

12. Evading IDS, Firewalls, and Honeypots



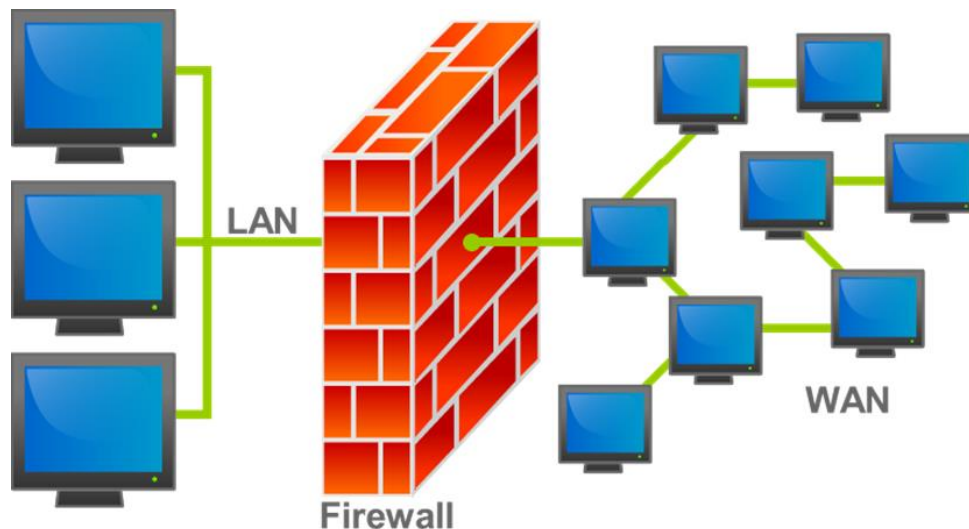
ETHICAL HACKING



Theory

Firewall

A firewall is a hardware or software appliance to secure the internal trusted network from the intruders by monitoring and controlling incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted internal network and an untrusted external network.



Types of Firewalls

1. Packet filter firewalls
2. Circuit-level gateways
3. Stateful inspection firewalls
4. Application-level gateways

Packet Filter Firewalls

Packet filtering firewall is used to control network access by monitoring outgoing and incoming packets and allowing them to pass or halt based on the source and destination Internet Protocol (IP) addresses, protocols and ports and this work on IP layer of TCP/IP. The packet filtering firewall examines the header of each packet based on a specific set of rules.

Circuit-level gateways

Circuit level gateways work at the session layer of the OSI model; they monitor TCP handshake to determine whether a requested session is legitimate or not. Information passed to a remote computer through a circuit level gateway firewall appears to be originated at user's computer. Firewall technology supervises TCP handshaking among packets to confirm that the session is genuine.

Circuit level gateways are relatively inexpensive and have the advantage of hiding information about the private network. On the other hand, they do not filter individual packets.

Application-level gateways

Application-level gateways can filter packets at the application layer of the OSI model. Application-level gateways examine traffic and filter on application specific commands such as HTTP, POST and GET. This works on the application layer of the TCP/IP Model.

Stateful inspection firewalls

Stateful inspection firewalls combine the aspects of the other three types of firewalls. They filter packets at the network layer, to determine whether session packets are legitimate, and they evaluate the contents of packets at the application layer. Traffic is filtered at three layers based on a wide range of the specified application, session, and packet filtering rules

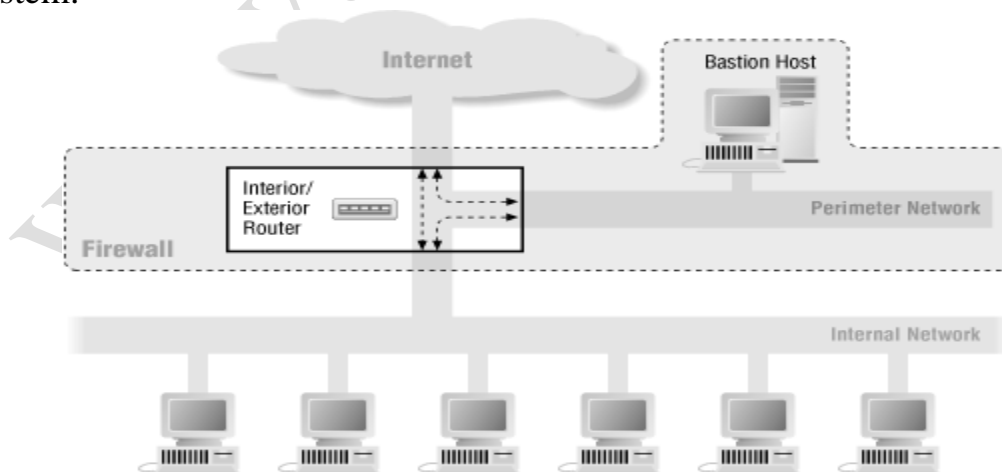
Types of Firewall Architectures

The different types of firewall architectures are

1. Bastion Host
2. Screened Subnet (DMZ - Demilitarized Zone)
3. Dual-Homed Firewall

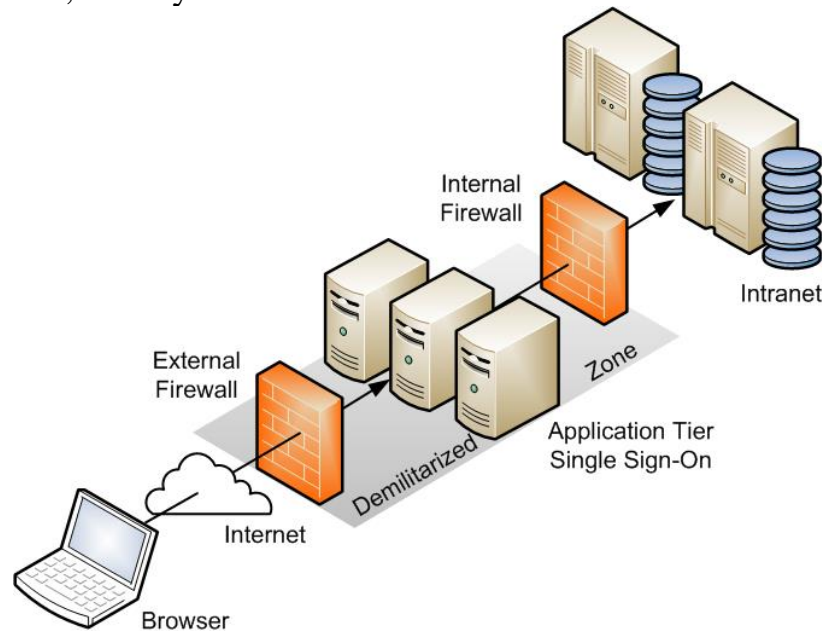
Bastion Host

A Bastion host is a computer that is fully exposed to attack. The system is on the public side of the demilitarized zone, unprotected by a firewall or filtering router. Frequently the roles of these systems are critical to the network security system.



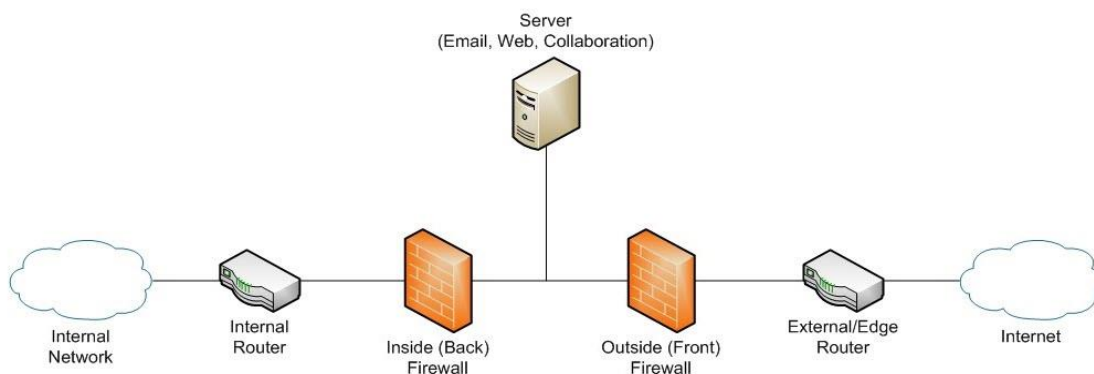
Screened Subnet (DMZ)

A screened subnet (also known as a "triple-homed firewall") is a network architecture that uses a single firewall with three network interfaces. In computer security, a DMZ or demilitarized zone is a physical or logical subnetwork that contains and exposes an organization's external-facing services to a larger and untrusted network, usually the Internet.



Dual-Homed Firewall

A dual-homed host (or dual-homed gateway) is a system fitted with two network interfaces (NICs) that sits between an untrusted network (like the Internet) and trusted network (such as a corporate network) to provide secure access. Dual-homed is a general term for proxies, gateways, firewalls, or any server running security applications or providing security services directly to an untrusted network.



Dual-homed hosts can be seen as a special case of bastion hosts and multi-homed hosts. They fall into the category of application-based firewalls. Dual-homed hosts can act as firewalls provided that they do not forward IP datagrams unconditionally.

List of Firewall Products

Software firewalls

- Lavasoft Personal Firewall
- NetLimiter
- Windows Firewall
- ZoneAlarm
- Nftables
- Netfilter/iptables
- IPFilter
- Norton 360
- PeerBlock
- Shorewall

Hardware firewalls

- Clavister
- FortiGate
- Sophos
- Juniper SSG
- Sonicwall
- Barracuda Firewall
- WinGate
- Endian Firewall
- Cisco ASA Firepower
- CiscoPI

Differences between stateful and stateless firewalls

Stateful firewalls monitor all aspects of the traffic streams, their characteristics and communication channels. These firewalls can integrate encryption or tunnels, identify TCP connection stages, packet state and other key status updates.

Stateless firewalls use clues from the destination address, source and other key values to assess whether threats are present, then block those deemed untrusted. Preset rules enforce whether traffic is permitted or denied, but the system is typically unable to determine the difference between truly desired communications and sophisticated attempts to disguise unauthorized communications as trusted ones.

PARAMETERS	STATELESS	STATEFULL
Philosophy	Treats each packet in isolation and does not relates to connection state	stateful firewalls maintain context about active sessions and use "state information" To speed packet processing
Filtering decision	Based on information in packet headers	Based on flows
Memory and CPU intensive	Low	high
Security	Low	High

Connection status	Unknown	Known
Performance	Fast	Slower
Related terms	Header info, IP address, port no etc.	State information, pattern matching etc.

The difference between the stateful and stateless firewalls are the stateful firewalls are capable of monitoring and detecting states of all traffic on a network to track and defend based on traffic patterns and flows. Stateless firewalls, however, only focus on individual packets, using preset rules to filter traffic.

Honeypot

In computer terminology, a honeypot is a computer security mechanism set to detect or deflect attempts at unauthorized access to the information systems. In other words, it is a simple trap to catch the hackers. In honeypots, we will emulate the required devices in an environment, and we will let attackers come there and try to perform attacks. But meanwhile, we will get the identity of the attacker. So that we can take action against attacks. Honeypots are of two types

Low Interaction Honeypot

Low interaction honeypots allow only limited interaction for an attacker. All services offered by a low interaction honeypot are emulated. Thus, these are not themselves vulnerable and will not become infected by the exploit attempted against the vulnerability.

High interaction honeypot

High interaction honeypots make use of the actual vulnerable service or software. These are usually complex as they involve real vulnerable operating systems and applications. In this type of Honeypots, nothing is emulated everything is real and provide a far more detailed picture of how an attack or intrusion progresses or how a particular malware executes in real-time.

List of honeypots

Database Honeypots

- Elastic honey - A simple elastic search honeypot
- NoSQL Honeypot Framework - A framework for NoSQL databases
- ESPot - elasticsearch honeypot

Anti-honeypot stuff

- Kippo detect - This is not a honeypot, but it detects kippo.

ICS/SCADA honeypots

- Conpot - ICS/SCADA honeypot
- Scada-honeynet - Mimics many of the services from a popular PLC and better helps SCADA researchers understand the potential risks of exposed control system devices

Service Honeypots

- Honey NTP - NTP logger/honeypot
- Honeypot camera - Observation camera honeypot
- Troje - A honeypot built around lxc containers. It will run each connection with the service within a separate lxc container.
- Slimp honeypot - A simple low-interaction port monitoring honeypot

Web honeypots

- Glastopf - Web Application Honeypot
- PhpMyAdmin honeypot - A simple and effective phpMyAdmin honeypot
- Servlet pot - Web Application Honeypot
- Node pot - A Nodejs web app NoSQL honeypot framework application
- Basic Auth Pot - HTTP basic authentication honeypot
- Shadow Daemon - A modular Web Application Firewall / High-Interaction Honeypot for PHP, Perl & Python apps
- Google Hack Honeypot - designed to provide reconnaissance against attackers that use search engines as a hacking tool against your resources.
- Smart Honeypot - PHP script demonstrating a smart honeypot
- WP Smart Honeypot - WordPress plugin to reduce comment spam with a smarter honeypot
- Word pot - A WordPress Honeypot
- Bukkit Honeypot - A honeypot plugin for Bukkit
- Laravel Application Honeypot - Simple spam prevention package for Laravel applications
- Stack Honeypot - Inserts a trap for spambots into responses
- Eos Honeypot Bundle - Honeypot type for Symfony2 forms
- Shock pot - WebApp Honeypot for detecting Shell Shock exploit attempts

Intrusion Detection System (IDS)

An intrusion detection system (IDS) is a device or software application that monitors network or computer system operations for malicious activities, policy violations and reports to a controlling station.

Capabilities of IDS

- Monitoring the operation of routers, firewalls, key management servers and files that are needed by other security controls aimed at detecting, preventing or recovering from cyber attacks.
- Including an extensive attack signature database against which information from the system can be matched.
- Recognizing and reporting when the IDS detects that data files have been altered.
- Generating an alarm and notifying the security operations team when there is a security breach.

IDS detection methods

Signature-based

Signature-based IDS performs detection of attacks by looking for specific patterns, such as byte sequences in network traffic, or known malicious instruction sequences used by malware. Signature-based IDS is very helpful for detecting already known attacks, but it fails in detecting new attacks, for which no pattern is available. Signatures fall into two categories

Attack signatures - They describe action patterns that may pose a security threat. Typically, they are presented as a time-dependent relationship between a series of activities.

Selected text strings - Signatures to match text strings which look for suspicious action (for example - calling /etc./passwd)

Anomaly-based

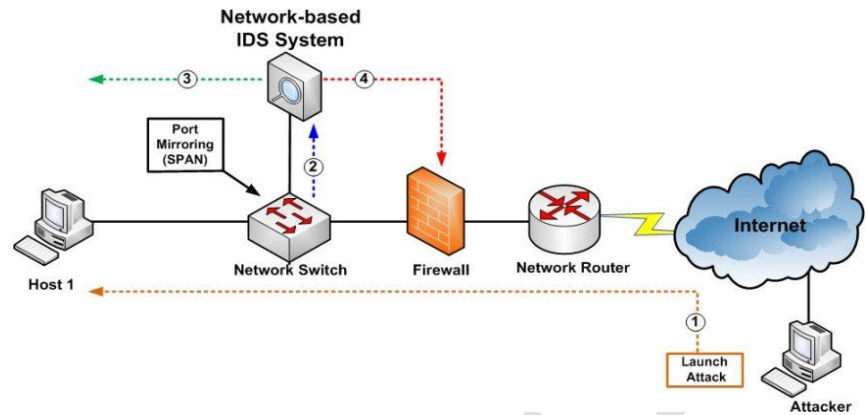
Anomaly detectors construct profiles that represent normal usage and then use current behavior data to detect a possible mismatch between profiles and recognize possible attack attempts. In order to match event profiles, the system is required to produce initial user profiles to train the system about legitimate user behaviors, which is a difficult and time-consuming task. Everything that does not match the stored profile is considered to be a suspicious action.

Types of IDS

1. Network-based Intrusion Detection System
2. Host-based Intrusion Detection System

Network-based IDS

NIDS is an IDS which can be configured on a network to monitor intrusions. This will notify the administrators about any possible signature match of attacks.

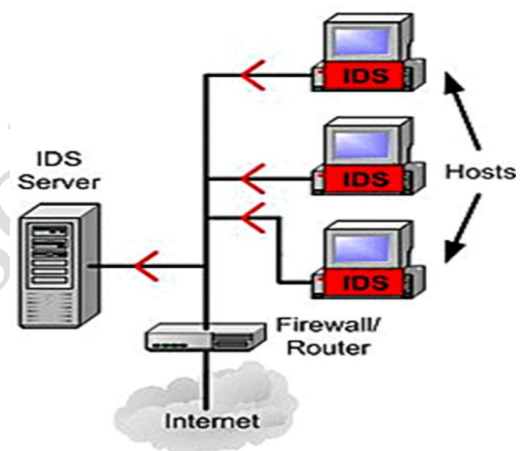


Host-based IDS

HIDS are the IDS systems which will be configured on the standalone machines and will only detect intrusions for that particular machine.

HIDS might detect which program accesses what resources and discover malicious attempts, for example, a word-processor has suddenly started modifying the system password database, which can be considered as a malicious attempt on sensitive data stored on the host machine.

Host Based IDS



List of Intrusion detection systems

- Snort IDS
- SonicWall
- Juniper
- McAfee Security Agent
- Palo Alto
- Cisco ASA Security Agent

Intrusion Prevention System (IPS)

An Intrusion Prevention System (IPS) is a network security/threat prevention technology that examines network traffic flows to detect and prevent vulnerability exploits. Vulnerability exploits usually come in the form of malicious inputs to a target application or service that attackers use to interrupt and gain control of an application or machine.

The IPS often sits directly behind the firewall and provides a complementary layer of analysis for dangerous content detection. The Intrusion Detection System (IDS) which is a passive system that scans traffic and reports back on threats but the Intrusion Prevention System (IPS) is placed in the direct communication path

between source and destination, that can actively analyze and take automated actions on all traffic that enter the network. These actions include:

- Sending an alarm to the administrator
- Dropping the malicious packets
- Blocking traffic from the source address
- Resetting the connection

Types of IPS

Host-based intrusion prevention systems

Host-based intrusion prevention systems are used to protect both servers and workstations through software that runs between your system's applications and OS kernel. The software is preconfigured to determine the protection rules based on intrusion and attack signatures. The HIPS will catch suspicious activity on the system and then, depend on the predefined rules; it will either block or allow the event to happen. HIPS monitor activities such as application or data requests, network connection attempts, and read or write attempts.

Network-based intrusion prevention systems

Network-based intrusion prevention system is a solution for network-based security. NIPS will intercept all network traffic and monitor it for suspicious activity and events, either blocking the requests or passing it. One interesting aspect of NIPS is that if the system finds an offending packet of information it can rewrite the packet so that attempt for the attack will fail, but the organization can mark this event to gather evidence against the intruder, without their knowledge.

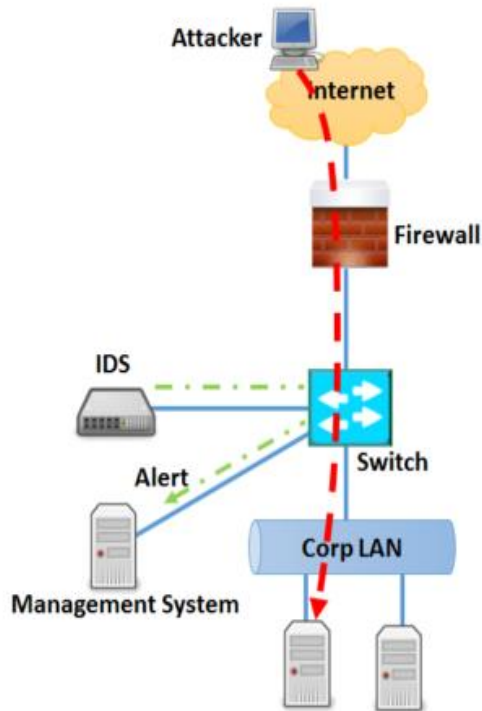
IDS vs IPS

Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) are both parts of the network infrastructure. IDS/IPS compare network packets to a cyberthreat database containing known signatures of cyberattacks — and flag any matching packets.

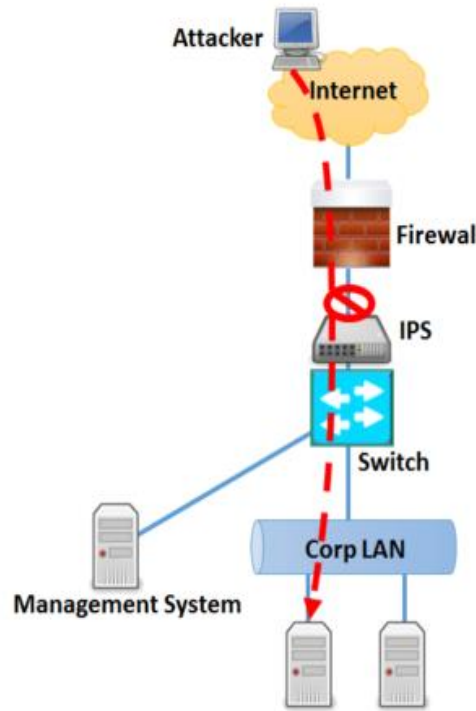
Both IDS/IPS read network packets and compare the contents to a database of known threats. The primary difference between them is what happens next. IDS are detection and monitoring tools that don't take action on their own. IPS is a control system that accepts or rejects a packet based on the ruleset. IDS require a human or another system to look at the results and determine what actions to take next, which could be a full-time job depending on the amount of network traffic generated each day. IDS make a better post-mortem forensics tool for the CSIRT to use as part of their security incident investigations. The purpose of the IPS, on the other hand, is to catch dangerous packets and drop them before they reach their

target. It's more passive than an IDS, simply requiring that the database gets regularly updated with new threat data.

Intrusion Detection System



Intrusion Prevention System



Many IDS/IPS vendors have integrated newer IPS systems with firewalls to create a Unified Threat Management (UTM) technology that combines the functionality of those two similar systems into a single unit. Some systems provide both IDS and IPS functionality in one unit.

Countermeasures

- Shut down switch ports associated with the known attack hosts.
- Perform an in-depth analysis of network traffic to detect all possible threats.
- Use a traffic normalizer to remove potential ambiguity from the packet stream before it reaches to the IDS.
- Harden the security of all communication devices such as modems, routers, switches, etc.

References:

1. Beal, V. (n.d.). Intrusion Detection (IDS) and Prevention (IPS) Systems. Retrieved from https://www.webopedia.com/DidYouKnow/Computer_Science/intrusion_detection_prevention.asp
2. WHAT IS AN INTRUSION PREVENTION SYSTEM? (n.d.). Retrieved from <https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-prevention-system-ips>
3. Firewall image reference: Firewall (computing). (2018, July 03). Retrieved from [https://en.wikipedia.org/wiki/Firewall_\(computing\)](https://en.wikipedia.org/wiki/Firewall_(computing))
4. Recommended Deployment Configurations. (n.d.). Retrieved from https://irmbor.co.rs/~dspalovic/assets/docsOracle/E36387/html/ol_recdepcfg_sec.html
5. Wilkins, S. (2015, August 25). A Guide To DMZs And Screened Subnets - DMZs And Screened Subnets. Retrieved from <http://www.tomsitpro.com/articles/dmz-screen-subnets-guide,2-919.html>
6. Intrusion Detection Systems. (n.d.). Retrieved from <http://www.anses.net.in/index.php/network-and-data-security/technology-for-internet-security/intrusion-detection-systems/>
7. <https://ipwithease.com/difference-between-ips-and-ids-in-network-security/>
8. <https://ipwithease.com/stateless-vs-stateful-firewall/>



Practicals

INDEX

S. No.	Practical Name	Page No.
1	Detecting Malicious Traffic in a network using SNORT-NIDS	1
2	Using KFSensor to build a Honeypot	8
3	Using of IPtables	12
4	Installation and usage of Pentbox	17



THIS DOCUMENT INCLUDES ADDITIONAL PRACTICALS WHICH MAY OR MAY NOT BE COVERED DURING CLASSROOM TRAINING. FOR MORE DETAILS APPROACH LAB COORDINATORS

Practical 1: Detecting Malicious Traffic in a network using SNORT-NIDS

Description: In this practical you will learn how to install and configure Snort an open source Network Intrusion Detection System to detect the malicious traffic in your network if it receives any.

Part 1: Installing Snort IDS in Parrot Linux.

snort package is not available in parrot repository. So, we can't install it directly. we will install snort from ubuntu repositories.

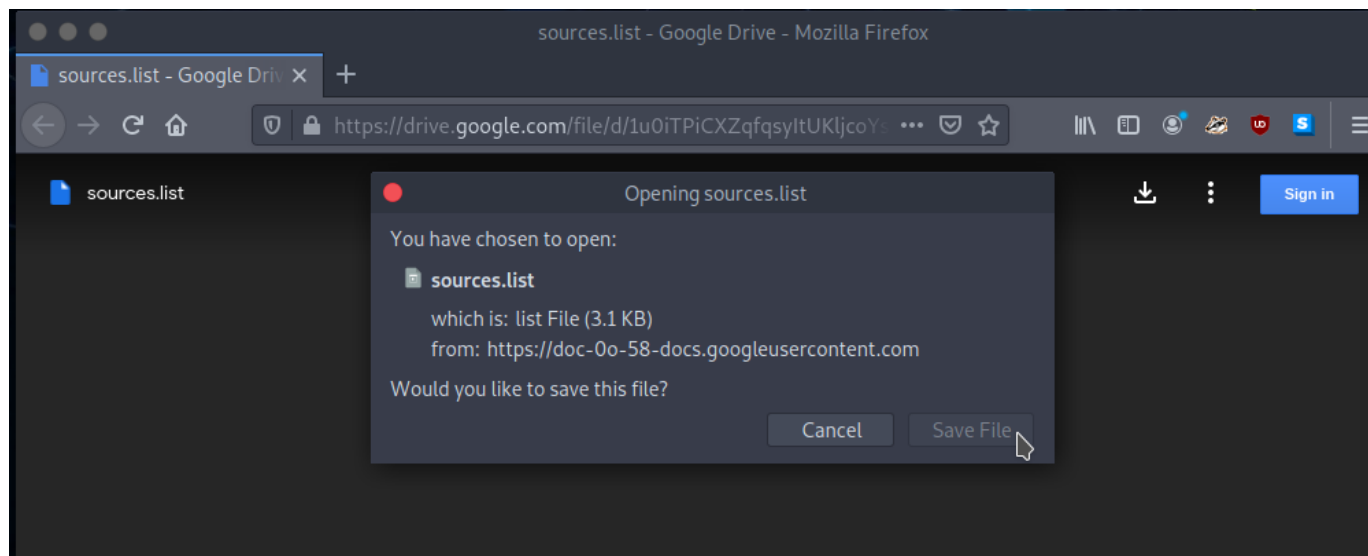
Step 1: Make existing **parrot.list** file as backup file by executing below command

```
[root@parrot-virtual]-[/etc/apt/sources.list.d]
#mv /etc/apt/sources.list.d/parrot.list /etc/apt/sources.list.d/parrot.list.bak
```

Step 2: Remove the already existing repository information from the system. to do that goto **/var/lib/apt/lists/** location and execute **rm***.

```
[root@parrot-virtual]-[/etc/apt/sources.list.d]
#cd /var/lib/apt/lists/
[root@parrot-virtual]-[/var/lib/apt/lists]
#ls
auxfiles
lock
mirror.parrot.sh_mirrors_parrot_dists_rolling_contrib_binary-amd64_Packages
mirror.parrot.sh_mirrors_parrot_dists_rolling_InRelease
mirror.parrot.sh_mirrors_parrot_dists_rolling_main_binary-amd64_Packages
mirror.parrot.sh_mirrors_parrot_dists_rolling_non-free_binary-amd64_Packages
mirror.parrot.sh_mirrors_parrot_dists_rolling-security_InRelease
mirror.parrot.sh_mirrors_parrot_dists_rolling-security_main_binary-amd64_Packages
mirror.parrot.sh_mirrors_parrot_dists_rolling-security_non-free_binary-amd64_Packages
partial
[root@parrot-virtual]-[/var/lib/apt/lists]
#rm *
rm: cannot remove 'auxfiles': Is a directory
rm: cannot remove 'partial': Is a directory
[x]-[root@parrot-virtual]-[/var/lib/apt/lists]
#ls
auxfiles partial
```

Step 3: Download the ubuntu repositories sources.list file from this link: <https://drive.google.com/file/d/1u0iTPiCXZqfqsyItUKljcoYsmDyDoMXo/view>, and save.



Step 4: goto Downloads and copy the sources.list file to `/etc/apt/sources.list.d/parrot.list` using below command.

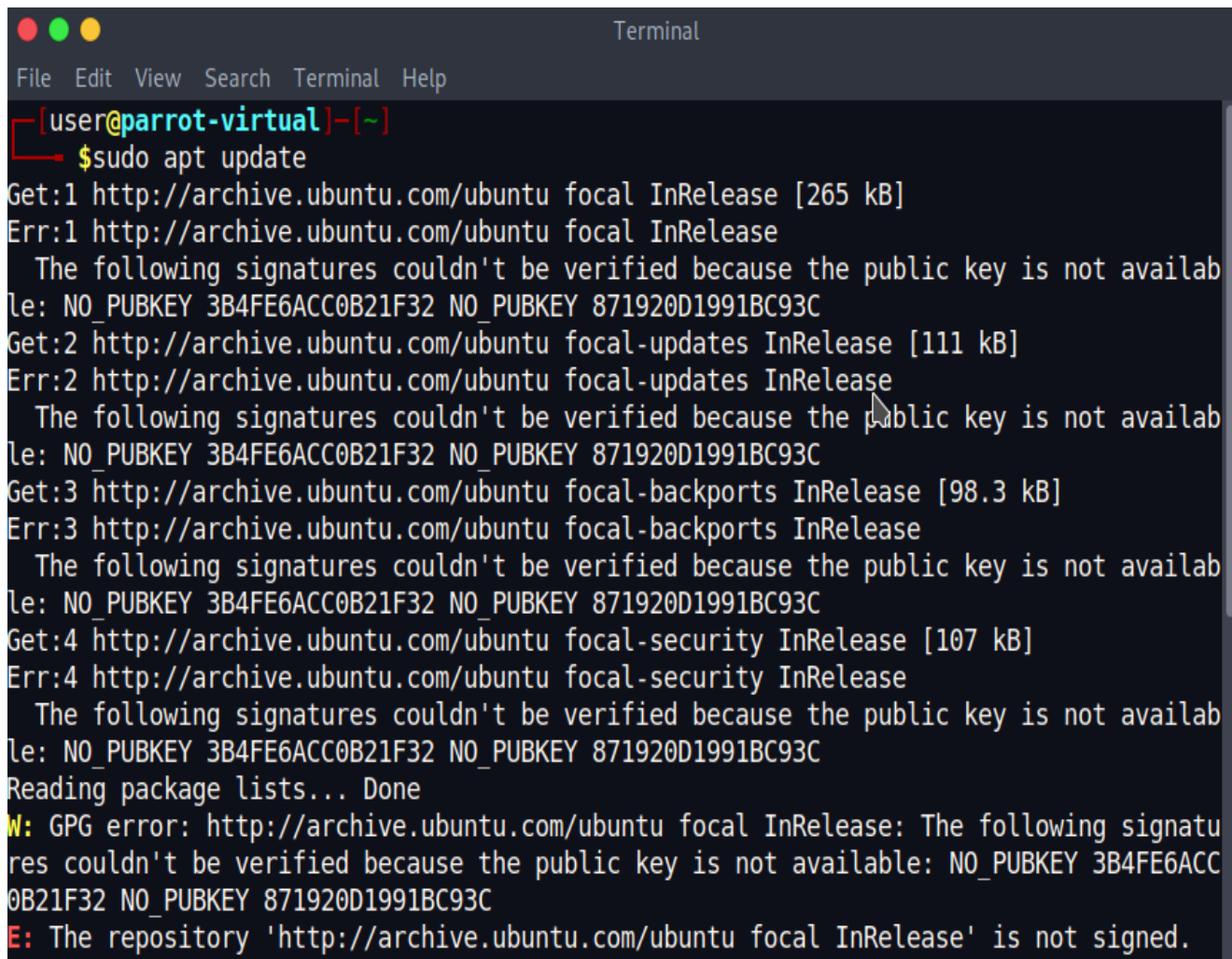
```

Terminal
File Edit View Search Terminal Help
[user@parrot-virtual]-[~]
└─$ cd Downloads/
[user@parrot-virtual]-[~/Downloads]
└─$ ls
nmap-vulners sources.list
[user@parrot-virtual]-[~/Downloads]
└─$ sudo cp sources.list /etc/apt/sources.list.d/parrot.list
[user@parrot-virtual]-[~/Downloads]
└─$ cd /etc/apt/sources.list.d/
[user@parrot-virtual]-[/etc/apt/sources.list.d]
└─$ ls
parrot.list parrot.list.bak

```

Step 5: Execute the following command to update Parrot Linux repository.

- **Command:** `sudo apt update`



```

Terminal
File Edit View Search Terminal Help

[user@parrot-virtual]~$ sudo apt update
Get:1 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Err:1 http://archive.ubuntu.com/ubuntu focal InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0B21F32 NO_PUBKEY 871920D1991BC93C
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Err:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0B21F32 NO_PUBKEY 871920D1991BC93C
Get:3 http://archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Err:3 http://archive.ubuntu.com/ubuntu focal-backports InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0B21F32 NO_PUBKEY 871920D1991BC93C
Get:4 http://archive.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Err:4 http://archive.ubuntu.com/ubuntu focal-security InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0B21F32 NO_PUBKEY 871920D1991BC93C
Reading package lists... Done
W: GPG error: http://archive.ubuntu.com/ubuntu focal InRelease: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 3B4FE6ACC0B21F32 NO_PUBKEY 871920D1991BC93C
E: The repository 'http://archive.ubuntu.com/ubuntu focal InRelease' is not signed.
  
```

we got some public key error while updating, that is because as we added ubuntu repositories to parrot we need to add the public key of repositories to access repositories.

Step 6: To add keys execute the following commands.

`sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys <key>`

replace key with the key value you got in the error.

```

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
3B4FE6ACC0B21F32
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
871920D1991BC93C
  
```

```
[user@parrot-virtual]~$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 3B4FE6ACC0B21F32
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
Executing: /tmp/apt-key-gpghome.I3Nb7tUgTE/gpg.1.sh --keyserver keyserver.ubuntu.com --recv-keys 3B4FE6ACC0B21F32
gpg: key 3B4FE6ACC0B21F32: public key "Ubuntu Archive Automatic Signing Key (2012) <ftpmaster@ubuntu.com>" imported
gpg: Total number processed: 1
gpg: imported: 1

[user@parrot-virtual]~$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 871920D1991BC93C
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
Executing: /tmp/apt-key-gpghome.iEoWrZxNfR/gpg.1.sh --keyserver keyserver.ubuntu.com --recv-keys 871920D1991BC93C
gpg: key 871920D1991BC93C: public key "Ubuntu Archive Automatic Signing Key (2018) <ftpmaster@ubuntu.com>" imported
gpg: Total number processed: 1
gpg: imported: 1

[user@parrot-virtual]~$
```

Step 7: Now if we try to update, it will update successfully.

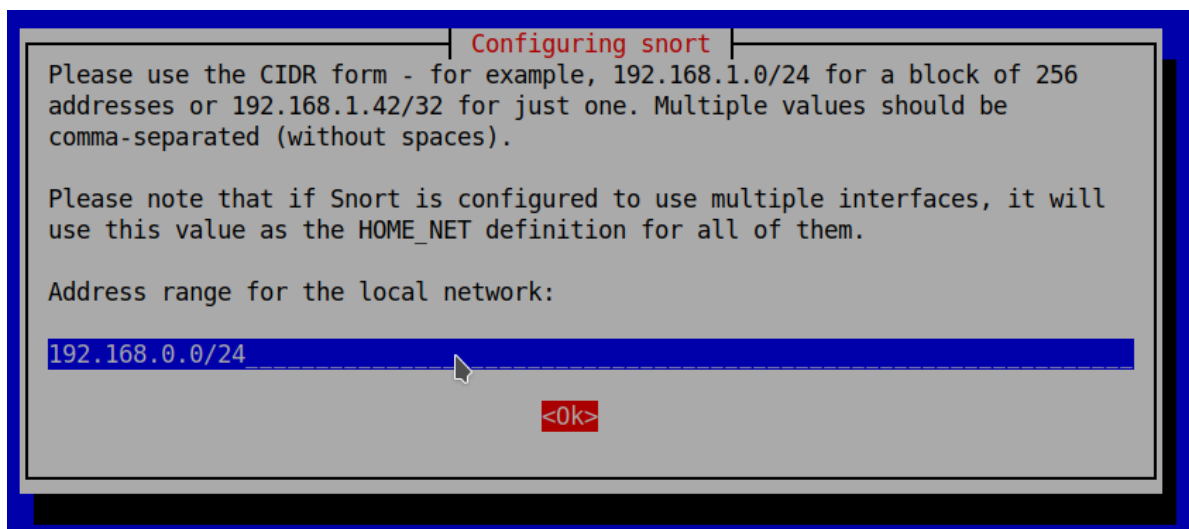
```
[user@parrot-virtual]~$ sudo apt update
Get:1 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [970 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal/main Translation-en_GB [485 kB]
Get:7 http://archive.ubuntu.com/ubuntu focal/main Translation-en [506 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal/restricted amd64 Packages [22.0 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal/restricted Translation-en [6,212 B]
Get:10 http://archive.ubuntu.com/ubuntu focal/restricted Translation-en_GB [3,860 B]
Get:11 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8,628 kB]
Get:12 http://archive.ubuntu.com/ubuntu focal/universe Translation-en_GB [319 kB]
Get:13 http://archive.ubuntu.com/ubuntu focal/universe Translation-en [5,124 kB]
Get:14 http://archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:15 http://archive.ubuntu.com/ubuntu focal/multiverse Translation-en_GB [105 kB]
Get:16 http://archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:17 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [555 kB]
Get:18 http://archive.ubuntu.com/ubuntu focal-updates/main Translation-en [143 kB]
Get:19 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [67.1 kB]
Get:20 http://archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [10.8 kB]
```


Step 8: Install the snort using the following command

- **Command:** apt install snort -y

```
[user@parrot-virtual]~$ sudo apt install snort -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libdaq2 libdumbnet1 oinkmaster snort-common snort-common-libraries
  snort-rules-default
Suggested packages:
  snort-doc
The following NEW packages will be installed:
  libdaq2 libdumbnet1 oinkmaster snort snort-common snort-common-libraries
  snort-rules-default
0 upgraded, 7 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,424 kB of archives.
After this operation, 7,338 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/universe amd64 snort-common-libraries am
d64 2.9.7.0-5build1 [413 kB]
26% [Working]
```

Step 9: While installing snort, we need to provide network range (if you don't know your network range, please click **Ok** without any changes.) In this case, we modified it to **192.168.0.0/24** network range as shown below



Step 10: check snort installed or not with the following command.

- **Command:** Sudo snort --version

```
[user@parrot-virtual]~]
$ sudo snort --version

,,_      -*> Snort! <*-
o"  )~   Version 2.9.7.0 GRE (Build 149)
'    '   By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
         Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
         Copyright (C) 1998-2013 Sourcefire, Inc., et al.
         Using libpcap version 1.9.1 (with TPACKET_V3)
         Using PCRE version: 8.39 2016-06-14
         Using ZLIB version: 1.2.11
```

Step 11: Now remove the ubuntu repositories and place parrot repositories back.

```
[root@parrot-virtual]~]
# rm /etc/apt/sources.list.d/parrot.list
[root@parrot-virtual]~]
# mv /etc/apt/sources.list.d/parrot.list.bak /etc/apt/sources.list.d/parrot.list
[root@parrot-virtual]~]
#
```

Part 2: Configuring Snort

To know the IP address and network interface name, execute **ifconfig**. To make snort act as IDS, we need to modify our IP in snort configuration file according to our requirement.

Step 1: Open the configuration file **/etc/snort/snort.conf** in your favourite text editor (in this case we are opening it in **VIM** editor) and find a line which contains **ipvar HOME_NET any** (Probably that would be line 51)

```
[user@parrot-virtual]~]
$ sudo vim /etc/snort/snort.conf
```

```
# Note to Debian users: this value is overridden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET any
```

Step 2: change ‘any’ into **your IP** (parrot or on which machine you are running snort) press **Esc** on keyboard and type **:wq** and press **enter** to save the changes.

```
46 # Note to Debian users: this value is overridden when starting$
47 # up the Snort daemon through the init.d script by the$
48 # value of DEBIAN_SNORT_HOME_NET s defined in the$
49 # /etc/snort/snort.debian.conf configuration file$
50 #$
51 ipvar HOME_NET 192.168.0.104$
52 $
53 # Set up the external network addresses. Leave as "any" in most situat
54 ipvar EXTERNAL_NET any$
```

Part 3: Running Snort in Parrot Linux to detect intrusions

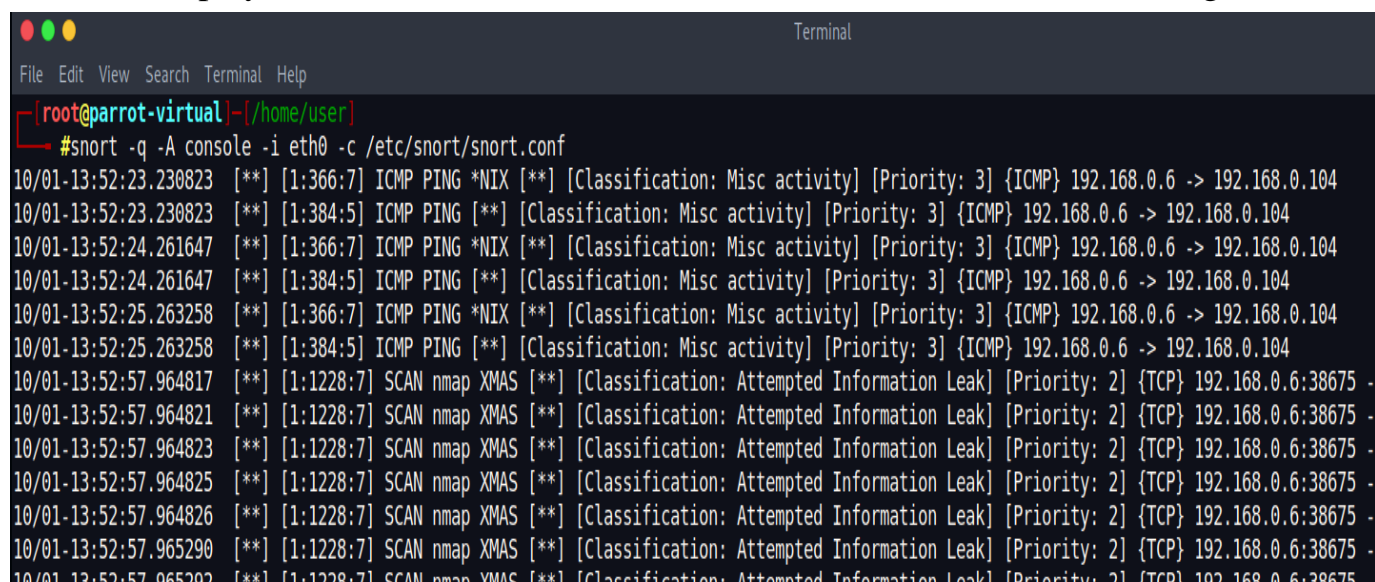
Step 1: Execute the following commands, to start snort

- **sudo /etc/init.d/snort start**
- **sudo snort -q -A console -i eth0 -c /etc/snort/snort.conf**

```
[user@parrot-virtual]~$
$ sudo /etc/init.d/snort start
Starting snort (via systemctl): snort.service.
```

```
[root@parrot-virtual]~/home/user$
# snort -q -A console -i eth0 -c /etc/snort/snort.conf
```

Step 2: Now, snort is capable of detecting malicious traffic based on pre-configured rules and displays attack information on the terminal as shown in the below image.

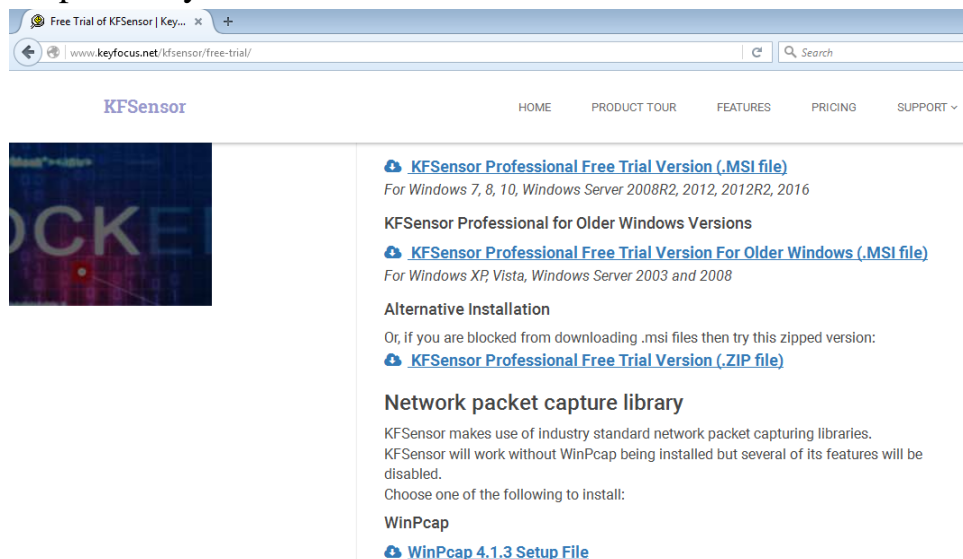


```
Terminal
File Edit View Search Terminal Help
[root@parrot-virtual]~/home/user$
# snort -q -A console -i eth0 -c /etc/snort/snort.conf
10/01-13:52:23.230823  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:23.230823  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:24.261647  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:24.261647  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:25.263258  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:25.263258  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.6 -> 192.168.0.104
10/01-13:52:57.964817  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.964821  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.964823  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.964825  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.964826  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.965290  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
10/01-13:52:57.965292  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.0.6:38675 -> 192.168.0.104
```

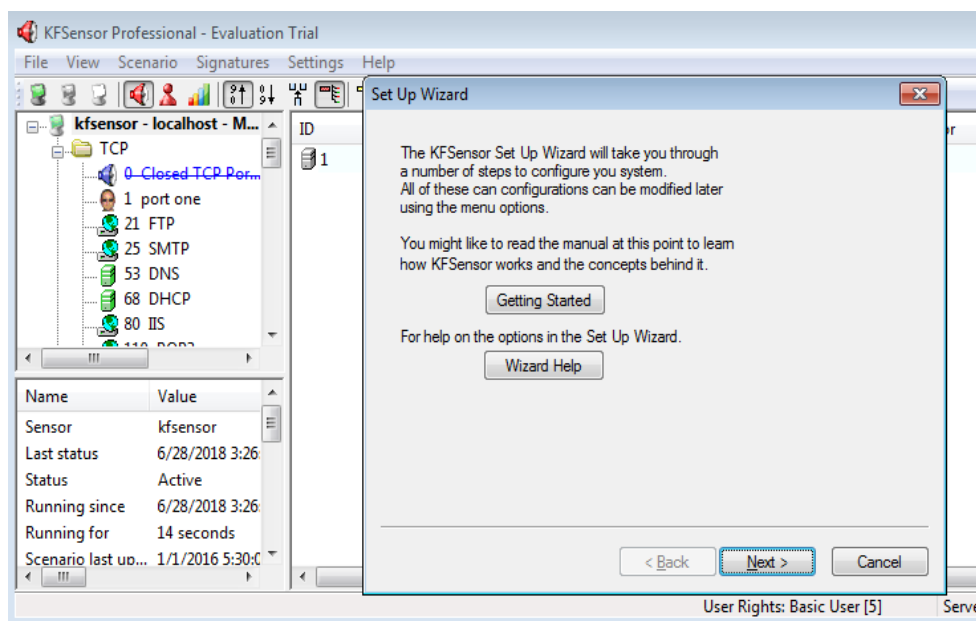

Practical 2: Using KFSensor to build a Honeypot.

Description: In this practical you will learn how to set up a honeypot in a network to mislead the attacker in attacking our systems, by giving fake results to the attacker. And it also gives some alert when it detects any scanning activities on the network.

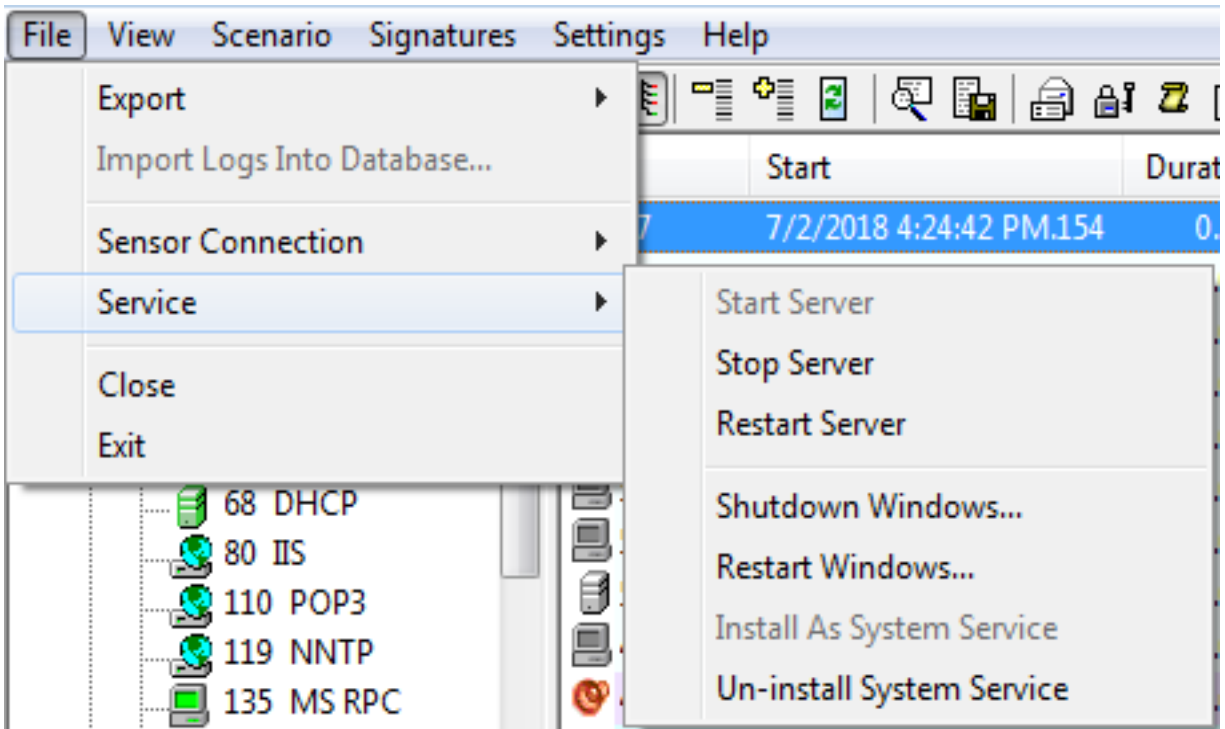
Step 1: Visit KFSensor official website <http://www.keyfocus.net/kfsensor/free-trial/> and register with your details. Download **KFSensor** package and **WinPcap** software package as a dependency to KFSensor.



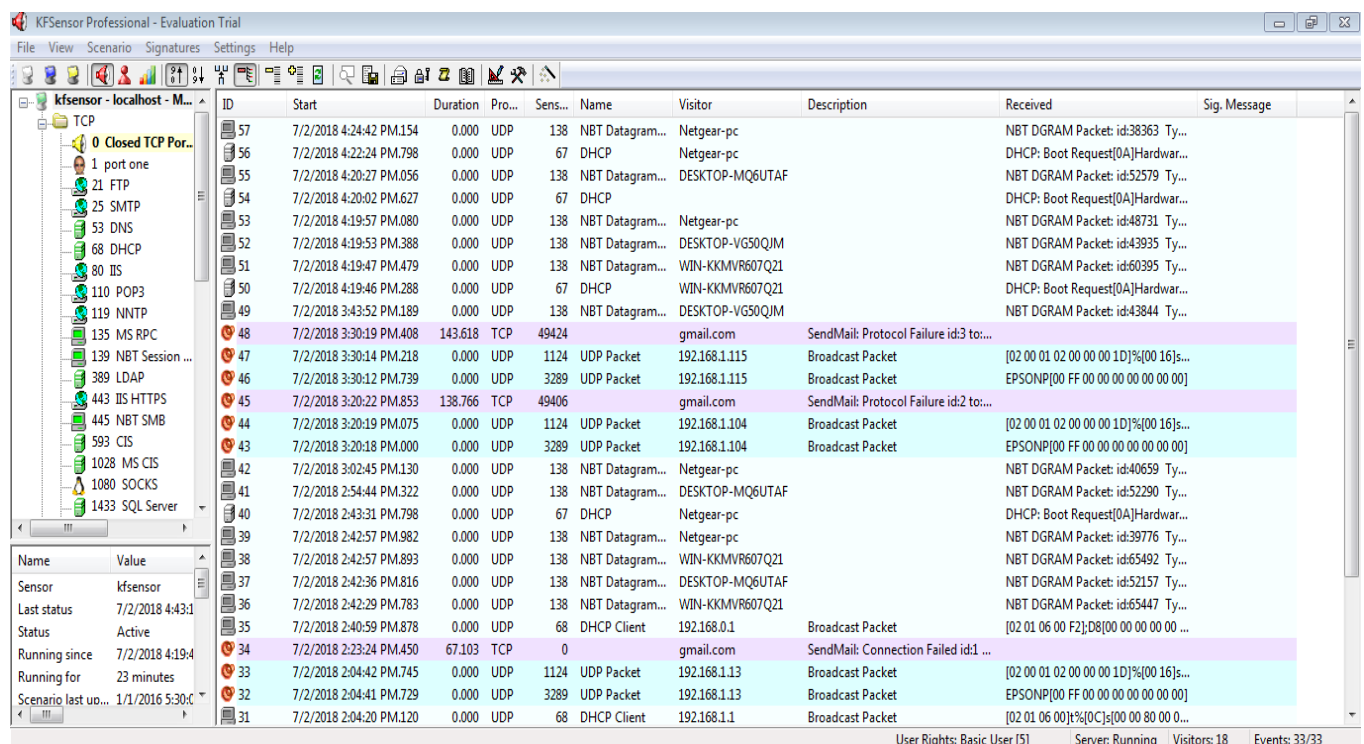
Step 2: Install **WinPcap** first and install **KFSensor** and restart your computer once. After rebooting your computer please go to start menu, find KFSensor and launch the application.



Step 3: Under **File** tab, select **Service** and then click on **Start Service** and proceed with the application wizard to turn your PC into a honeypot machine to attract the attackers.



Step 4: If anyone performs an attack on the computer, KFSensor will display alerts.



Step 5: Right-click on any alert, select **Event Details** to view the information about the alert.

ID	Start	Duration	Pro...	Sens...	Name	Visitor	Description	Received	Sig. Message
57	7/2/2018 4:24:42 PM.154	0.000	UDP	138	NBT Datagram...	Netgear-pc		NBT DGRAM Packet: id:38363 Ty...	
56	7/2/2018 4:22:24 PM.798	0.000	UDP	67	DHCP	Netgear-pc		t Request[0A]Hardwar...	
55	7/2/2018 4:20:27 PM.056	0.000	UDP	138	NBT Datagram...	DESKTOP-MQ6UTAF		M Packet: id:52579 Ty...	
54	7/2/2018 4:20:02 PM.627	0.000	UDP	67	DHCP			t Request[0A]Hardwar...	
53	7/2/2018 4:19:57 PM.080	0.000	UDP	138	NBT Datagram...	Netgear-nr		M Packet: id:48731 Tv...	

Event - 57

Summary

Details

Signature

Data

Event

Sensor ID: kfsensor

Event ID: 57

Start Time: 7/2/2018 4:24:42 PM.154

Severity: Low

Description:

Visitor

IP: 192.168.1.101

Port: 138

Domain: Netgear-pc

Sensor

Name: NBT Datagram Service

Protocol: UDP

Port: 138

Signature

Message:

Request Data - 168 Bytes

NBT DGRAM Packet: id:38363 Type: Direct Group

Source: NETGEAR-PC<20 File Server Service>

Destination: WORKGROUP<1d Master Browser>

SMB: [trans]

name: {}

Expand

Event - 57

Summary

Details

Signature

Data

Event

Sensor ID: kfsensor

Event ID: 57

Start Time: 7/2/2018 4:24:42 PM.154

Type: Connection

End Time: 7/2/2018 4:24:42 PM.154

Severity: Low

Description:

Closed By: Visitor

Limit Exceeded:

Received: 168 Bytes

Response: 0 Bytes

Visitor

IP: 192.168.1.101

Port: 138

Domain: Netgear-pc

Sensor

Name: NBT Datagram Service

IP:

Port: 138

Bound:

Protocol: UDP

Action: SimStdServer

Sim Server:

Create Visitor Rule

Step 6: Right-click on any alert, select **Create Visitor Rule** to create a customized rule to get alerts

ID	Start	Duration	Pro...	Sens...	Name	Visitor	Description	Received	Sig. Message
57	7/2/2018 4:24:42 PM.154	0.000	UDP	138	NBT Datagram...	Netgear-pc		NBT DGRAM Packet: id:38363 Ty...	
56	7/2/2018 4:22:24 PM.798	0.000	UDP	67	DHCP	Netgear-pc		DHCP: Boot Request[0A]Hardwar...	
55	7/2/2018 4:20:27 PM.056	0.000	UDP	138	NBT Datagram...	DESKTOP-MQ6UTAF		NBT DGRAM Packet: id:52579 Ty...	
54	7/2/2018 4:20:02 PM.627	0.000	UDP	67	DHCP			DHCP: Boot Request[0A]Hardwar...	
53	7/2/2018 4:19:57 PM.080	0.000	UDP	138	NBT Datagram...	Netgear-nr		NBT DGRAM Packet: id:48731 Ty...	

Add Visitor Rule [X]

Conditions

Rule Name: NBT Datagram Service 192.168.1.101 port 138

First IP: 192.168.1.101 [Min]

Last IP: [Max]

Host DNS Name: []

Protocol:

- ☐ TCP
- ☒ UDP
- ☐ ICMP
- ☐ WIN
- ☐ Any

Sensor IP: []

Sensor Port: 138

Visitor Port: []

Min Connections: []

Max Connections: []

Actions

Close ☐

Ignore ☐

Set Severity: **No Change** ▼

- No Change
- Low
- Medium
- High

[OK] [Cancel] [Help]

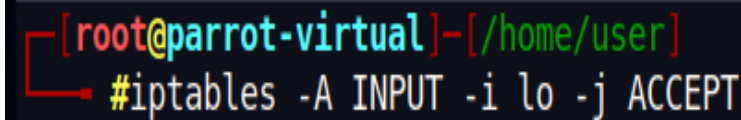
Practical 3: IPtables

Description: In this practical we will learn how to block ICMP requests coming to our system, using **iptables** and how to block traffic any port or IP's from accessing our system. We will learn how to list the iptables rules and delete particular rules from the list.

Part 1: How to block ICMP requests using iptables in Linux

Step 1: To accept the ICMP requests from the loopback. Execute the following command.

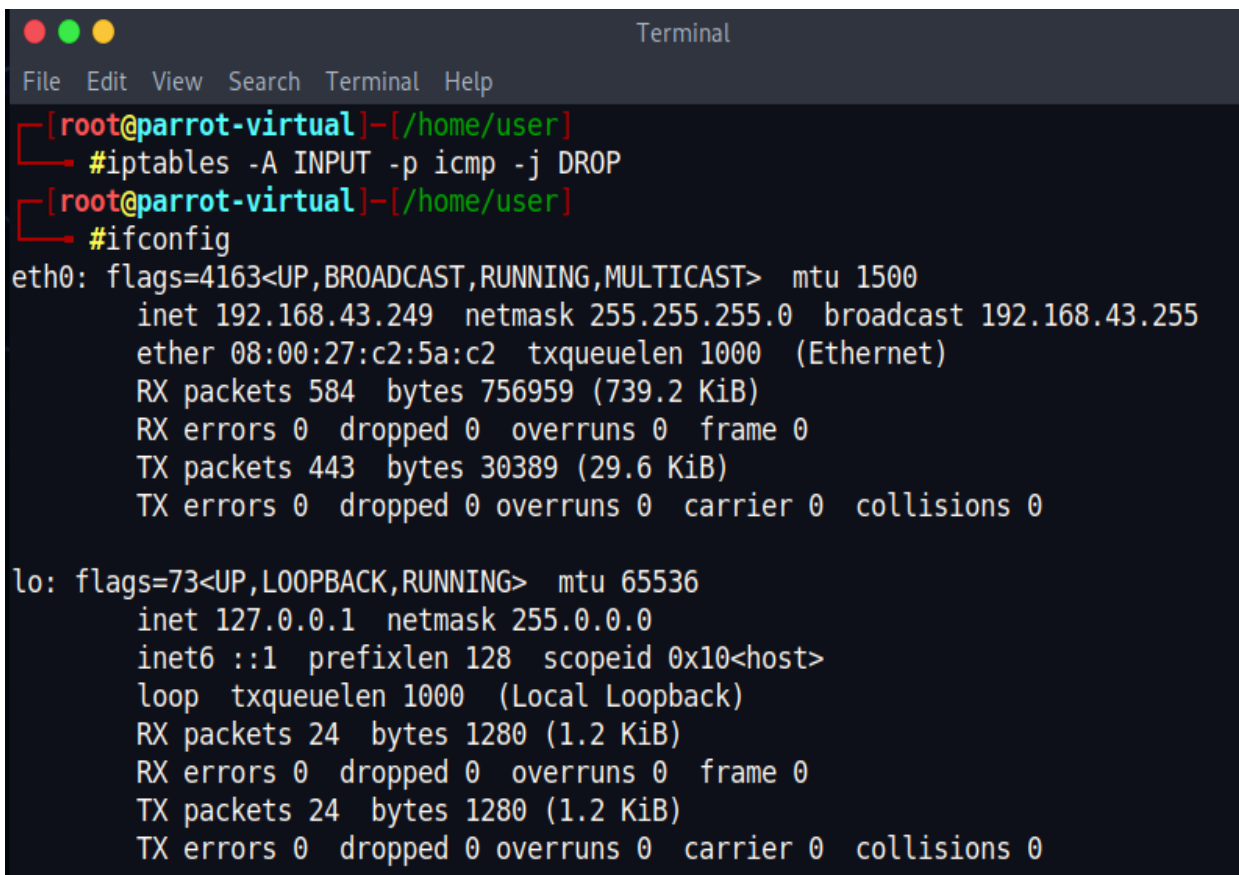
- **Command:** `iptables -A INPUT -i lo -j ACCEPT`



```
[root@parrot-virtual]-[/home/user]
#iptables -A INPUT -i lo -j ACCEPT
```

Step 2: To block ICMP requests from other IP's, execute below command in the terminal.

- **Command:** `iptables -A INPUT -p icmp -j DROP`



```
Terminal
File Edit View Search Terminal Help
[root@parrot-virtual]-[/home/user]
#iptables -A INPUT -p icmp -j DROP
[root@parrot-virtual]-[/home/user]
#ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.43.249 netmask 255.255.255.0 broadcast 192.168.43.255
    ether 08:00:27:c2:5a:c2 txqueuelen 1000 (Ethernet)
    RX packets 584 bytes 756959 (739.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 443 bytes 30389 (29.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 24 bytes 1280 (1.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 24 bytes 1280 (1.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Step 3: If we ping the device from external devices, it does not allow ICMP packets.

```
windows@windows:~$ ping 192.168.43.249
PING 192.168.43.249 (192.168.43.249) 56(84) bytes of data.
^C
--- 192.168.43.249 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4083ms
```

Step 4: If we ping from our system, we can get ICMP reply back

```
[root@parrot-virtual]-[/home/user]
#ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.080 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.053 ms
^C
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2022ms
rtt min/avg/max/mdev = 0.037/0.056/0.080/0.017 ms
```

Part 2: How to block specific port in Linux using IPtables

Step 1: To block access to a particular port, from other IP's using iptables, execute the following command in the terminal. Here we block the port 80.

- **Command:** iptables -A INPUT -p tcp --dport 80 -j DROP

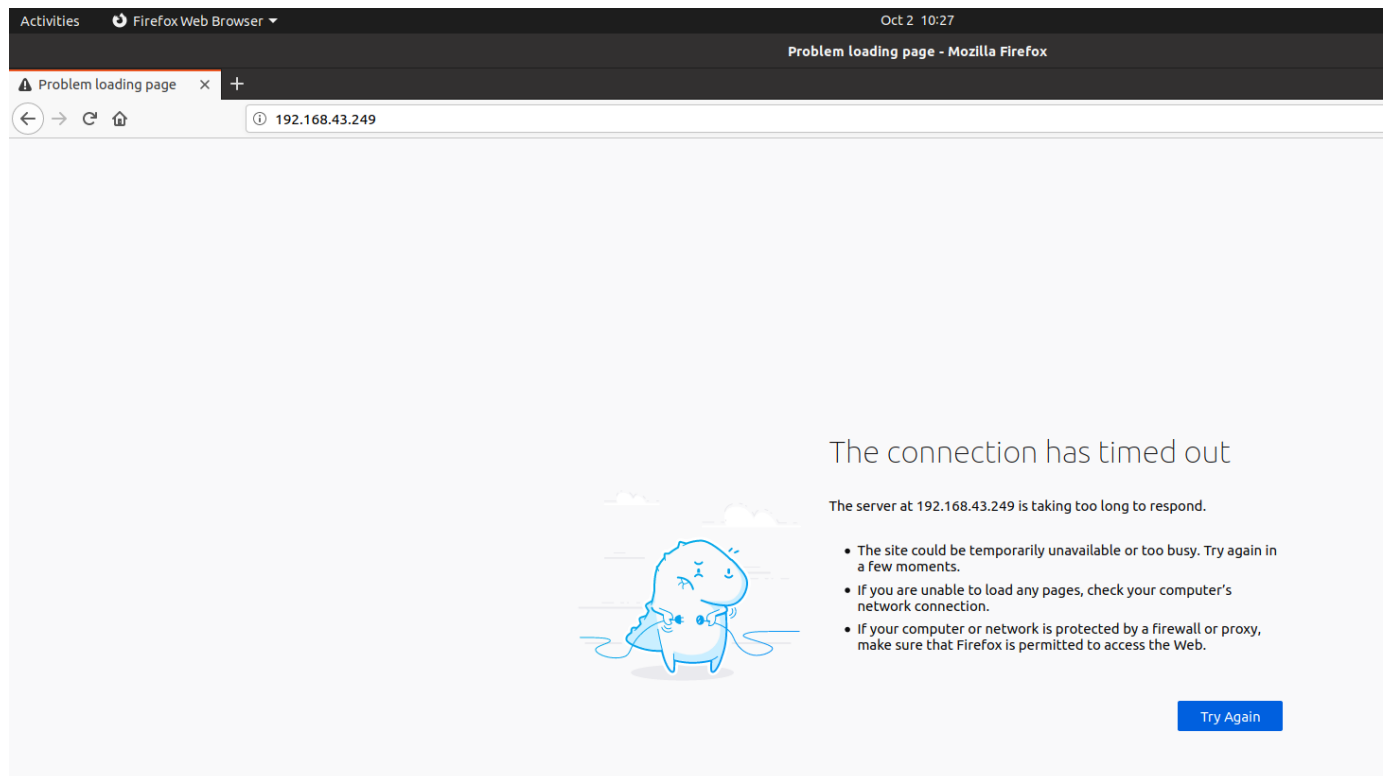
```
[root@parrot-virtual]-[/home/user]
#iptables -A INPUT -p tcp --dport 80 -j DROP
```


Step 2: Start the service on the blocked port and check the service status.

```
[root@parrot-virtual]-[/home/user]
#service apache2 start
[root@parrot-virtual]-[/home/user]
#service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-02 05:55:01 BST; 4s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 1859 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 1873 (apache2)
    Tasks: 6 (limit: 2298)
   Memory: 28.5M
    CGroup: /system.slice/apache2.service
           └─1873 /usr/sbin/apache2 -k start
             └─1875 /usr/sbin/apache2 -k start
               └─1876 /usr/sbin/apache2 -k start
                 └─1877 /usr/sbin/apache2 -k start
                   └─1878 /usr/sbin/apache2 -k start
                     └─1879 /usr/sbin/apache2 -k start

Oct 02 05:55:01 parrot-virtual systemd[1]: Starting The Apache HTTP Server...
Oct 02 05:55:01 parrot-virtual apachectl[1869]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, because the IP address 192.168.43.249 has not been assigned an IPv4 or IPv6 address yet.
Oct 02 05:55:01 parrot-virtual systemd[1]: Started The Apache HTTP Server.
lines 1-19/19 (END)
```

Step 3: If we try to access 80 port from an external device, iptables blocks the request.



Step 4: If we want to block any request from particular IP, execute the below command

- **Command:** iptables -A INPUT -s <IP address> -j DROP

```
[root@parrot-virtual]-[/home/user]
#iptables -A INPUT -s 192.168.43.247 -j DROP
```

Step 5: If we want block requests from a series of IP addresses execute the below command.

- **Syntax:** iptables -A INPUT -m iprange --src-range <ip range> -j DROP
- **Command:** iptables -A INPUT -m iprange --src-range 192.168.1.5-192.168.1.25 -j DROP

```
[root@parrot-virtual]-[/home/user]
#iptables -A INPUT -m iprange --src-range 192.168.43.5-192.168.43.25 -j DROP
```

Step 6: To list out the rules added in the IPtables, execute the below command.

- **Command:** iptables -L --line-numbers

```
[root@parrot-virtual]-[/home/user]
#iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num target      prot opt source                destination
1  ACCEPT        all  --  anywhere                anywhere
2  DROP          icmp --  anywhere                anywhere
3  DROP          tcp  --  anywhere                anywhere    tcp dpt:http
4  DROP          all  --  windows                anywhere
5  DROP          all  --  anywhere                anywhere    source IP range
192.168.43.5-192.168.43.25

Chain FORWARD (policy ACCEPT)
num target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
num target      prot opt source                destination
```

Step 7: To delete any rule in IPtables by using the rule number, execute the below command.

- **Syntax:** iptables -D INPUT <rule number>
- **Command:** iptables -D INPUT 3

```
[root@parrot-virtual]~[/home/user]
#iptables -D INPUT 3
[root@parrot-virtual]~[/home/user]
#iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target      prot opt source                destination
1    ACCEPT      all  --  anywhere              anywhere
2    DROP        icmp --  anywhere              anywhere
3    DROP        all  --  windows              anywhere
4    DROP        all  --  anywhere              anywhere          source IP range
192.168.43.5-192.168.43.25

Chain FORWARD (policy ACCEPT)
num  target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
num  target      prot opt source                destination
[root@parrot-virtual]~[/home/user]
#
```

- When any rule is deleted the next rules will move up. We can compare that with the previous screenshot.

Practical 4: Installation and usage of Pentbox

Description: In this practical we will learn how to configure and use the pentbox tool as a honey pot.

Step 1: In terminal execute the below command to clone the pentbox tool from GitHub

- **Command:** git clone https://github.com/H4CK3RT3CH/pentbox-1.8.git

```
[user@parrot-virtual]-[/opt/test]
$ sudo git clone https://github.com/H4CK3RT3CH/pentbox-1.8.git
[sudo] password for user:
Cloning into 'pentbox-1.8'...
remote: Enumerating objects: 306, done.
remote: Total 306 (delta 0), reused 0 (delta 0), pack-reused 306
Receiving objects: 100% (306/306), 808.04 KiB | 537.00 KiB/s, done.
Resolving deltas: 100% (113/113), done.
```

Step 2: After completion of cloning, List the files present in the **pentbox-1.8** directory.

```
[user@parrot-virtual]-[/opt/test]
$ ls
pentbox-1.8
[user@parrot-virtual]-[/opt/test]
$ cd pentbox-1.8/
[user@parrot-virtual]-[/opt/test/pentbox-1.8]
$ ls
changelog.txt  lib      pb_update.rb  readme.txt  tools
COPYING.txt   other    pentbox.rb    todo.txt
[user@parrot-virtual]-[/opt/test/pentbox-1.8]
$
```

Step 3: To start and configure the pentbox tool, execute the below command in the terminal And select the option **2** (Network tools).

- **Command:** ./pentbox.rb

```
[user@parrot-virtual]~/opt/test/pentbox-1.8
$ sudo ./pentbox.rb

PentBox 1.8

  ____  _.'@@@@@'.
 |__| (@@@@@@@@@)
      (@@@@@@@@@)
  `YY~--YY'
    ||    ||

----- Menu          ruby2.7.1 @ x86_64-linux-gnu

1- Cryptography tools
2- Network tools
3- Web
4- Ip grabber
5- Geolocation ip
6- Mass attack
7- License and contact
8- Exit

-> 2
```

Step 4: After selection of Network tools, It will list different options under network tools and select option 3 (Honeypot).

```
6- Mass attack
7- License and contact
8- Exit

-> 2

1- Net DoS Tester
2- TCP port scanner
3- Honeypot
4- Fuzzer
5- DNS and host gathering
6- MAC address geolocation (samy.pl)
0- Back

-> 3

// Honeypot //

You must run PentBox with root privileges.

Select option.

1- Fast Auto Configuration
2- Manual Configuration [Advanced Users, more options]

-> 2
```

Step 5: After selection of the Honeypot, select **2** (manual configuration).

```
6- Mass attack
7- License and contact
8- Exit

-> 2

1- Net DoS Tester
2- TCP port scanner
3- Honeypot
4- Fuzzer
5- DNS and host gathering
6- MAC address geolocation (samy.pl)
0- Back

-> 3
```

```
// Honeypot //
```

```
You must run PentBox with root privileges.
```

```
Select option.
```

```
1- Fast Auto Configuration
2- Manual Configuration [Advanced Users, more options]

-> 2
```

Step 6: Give the port number on which we want to run service to mislead attackers

```
Insert port to Open.
```

```
-> 4567
```

Step 7: If we want to give any false message, who tries to access that port. Enter that message in the given section.

```
Insert false message to show.
```

```
-> I caught you, don't try to intrude into my system
```

Step 8: To save the log of IPs who access or tries to access the port, enter **y**, otherwise **n**.

```
Save a log with intrusions?  
(y/n)    -> y  
Log file name? (incremental)
```

Step 9: Provide the specific location to save the logs, or press Enter to continue with the default location.

```
Default: */pentbox/other/log_honeypot.txt  
->
```

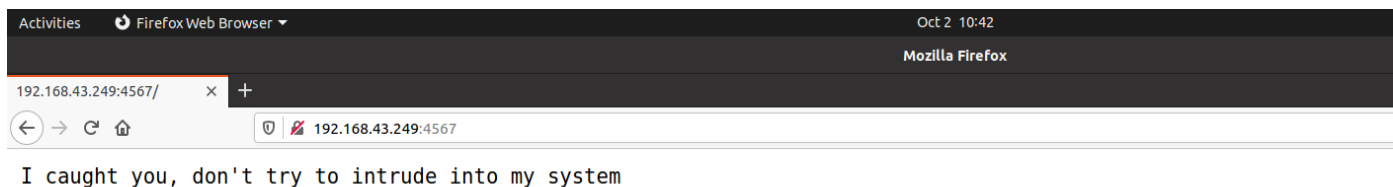
Step 10: If we want beep sound whenever one tries to access the port select **y**.

```
Activate beep() sound when intrusion?  
(y/n)    -> y
```

Step 11: Now honeypot will be activated on the specified port

```
HONEYPOT ACTIVATED ON PORT 4567 (2020-10-02 06:11:39 +0100)
```

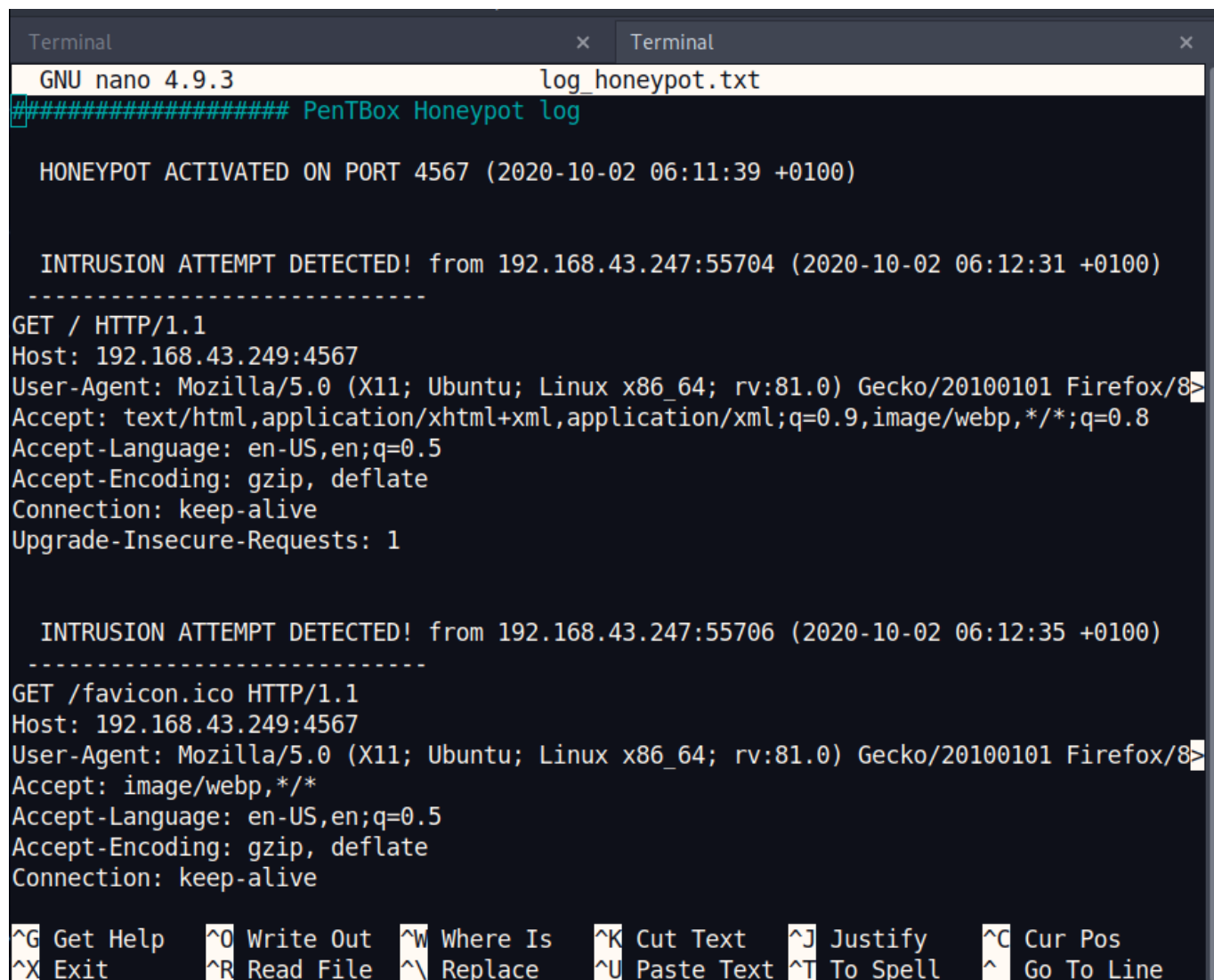
Step 12: If anyone tries to access that port it will show the false message we specified.



Step 13: It also logs the IP details to the log file in the path specified path.

```
[user@parrot-virtual]-[/opt/test/pentbox-1.8]
└─$ cd other/
[user@parrot-virtual]-[/opt/test/pentbox-1.8/other]
└─$ ls
hosts.txt  http_dirs.txt  log  log_honeypot.txt  pentbox-wlist.txt
[user@parrot-virtual]-[/opt/test/pentbox-1.8/other]
└─$ nano log_honeypot.txt
```

Step 14: Open the log file we can find the IP details of anyone who tries to access the port



```
GNU nano 4.9.3 log_honeypot.txt
##### PentBox Honeypot log

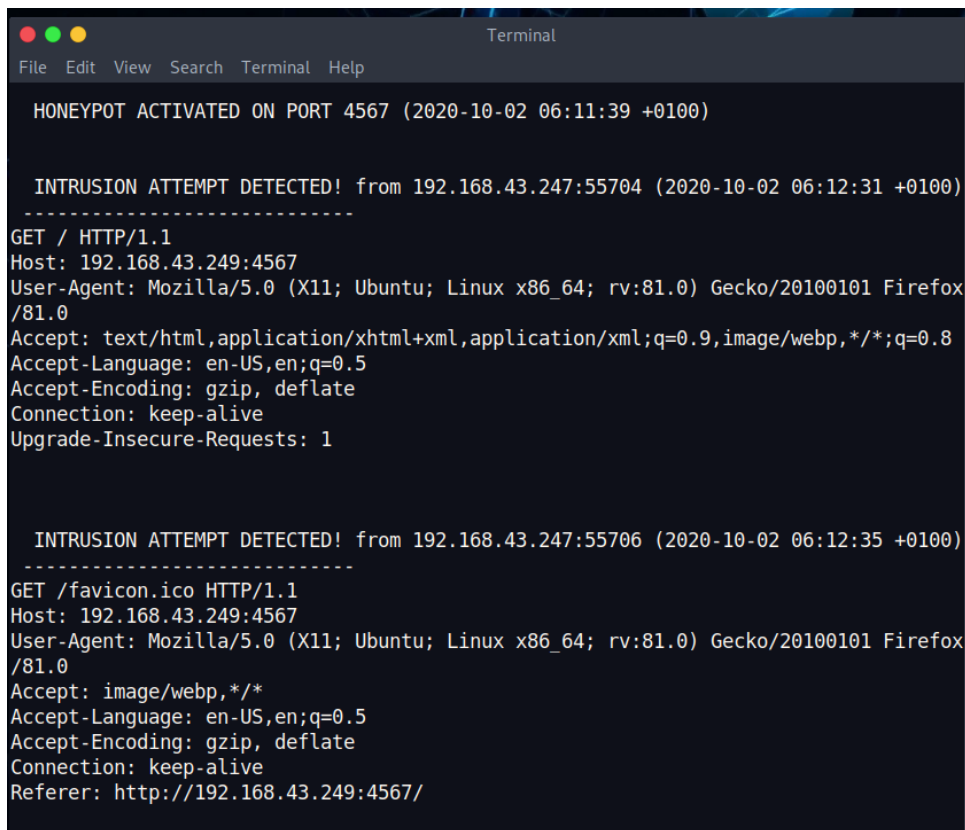
HONEYPOT ACTIVATED ON PORT 4567 (2020-10-02 06:11:39 +0100)

INTRUSION ATTEMPT DETECTED! from 192.168.43.247:55704 (2020-10-02 06:12:31 +0100)
-----
GET / HTTP/1.1
Host: 192.168.43.249:4567
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/8
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

INTRUSION ATTEMPT DETECTED! from 192.168.43.247:55706 (2020-10-02 06:12:35 +0100)
-----
GET /favicon.ico HTTP/1.1
Host: 192.168.43.249:4567
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/8
Accept: image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```


Step 15: We can also see the log details in the Honeypot terminal also

A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help) and three colored window control buttons (red, yellow, green) in the top-left corner. The terminal displays the following text:

```
HONEYPOT ACTIVATED ON PORT 4567 (2020-10-02 06:11:39 +0100)

INTRUSION ATTEMPT DETECTED! from 192.168.43.247:55704 (2020-10-02 06:12:31 +0100)
-----
GET / HTTP/1.1
Host: 192.168.43.249:4567
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

INTRUSION ATTEMPT DETECTED! from 192.168.43.247:55706 (2020-10-02 06:12:35 +0100)
-----
GET /favicon.ico HTTP/1.1
Host: 192.168.43.249:4567
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.43.249:4567/
```