

CSE 406

Computer Security

Project Final Report

DoS attack to the DNS server (using spoofed IP address)

Name : Abdur Rahman Fahad
Std ID : 1605069
Group : 02
Lab Group : B1

Steps of the attack

DoS attack (Denial-of-Service attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled.

Here we will attack the DNS (Domain Name Server) using spoofed IP address. We will perform the attack by sending lots of meaningless DNS query to the DNS server.

Here are the steps of the attack:

Setting up the Environment

For the simulation of this attack, we need 3 machines on the same network.

1. A Local DNS Server (which is to be attacked).
2. The attacker machine.
3. The victim User on the network to test the attack result.

So, at first I created 3 virtual machines (each Ubuntu 16.04 LTS) on Oracle VM Virtual Box using network type as Bridged Adapter so that, all the virtual machines were at the same network with host OS.

Setting up Local DNS Server

I wrote the following commands on terminal:

```
$ sudo apt-get install bind9
```

In this way I configured my DNS Server with **Bind9** on a virtual machine.

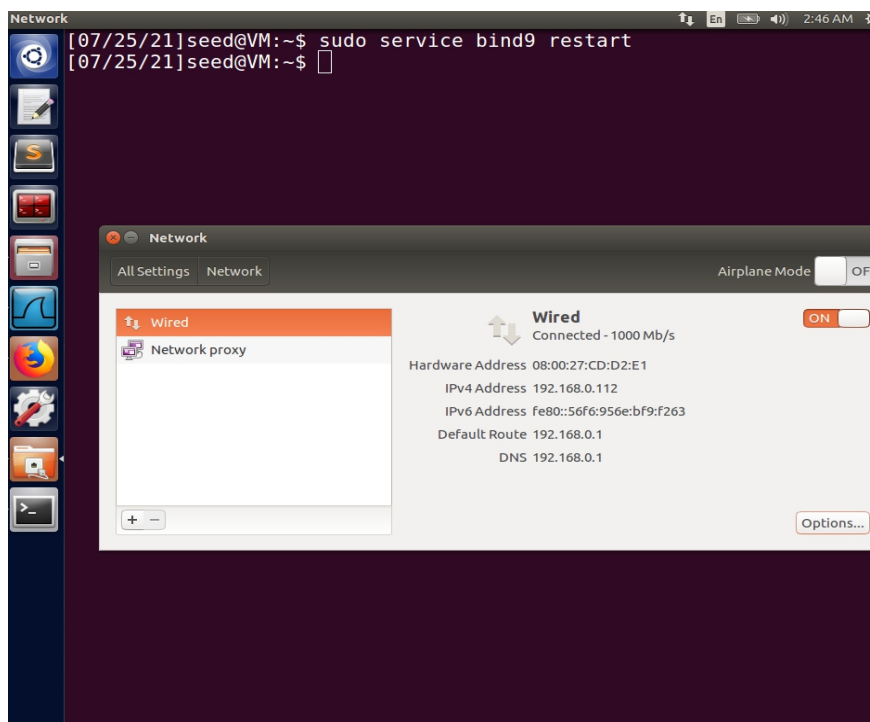
Then I wrote the commands in the terminal:

1. To flush the DNS cache,

```
$ sudo rndc flush
```

2. To start newly configured DNS Server:

```
$ sudo service bind9 restart
```



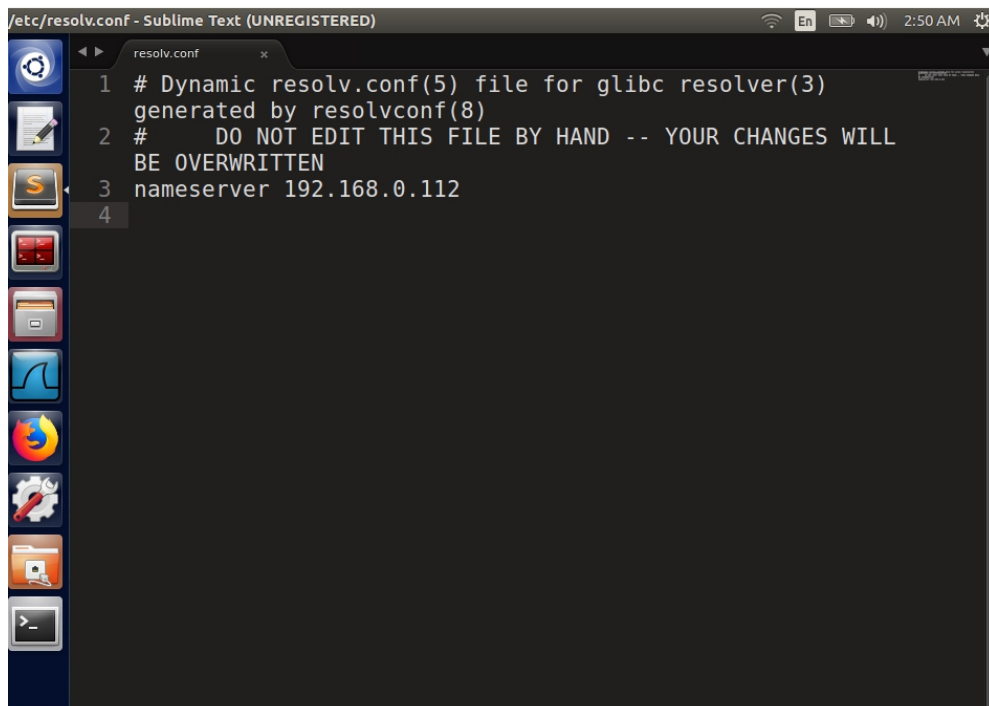
Setting up victim's DNS Server

To change victim's DNS server address, I opened `/etc/resolv.conf` file and changed the line

`nameserver 127.0.1.1`

to

`nameserver 192.168.0.112` (DNS Server's IP)

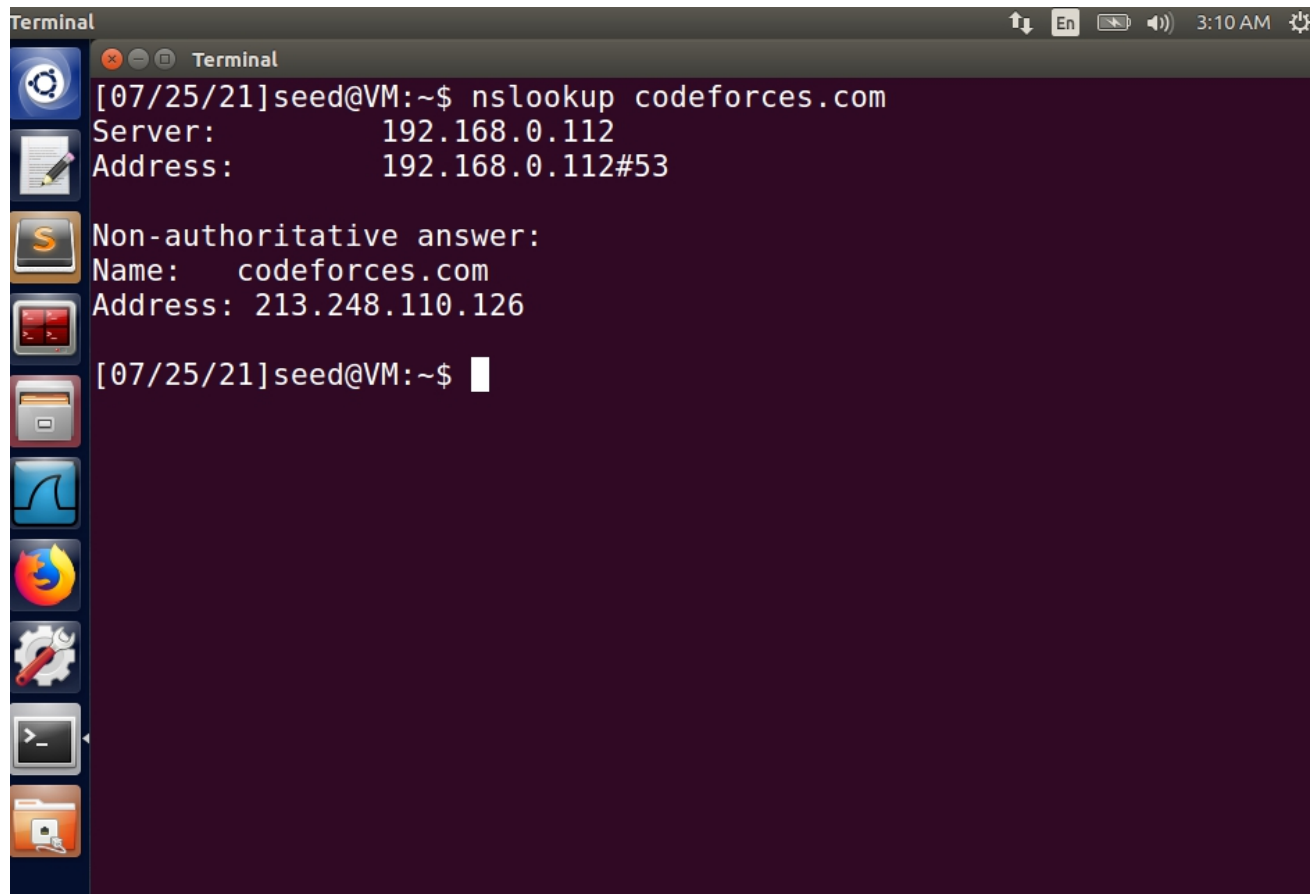


```
/etc/resolv.conf - Sublime Text (UNREGISTERED)
1 # Dynamic resolv.conf(5) file for glibc resolver(3)
  generated by resolvconf(8)
2 #     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL
  BE OVERWRITTEN
3 nameserver 192.168.0.112
4
```

Testing my new DNS Server

Now I'll be checking if the DNS Server is working perfectly before the attack.

To do that I'll run DNS queries from the user's machine which will be resolved by the DNS server.

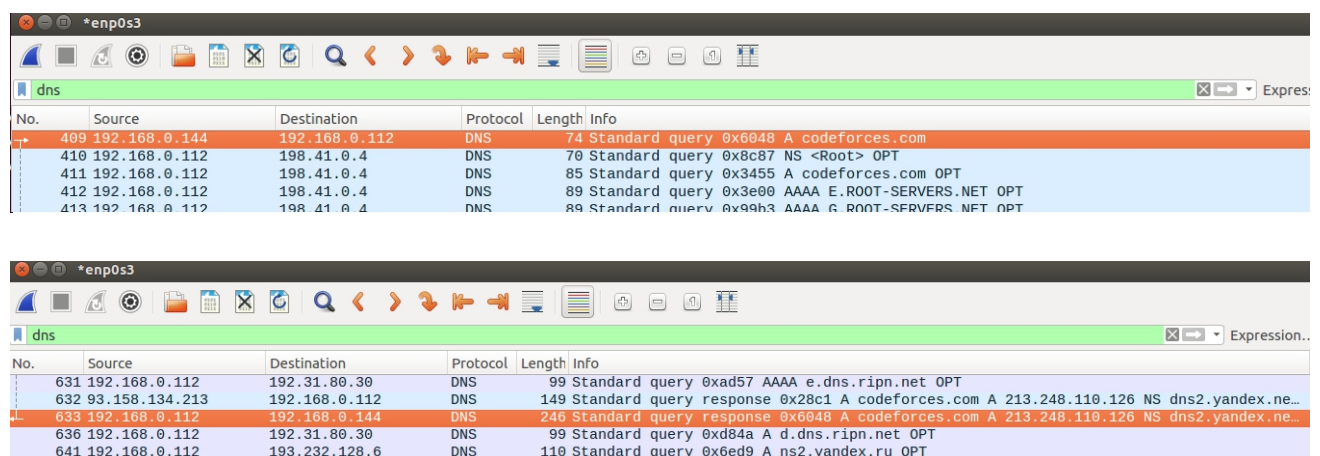


```
Terminal
[07/25/21]seed@VM:~$ nslookup codeforces.com
Server:                192.168.0.112
Address:               192.168.0.112#53

Non-authoritative answer:
Name:   codeforces.com
Address: 213.248.110.126

[07/25/21]seed@VM:~$
```

Query Answer from DNS Server (192.168.0.112)



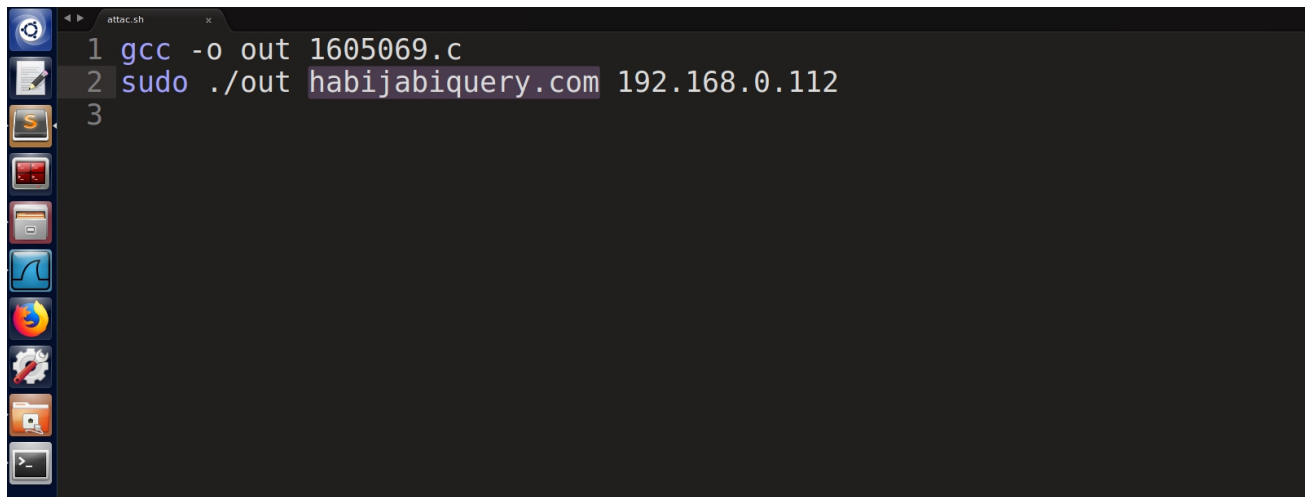
No.	Source	Destination	Protocol	Length	Info
409	192.168.0.144	192.168.0.112	DNS	74	Standard query 0x6048 A codeforces.com
410	192.168.0.112	198.41.0.4	DNS	70	Standard query 0x8c87 NS <Root> OPT
411	192.168.0.112	198.41.0.4	DNS	85	Standard query 0x3455 A codeforces.com OPT
412	192.168.0.112	198.41.0.4	DNS	89	Standard query 0x3e00 AAAA E.ROOT-SERVERS.NET OPT
413	192.168.0.112	198.41.0.4	DNS	89	Standard query 0x99b3 AAAA G.ROOT-SERVERS.NET OPT

No.	Source	Destination	Protocol	Length	Info
631	192.168.0.112	192.31.80.30	DNS	99	Standard query 0xad57 AAAA e.dns.ripn.net OPT
632	93.158.134.213	192.168.0.112	DNS	149	Standard query response 0x28c1 A codeforces.com A 213.248.110.126 NS dns2.yandex.ne...
633	192.168.0.112	192.168.0.144	DNS	246	Standard query response 0x6048 A codeforces.com A 213.248.110.126 NS dns2.yandex.ne...
636	192.168.0.112	192.31.80.30	DNS	99	Standard query 0xd84a A d.dns.ripn.net OPT
641	192.168.0.112	193.232.128.6	DNS	110	Standard query 0x6ed9 A ns2.yandex.ru OPT

Query observed from Server's Wireshark

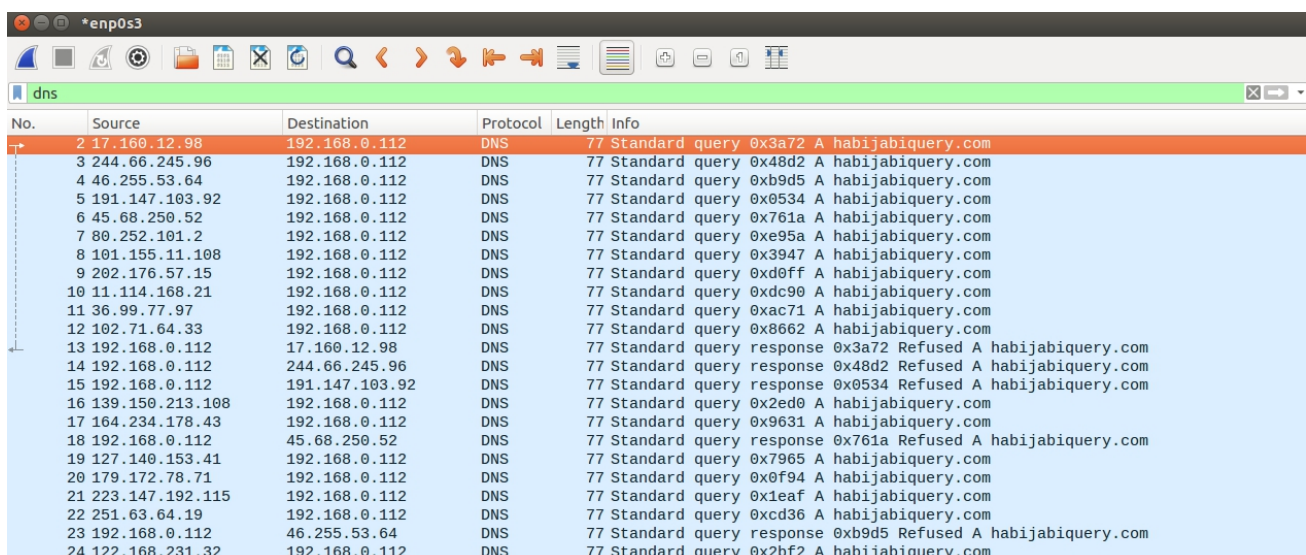
Initiating Attack and Observed Output

With my attack code I sent thousands of random bogus query (`habijabiquery.com`) to the DNS Server (`192.168.0.112`)



```
1 gcc -o out 1605069.c
2 sudo ./out habijabiquery.com 192.168.0.112
3
```

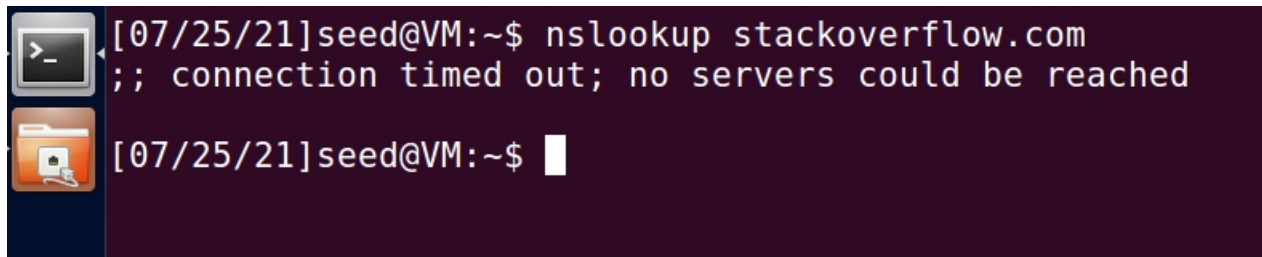
Then I observed those queries flooding the DNS Server



No.	Source	Destination	Protocol	Length	Info
2	17.160.12.98	192.168.0.112	DNS	77	Standard query 0x3a72 A habijabiquery.com
3	244.66.245.96	192.168.0.112	DNS	77	Standard query 0x48d2 A habijabiquery.com
4	46.255.53.64	192.168.0.112	DNS	77	Standard query 0xb9d5 A habijabiquery.com
5	191.147.103.92	192.168.0.112	DNS	77	Standard query 0x0534 A habijabiquery.com
6	45.68.250.52	192.168.0.112	DNS	77	Standard query 0x761a A habijabiquery.com
7	80.252.101.2	192.168.0.112	DNS	77	Standard query 0xe95a A habijabiquery.com
8	101.155.11.108	192.168.0.112	DNS	77	Standard query 0x3947 A habijabiquery.com
9	202.176.57.15	192.168.0.112	DNS	77	Standard query 0xd0ff A habijabiquery.com
10	11.114.168.21	192.168.0.112	DNS	77	Standard query 0xdc90 A habijabiquery.com
11	36.99.77.97	192.168.0.112	DNS	77	Standard query 0xac71 A habijabiquery.com
12	102.71.64.33	192.168.0.112	DNS	77	Standard query 0x8662 A habijabiquery.com
13	192.168.0.112	17.160.12.98	DNS	77	Standard query response 0x3a72 Refused A habijabiquery.com
14	192.168.0.112	244.66.245.96	DNS	77	Standard query response 0x48d2 Refused A habijabiquery.com
15	192.168.0.112	191.147.103.92	DNS	77	Standard query response 0x0534 Refused A habijabiquery.com
16	139.150.213.108	192.168.0.112	DNS	77	Standard query 0x2ed0 A habijabiquery.com
17	164.234.178.43	192.168.0.112	DNS	77	Standard query 0x9631 A habijabiquery.com
18	192.168.0.112	45.68.250.52	DNS	77	Standard query response 0x761a Refused A habijabiquery.com
19	127.140.153.41	192.168.0.112	DNS	77	Standard query 0x7965 A habijabiquery.com
20	179.172.78.71	192.168.0.112	DNS	77	Standard query 0xf94 A habijabiquery.com
21	223.147.192.115	192.168.0.112	DNS	77	Standard query 0x1eaf A habijabiquery.com
22	251.63.64.19	192.168.0.112	DNS	77	Standard query 0xcd36 A habijabiquery.com
23	192.168.0.112	46.255.53.64	DNS	77	Standard query response 0xb9d5 Refused A habijabiquery.com
24	122.168.231.32	192.168.0.112	DNS	77	Standard query 0x2bf2 A habijabiquery.com

It can be seen that the source addresses of the queries are random and not actual IP address of the attacker, thus IP spoofing was done.

Now we can see that the victim's query couldn't be resolved as the server ran out of resources, it was unreachable.

A terminal window with a dark purple background. On the left side, there are two icons: a terminal icon and a folder icon. The terminal icon is a small square with a white border and a black background, containing a white prompt character '>'. The folder icon is a small square with a white border and a dark background, containing a white folder icon. The text in the terminal is white. The first line shows the command '[07/25/21]seed@VM:~\$ nslookup stackoverflow.com' followed by the output ';; connection timed out; no servers could be reached'. The second line shows the prompt '[07/25/21]seed@VM:~\$' followed by a white cursor block.

```
[07/25/21]seed@VM:~$ nslookup stackoverflow.com
;; connection timed out; no servers could be reached

[07/25/21]seed@VM:~$
```

So the attack was successful.

Comment on Attack Success

We sent bogus queries to DNS Server with a spoofed IP address in the source IP address field of IP Header. It failed to find a valid entry in cache and so, the DNS server sent the query to authoritative DNS Servers and waited for the result, which also eventually failed.

By doing this with many infinite loop and many more requests than usual, eventually the cache of DNS server was filled with bad requests. This way, it was possible to flood the targeted DNS and the server will denied any further service from any legit user and our attack was successful.