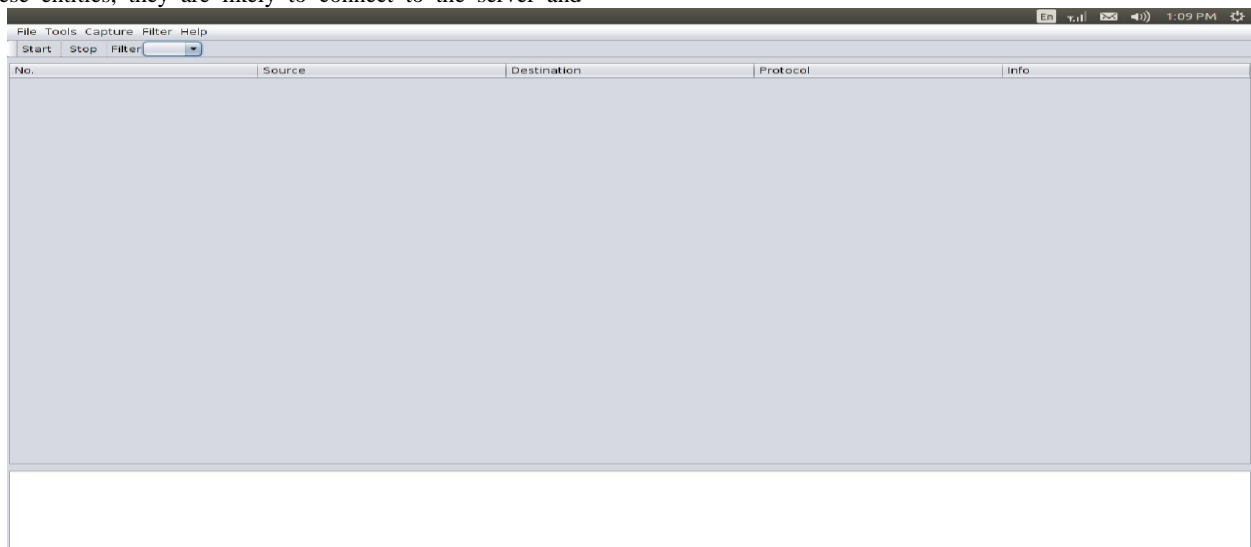
Overall design of the system is shown below

#### 3.2.1. Data Accumulator

This component of the system has primary function of accumulating data. To do so we have set up a Honeyd system this lures attackers by faking one or more web servers. These web servers are nothing but simple scripts which fake the behavior of web servers when attacker finds the presence of these entities, they are likely to connect to the server and

when this communication is going on Honeyd stealthily logs all the activity of the attacker. Though setting up Honeyd system is a trivial job, Honeyd provides lot of option which provides the whole system quite some depth. By using these options, we can give the simulated network a good realistic behaviour. Basic logging capability of Honeyd is restricted to text based logging only. We need to modify it to log in binary form as well, since our data analyzer reads data in binary form.

The project sets up a virtual network using low interaction Honeypot daemon Honeyd. The project has been tested for a complex network; one simple network having some primary features is shown below –



### 3.2.2. Data analyzer

Second and most important part of our system is a data analyzer. This part of the system is basically a java based Intrusion detection system which can work along Honeypot system. This is a basic pattern based IDS which uses snort rule base and some other rules to detect intrusion. It is a feature rich data analyzer which can detect intrusion. Function of the system is quite simple it reads the data logged by the Honeypot system and looks for intrusion pattern of rule base into the packets. This analyzer can work in both offline and online mode, In online mode it reads data directly from the network interface card while in offline mode it reads data from binary files (tcpdump) which also gives it an advantage that it can analyze data from other resources as well.

Component of data analyzer

I. Data Capturing Engine

II. Report Generator

III. Graph Generator

### I. Data Capturing Engine

This module of the system is responsible for capturing and analyzing the packet received. We have a java wrapper class jnetpcap for capturing and decoding the packets. The basic function of this module is very simple capture data from the active network interface and look into it for suspicious activity. To achieve this, we have two java classes namely Sniffer and PacketCapture. Sniffer class is responsible GUI component and mouse event management, it initializes the graphical component and takes the appropriate action like some mouse event occurs, for example pressing some button in the interface or selecting some menu option from the interface menu. In other words Sniffer class provides the user interface for our tool. In addition to this, it also does the most important task of filtering out the suspicious traffic. For every packet that the system come across it examines the details of the packet if something suspicious is found in it system marks it and keeps it aside. To come to the conclusion that the packet is suspicious, it uses a rule base which defines what patterns in a packet can be said suspicious. When a packet matches a pattern defined in the rule base the system marks the packet in user interface, generates the alert and logs the data in raw form. The GUI interface of the system is given below:

No.	Source	Destination	Protocol	Info
0	192.168.1.64	74.125.19.83	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
1	74.125.19.83	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
2	192.168.1.64	74.125.19.19	HTTP	GET /mail/?logout&hl=en HTTP/1.1
3	74.125.19.19	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
4	74.125.19.19	192.168.1.64	HTTP	HTTP/1.1 302 Moved Temporarily Ca
5	192.168.1.64	74.125.19.19	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
6	192.168.1.64	74.125.19.19	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
7	74.125.19.19	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
8	192.168.1.64	192.168.1.254		
9	192.168.1.64	192.168.1.254		
10	192.168.1.254	192.168.1.64		
11	192.168.1.254	192.168.1.64		
12	192.168.1.64	192.168.1.254		
13	192.168.1.254	192.168.1.64		
14	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=0 PSH=0 RST=0 SYN...
15	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
16	74.125.19.83	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
17	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
18	192.168.1.64	74.125.19.83	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
19	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
20	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
21	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
22	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
23	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
24	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
25	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
26	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
27	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
28	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
29	74.125.19.19	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
30	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
31	192.168.1.64	74.125.19.103	TCP	URG=0 ACK=1 PSH=1 RST=0 SYN...
32	192.168.1.64	74.125.19.19	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
33	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...
34	74.125.19.103	192.168.1.64	TCP	URG=0 ACK=1 PSH=0 RST=0 SYN...

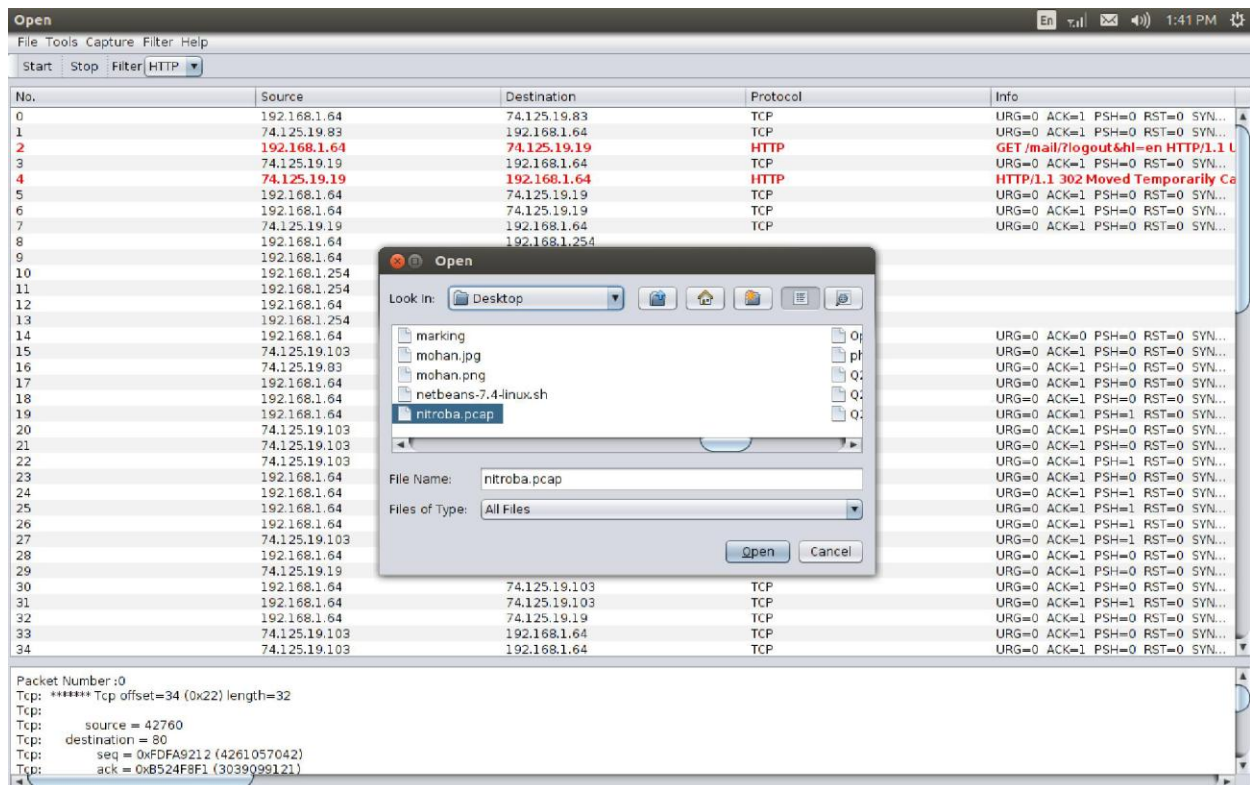
Packet Number :0	
Tcp:	***** Tcp offset=34 (0x22) length=32
Tcp:	
Tcp:	source = 42760
Tcp:	destination = 80
Tcp:	seq = 0xFDA9212 (4261057042)
Tcp:	ack = 0x8524F8F1 (3039099121)

### PacketCapture

PacketCapture class is responsible for reading the packet both in online and offline mode, It is also responsible for dumping the packet in raw form and printing the packet into the user interface. To capture packet, it first open up the appropriate network interface then sets up appropriate filter expression

and then it waits for the packets. For dumping the packet into the user interface, this module also does a bit of decoding. Functional demonstration of the PacketCapture is shown below:

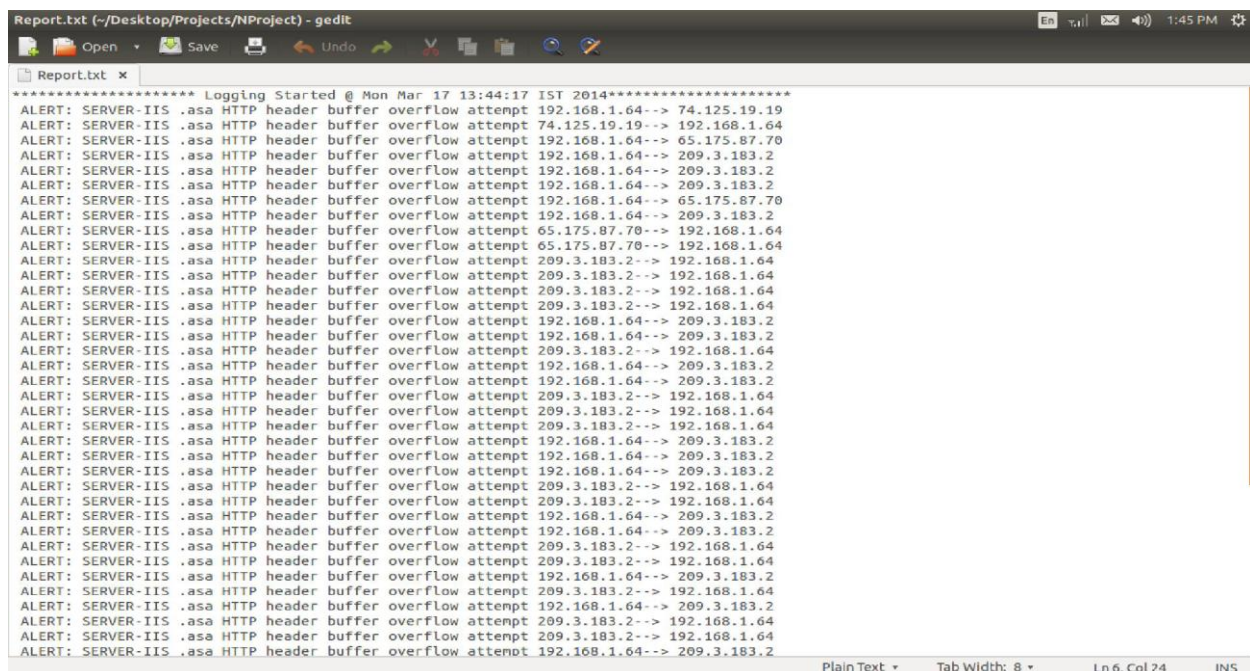




## II. Report Generator

The highlight of our system is its logging capability, since it logs the data both in text and binary format. Report generator class is responsible for creating a text based log for suspicious

activity only. Since, Honeyd already generates text based log for all the communicating data, this system does bother to create log for all the packets. A sample log is shown below:



## III. Graph Generator

This system is also capable of generating graph, this feature of the system greatly enhance the analyzing power of overall system.

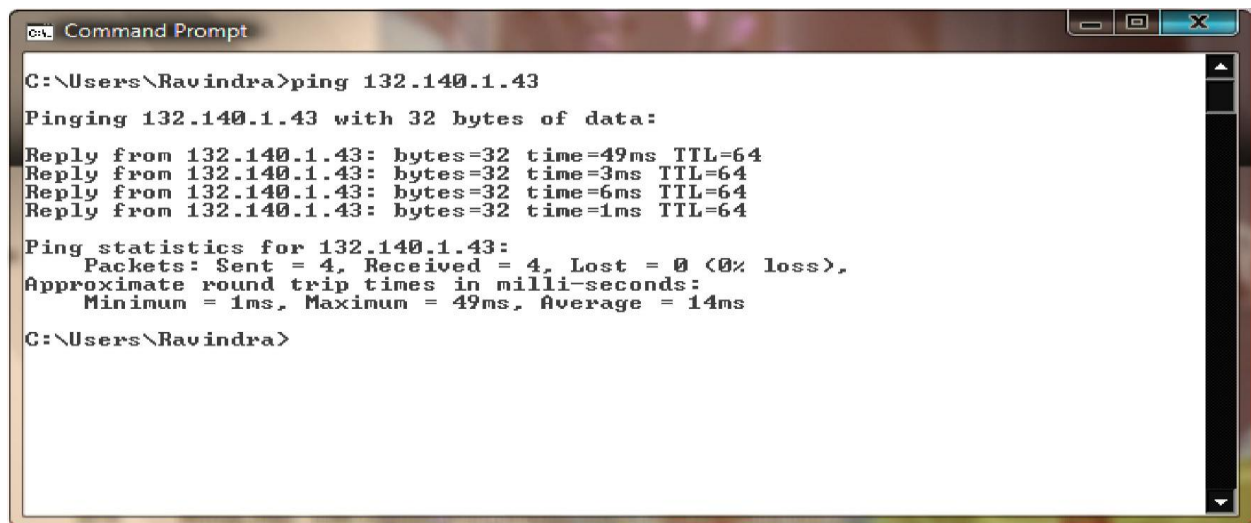
### 3.3 Results and Discussion

This experiment discussed results into two section: first is data accumulator and second is data gathering.

## I. Data Accumulator

We have been able to set up an advanced simulated network using Honeyd, earlier we have discussed how we can set up Honeyd, let us now see how it acquires data. We have successfully run the system which emulates various fake applications such as FTP and Telnet, we are also able to scan those IPs using nmap (Finger printing tool). Here is some screen shots showing the fake environment created by system:

- (i) Getting PING reply from virtual system:



```
Command Prompt

C:\Users\Ravindra>ping 132.140.1.43

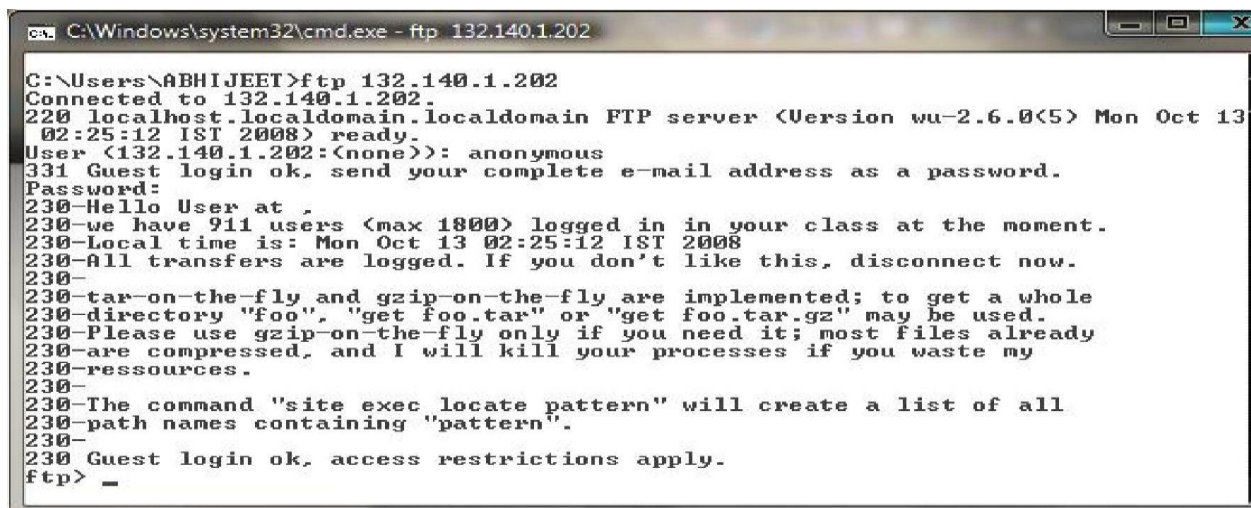
Pinging 132.140.1.43 with 32 bytes of data:

Reply from 132.140.1.43: bytes=32 time=49ms TTL=64
Reply from 132.140.1.43: bytes=32 time=3ms TTL=64
Reply from 132.140.1.43: bytes=32 time=6ms TTL=64
Reply from 132.140.1.43: bytes=32 time=1ms TTL=64

Ping statistics for 132.140.1.43:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 49ms, Average = 14ms

C:\Users\Ravindra>
```

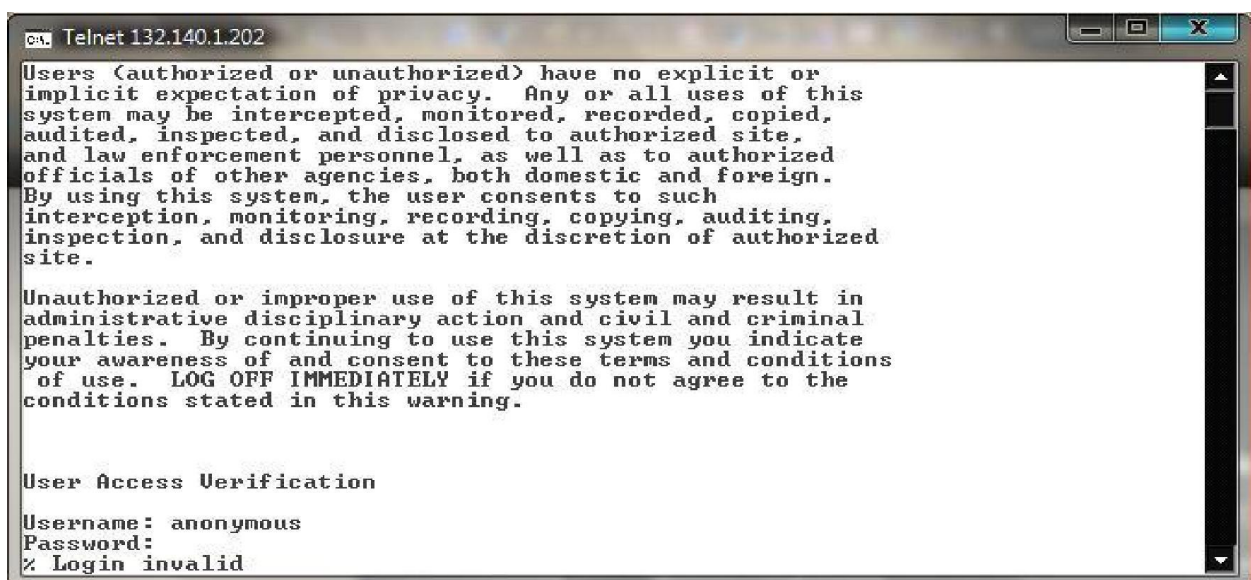
- (ii) FTP session with virtual system



```
C:\Windows\system32\cmd.exe - ftp 132.140.1.202

C:\Users\ABHIJEET>ftp 132.140.1.202
Connected to 132.140.1.202.
220 localhost.localdomain.localdomain FTP server (Version wu-2.6.0(5) Mon Oct 13
 02:25:12 IST 2008) ready.
User (132.140.1.202:(none)): anonymous
331 Guest login ok, send your complete e-mail address as a password.
Password:
230-Hello User at .
230-we have 911 users (max 1800) logged in in your class at the moment.
230-Local time is: Mon Oct 13 02:25:12 IST 2008
230-All transfers are logged. If you don't like this, disconnect now.
230-
230-tar-on-the-fly and gzip-on-the-fly are implemented; to get a whole
230-directory "foo", "get foo.tar" or "get foo.tar.gz" may be used.
230-Please use gzip-on-the-fly only if you need it; most files already
230-are compressed, and I will kill your processes if you waste my
230-resources.
230-
230-The command "site exec locate pattern" will create a list of all
230-path names containing "pattern".
230-
230-Guest login ok, access restrictions apply.
ftp> _
```

- (iii) A Telnet session with virtual system



```
Telnet 132.140.1.202

Users (authorized or unauthorized) have no explicit or
implicit expectation of privacy. Any or all uses of this
system may be intercepted, monitored, recorded, copied,
audited, inspected, and disclosed to authorized site,
and law enforcement personnel, as well as to authorized
officials of other agencies, both domestic and foreign.
By using this system, the user consents to such
interception, monitoring, recording, copying, auditing,
inspection, and disclosure at the discretion of authorized
site.

Unauthorized or improper use of this system may result in
administrative disciplinary action and civil and criminal
penalties. By continuing to use this system you indicate
your awareness of and consent to these terms and conditions
of use. LOG OFF IMMEDIATELY if you do not agree to the
conditions stated in this warning.

User Access Verification

Username: anonymous
Password:
% Login invalid
```

## **II. Data Gathered**

Honeyd is an efficient and simple tool to gather data; as Honeyd basically works at network level and understands major protocol udp, tcp and icmp, all other packets pertaining to other protocols are simply logged in and discarded. The Honeyd framework supports several ways of logging network activity. It can create connection logs that report attempted and completed connections for all protocols. As we have run many fake application can on virtual host, information can be gathered from the services themselves. If we gather data using Honeyd by exposing the virtual network for a longer period of time. We can gather some really useful data for research.

## **4. CONCLUSION**

Honeypots are very effective network security tool but it lacks supporting tools. We have tried to develop a tool which runs along the Honeypot to analyze data. Adding a tool like this greatly enhance the capability of the system. We have tested our tool on publicly available intrusion data which shows that only certain kind of attacks form the most of these attacks. If we make a system robust enough to tackle these attacks we will be able to prevent above 90 % of attacks.

Accuracy of the system depends on the ruleset. Better the ruleset more will be the accuracy. We have used the ruleset of the snort and some other open source. If we include more such ruleset it will increase the accuracy of the system. We have not spent much time into this matter as our primary goal was to enhance the capability of the Honeypot.

We have been able to develop a Honeypot system which analyzes data dynamically. It has all the advantage of Honeypot system plus it has an enhanced analyzing power. A GUI based system makes it capable to analyze any specific packet in depth. Its logging capability is also noteworthy as it can log data in both format binary and text. It also separates

the suspicious data from clean data which makes data more refined for research purpose. This eventually saves valuable time. We have concentrated on HTTP traffic but its capability can be enhanced by adding more and more protocols.

## **5. REFERENCES**

- [1] Dustin Lee, Jeff Rowe, Calvin Ko, Karl Levitt, "Detecting and Defending against Web-Server Fingerprinting", IEEE Computer Society , PP. 321-330, 2002.
- [2] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang and S. Zhou "Specification-based Anomaly Detection : A New Approach for Detecting Network Intrusions", ACM 1-58113-612-9/02/0011, PP. 18-22 , 2002.
- [3] Talasila Vamsidhar, Reddyboina Ashok and RayalaVenkat, "Intrusion Detection System for Web Application With Attack Classification" , Journal of Global Research in Computer Science , Volume 3, No. 12, PP. 44-50, 2012.
- [4] V. Jyothsna, V. V. Rama Prasad , K. Munivara Prasad , "A Review of Anomaly based Intrusion Detection Systems", International Journal of Computer Applications, Volume 28– No.7, PP. 26-35, 2011.
- [5] Juan M. Estevez-Tapiador, Pedro Garcia-Teodoro, Jesus E. Diaz-Verdejo "Measuring normality in HTTP traffic for anomaly-based intrusion detection", Elsevier Computer Networks, Volume 45, PP.175–193 , 2004.
- [6] Niels Provos , "Honeyd: A Virtual Honeypot Daemon ", Center for Information Technology Integration University of Michigan.