**Dos Attack To The DNS Server With Spoofed IP:**

**Student-id: 1505107**

**Source files:**

1. **DNS\_QueryBuilder.py :** This is a query builder. Given the domain name and dns server ip and port it will construct a dns query according to RFC-1035. This will return the query in hexadecimal that can be directly send using udp sockets and also if it is provided with the dns response, it can analyze the response and tell the ip address of the desired domain.
2. **dos.py :** This is the main attack tool. Inside it we are constructing packets, the payload is the dns query made using DNS\_QueryBuilder.py. Inside an infinite while loop we will continuously send dns query thus launch a dns flood attack.

**Attack Steps:**

1. We make a dns query and fix our target, make a packet.
2. Then we enter an infinite while loop.
3. As the packets are sent in an infinite loop, a keyboard interrupt catching code has been set up. If we use ‘Ctrl+c’ the code will stop executing and before terminating it will print the number of packets sent so far.

**Snapshots of the attack:**

1. **dns query:** here we test the DNS\_QueryBuilder.py. In test.py we simply take an udp socket and using DNS\_QueryBuilder.py construct a query to get ip of the domain “codeforces.com”. After sending it we wait for a udp response and analyze it using DNS\_QueryBuilder.py. For convenience the tester code has been given in a separate “tester codes” folder

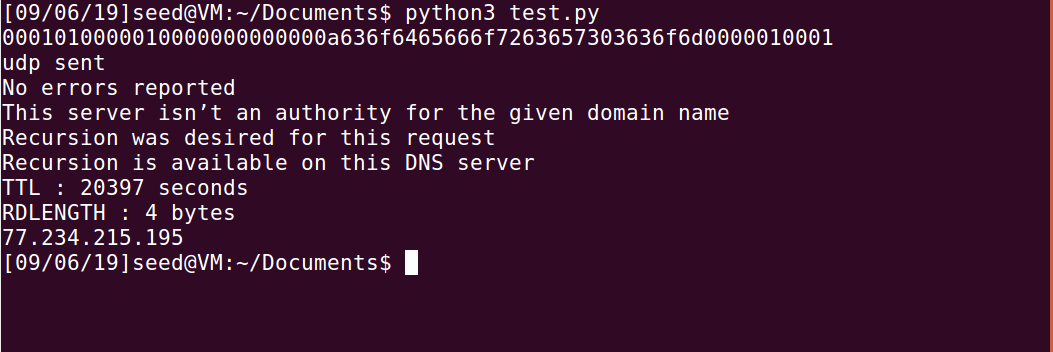


Fig. dns query in action



Fig. query observed in wireshark

1. **ip has been spoofed:**

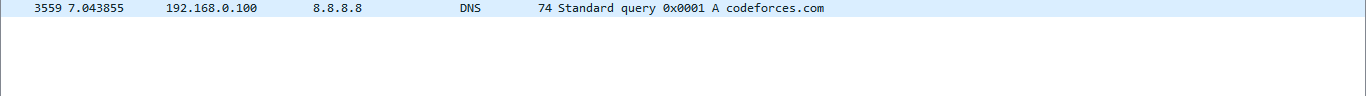


Fig. query sent but no response from dns server

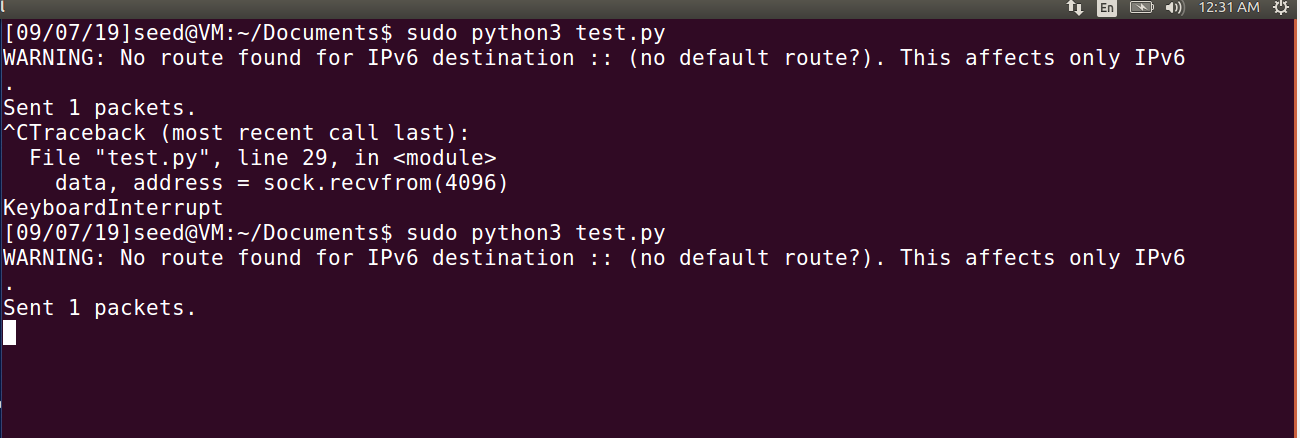
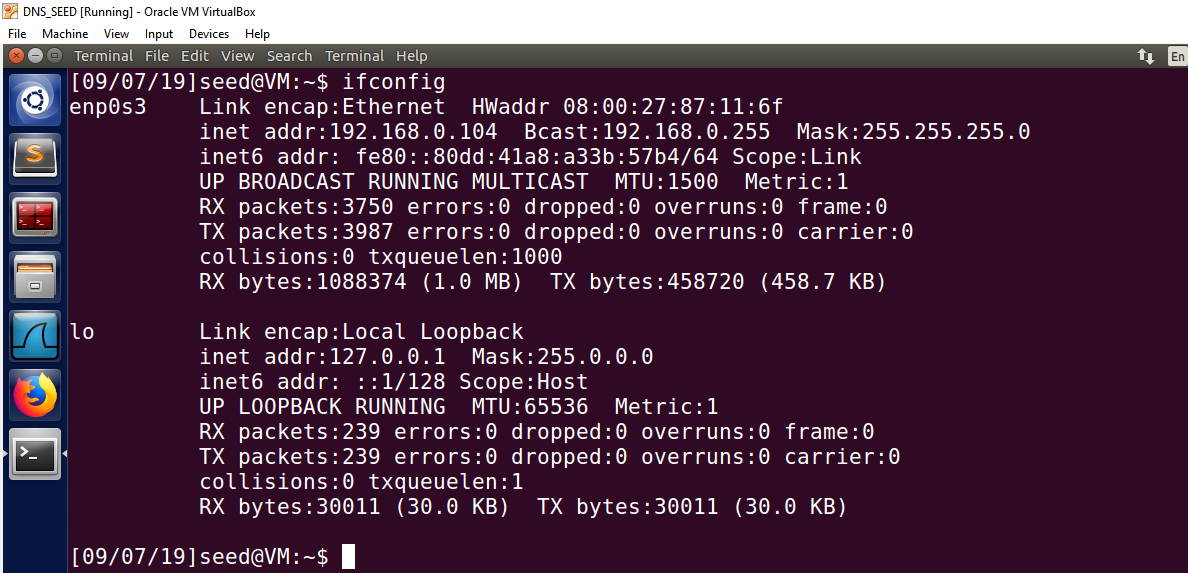


Fig. waiting for response

We can see that when we use scapy to change the src ip we send the dns query but do not get any response. In the terminal it is still waiting for a response.

1. **Attack**: Now we simulate the attack on dns server. We use bind9 in ubuntu to make the pc to work as a dns server. As the dns server we use seed ubuntu as bind9 is pre-built in it. After running we give the following command: ‘ifconfig’ to get the servers ip 

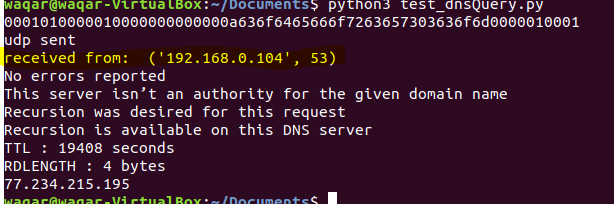


Fig. response from bind9 dns server

In another vm we use ubuntu 18.04 where we run the tester code written for DNS\_QueryBuilder.py and observe that indeed the response is coming from the dns server we have created. To do this we just need to set the destination ip and port as 192.168.0.104 and 53. The ip of the dns server may change when we change the network so we have to change the code accordingly.

Then we take another vm that will work as the user-pc. We edit the file ‘/etc/resolv.conf’. There we write -> “namespace ip\_of\_dns” - here ip of dns is 192.168.0.104 in my case. Now whenever we use our browser it will use the dns server on 192.168.0.104. If we use anything other than it say 105, we shall observe that the browser is unable to find the desired domain that proves that editing the file has ensured we use our dns.

Now finally if we use our attacker-pc and flood the dns then the browser of the user-pc won’t be able to find any domain

We run the code in three different terminals. Here we use a firefox extension “Load-Time” [<https://addons.mozilla.org/en-US/firefox/addon/load-time/>] to check if the attack is working

**Does the attack works?:**

The attack works but without spoofing the src ip.If we check the screenshots below, they taken before running the attack and after running the attack. Though not always it becomes impossible to load the website, the attack definitely increases the traffic and thus the load-time of the websites increases.

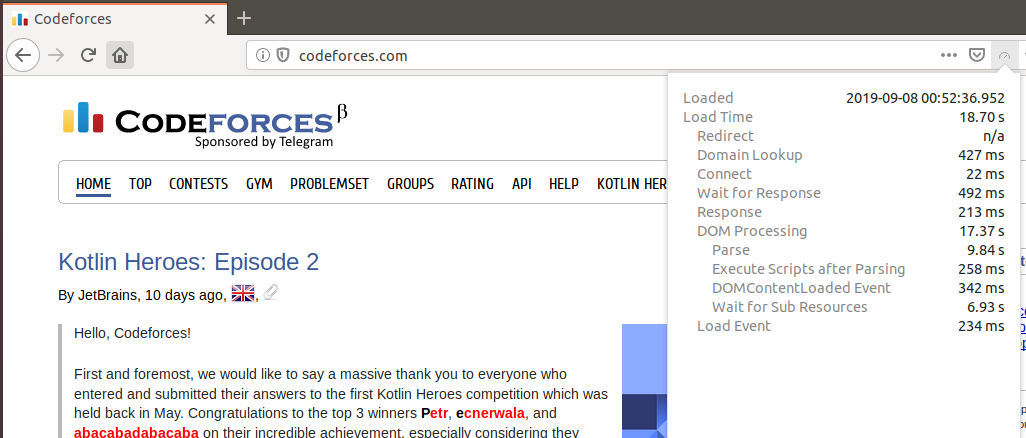


Fig. codeforces before the attack

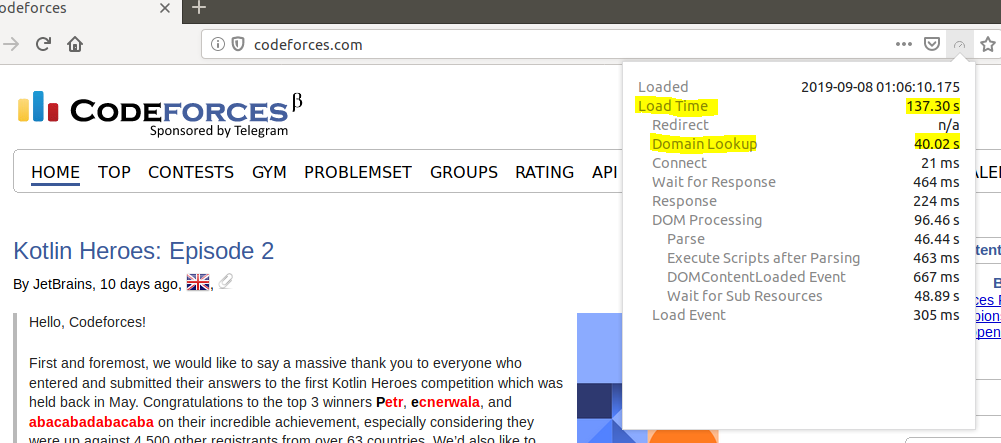


Fig. codeforces after the attack



Fig. wikipedia before the attack

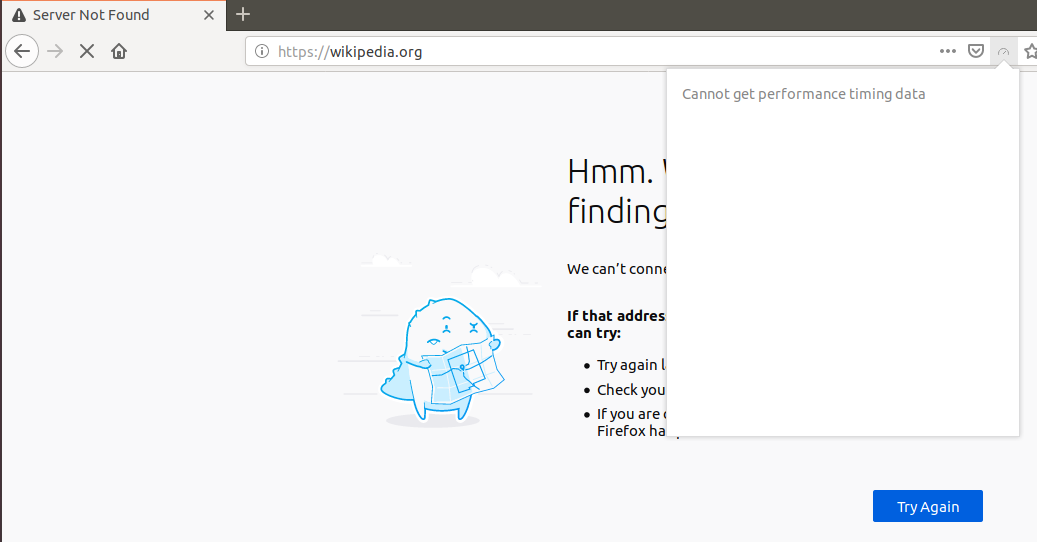


Fig wikipedia after the attack, it didn’t load

**Why spoofing is Not working:**

The plan was to flood the dns server with the query written from scratch according to RFC-1035 using an infinite while loop. To spoof the src ip scapy packet crafting library was supposed to be used, as sending the query through a udp won’t let us hide our ip. But the scapy library sends packet really slowly, e.g: within 30s scapy sends approx. 3500 packets where normal udp sends more than 100000 packets. Three terminals were used to run the code and within 30s they sent 121741, 124982 and 183077. So instead spoofing normal packets were sent so that the dns server can be flooded.