**Local DNS Attack SEED Lab**

Contents

[Environment Setup 3](#_Toc123136336)

[Task 1 13](#_Toc123136337)

[Task 2 18](#_Toc123136338)

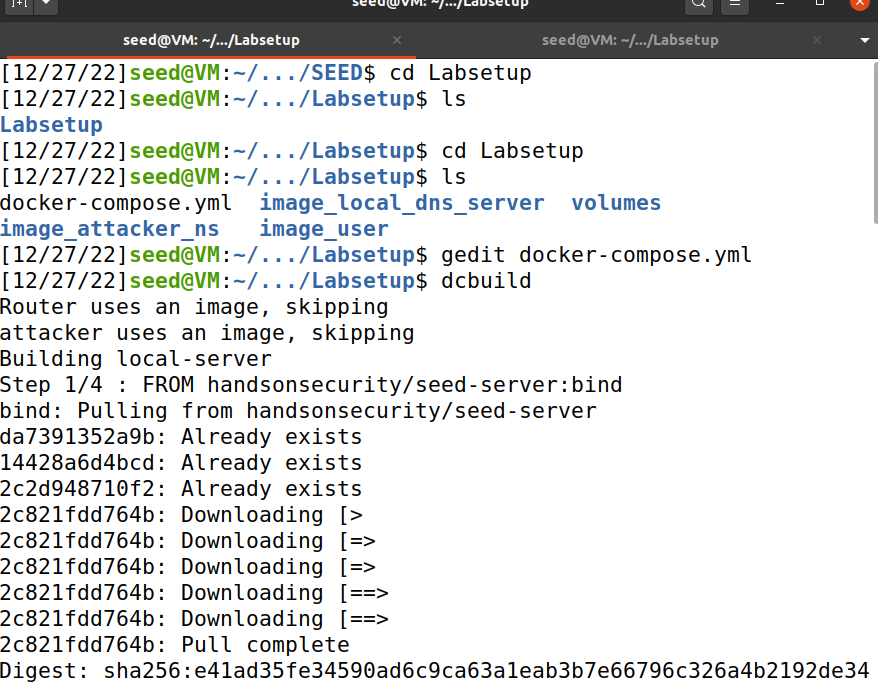
[Task 3 21](#_Toc123136339)

[Task 4 25](#_Toc123136340)

[Task 5 28](#_Toc123136341)

# Environment Setup

Building Dockers in order to proceed in the lab in proper environment.



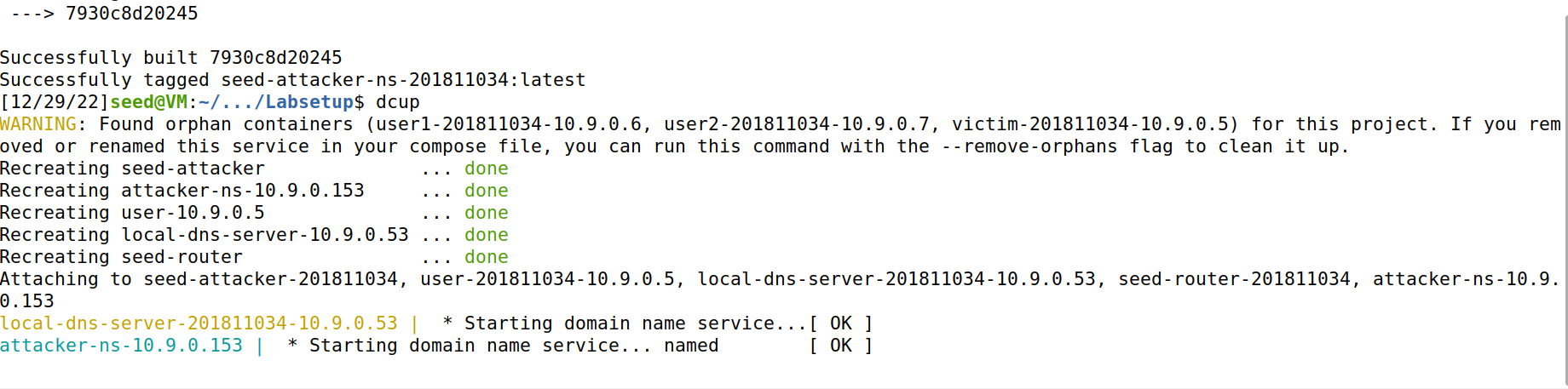
Adding my student ID in container names. In screenshot below ID is added in **container\_name** in Router and Local-Server Dockers.



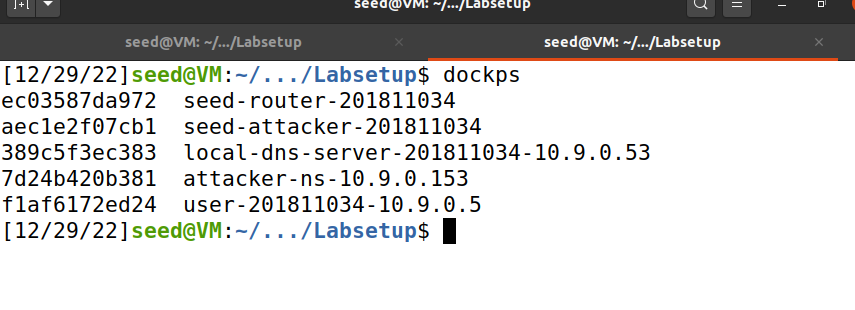
In the below screenshot I added ID in **container\_name** in User, Attacker and Attacker Name Server Dockers.



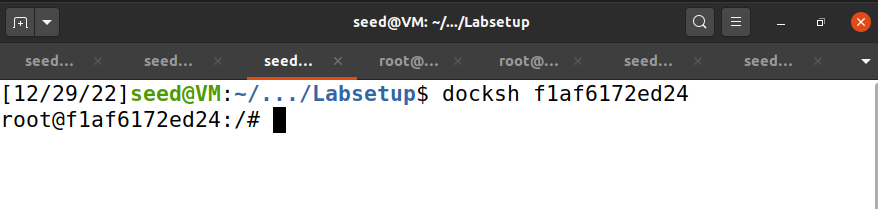
Setting up Dockers.



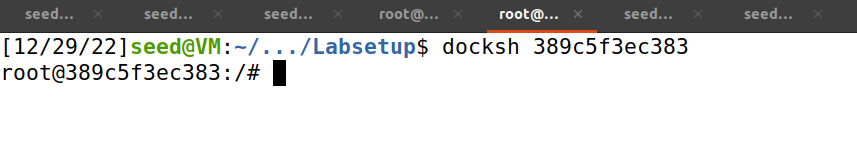
Here, all the Dockers available are displayed.



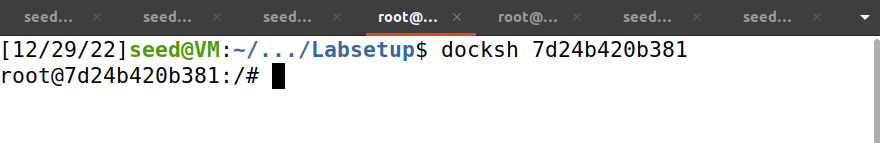
Setting User’s Docker in the terminal.



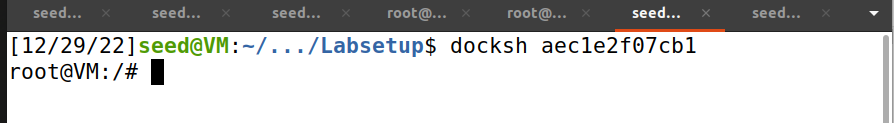
Setting Local DNS Server’s Docker in the terminal.



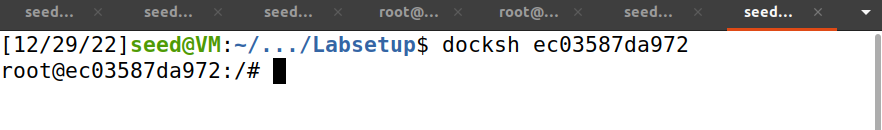
Setting Attacker’s Nameserver Docker in the terminal.



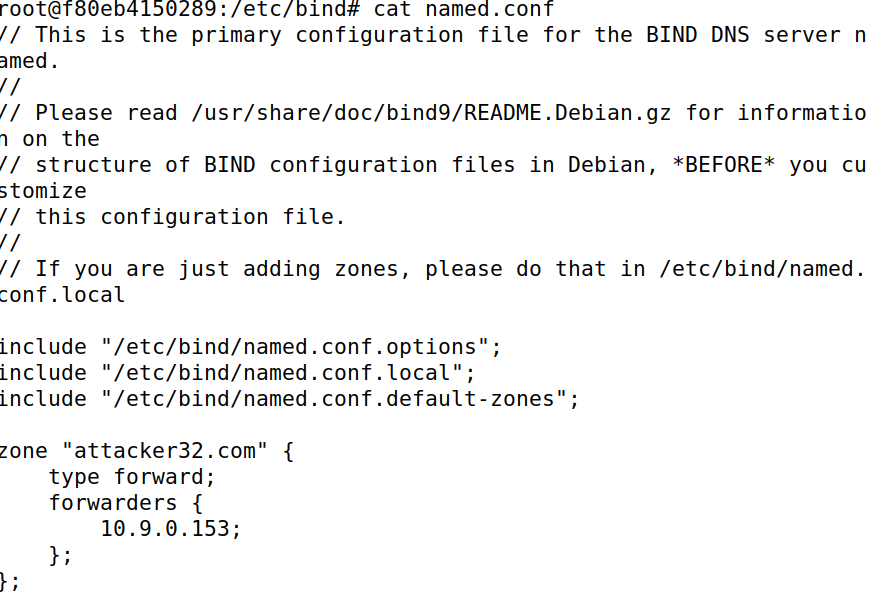
Setting Attacker’s Docker in the terminal.



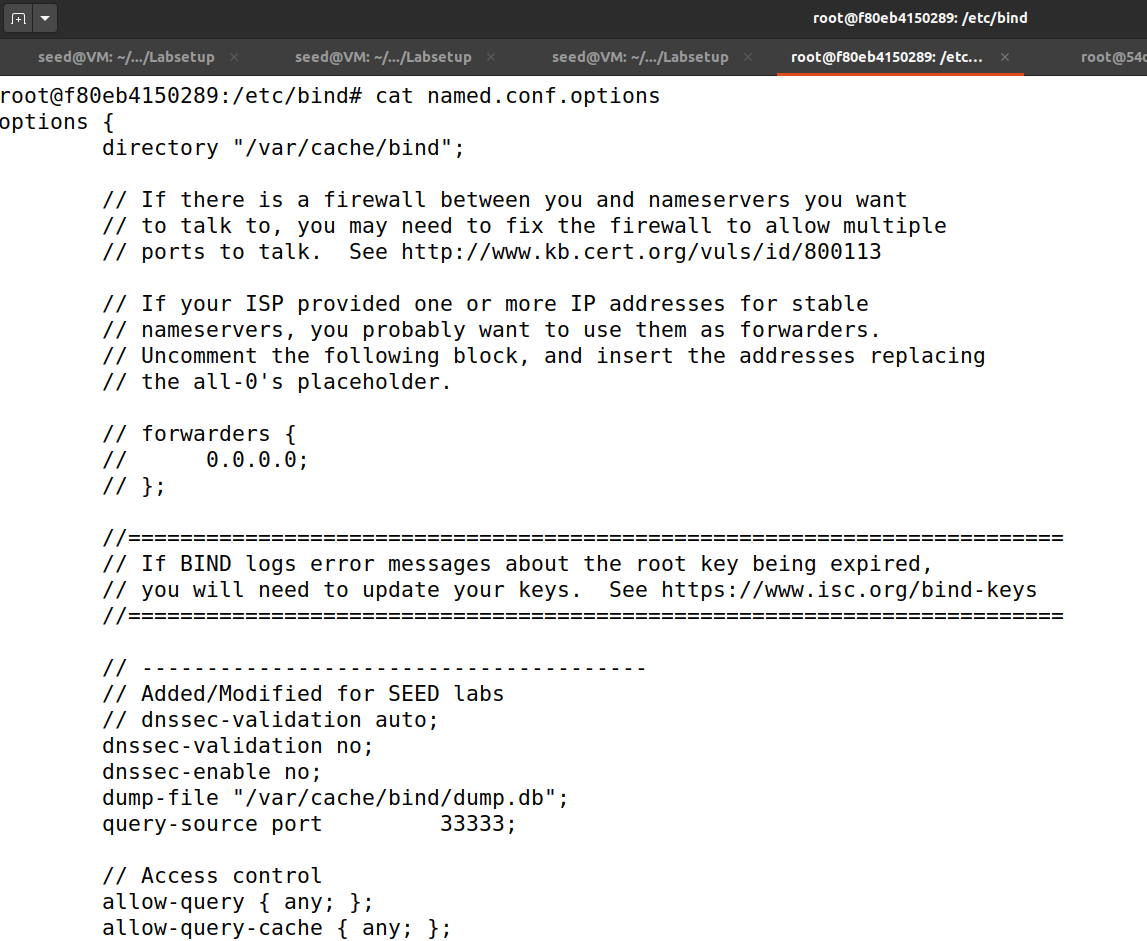
Setting Router’s Docker in the terminal.



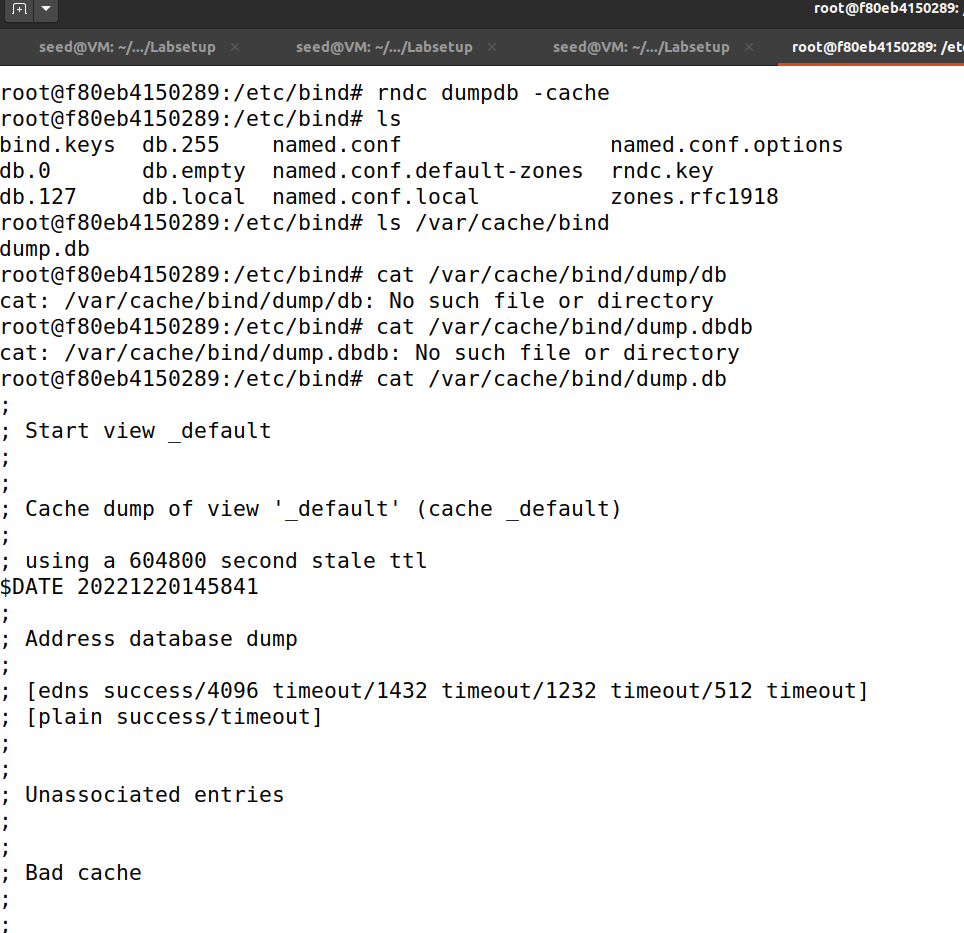
Confirming Attacker’s zones.

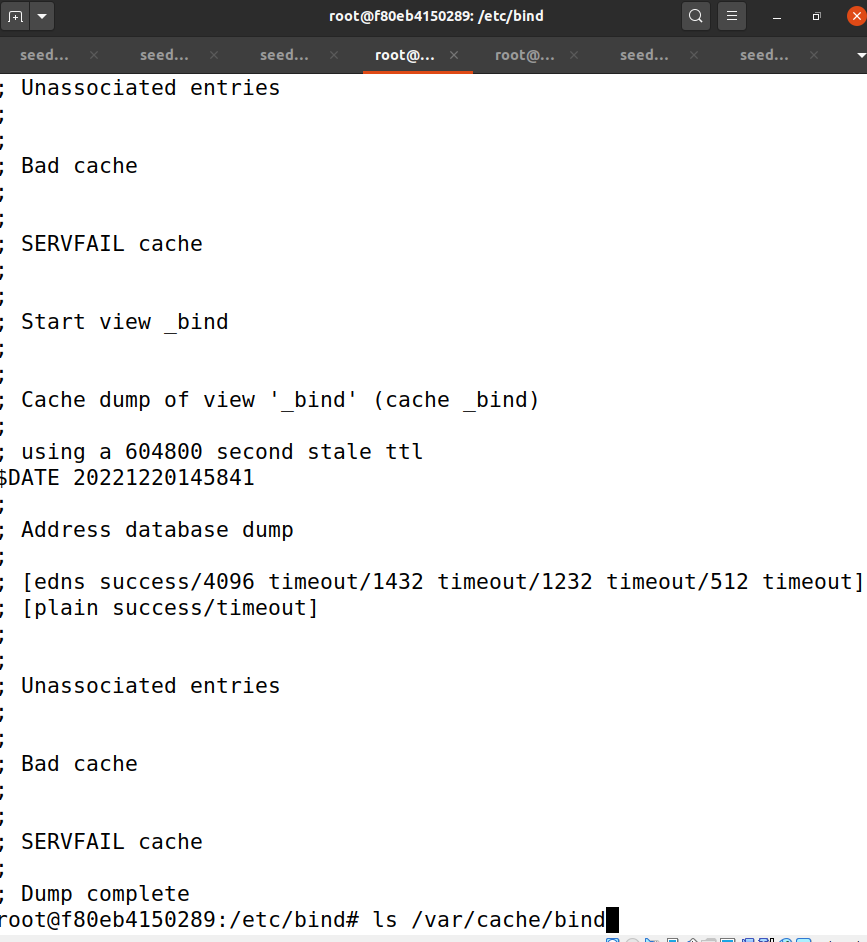


Confirmed that DNSSec is turned off and the **dump.db** location is taken.

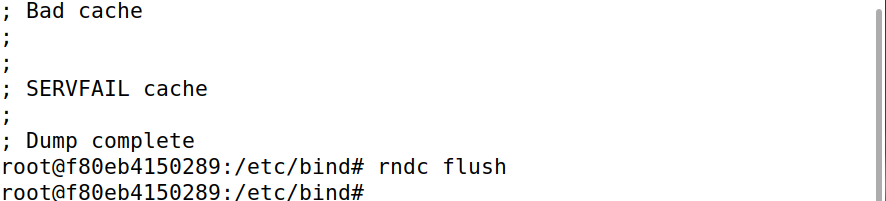


Putting command to check the dump cache.





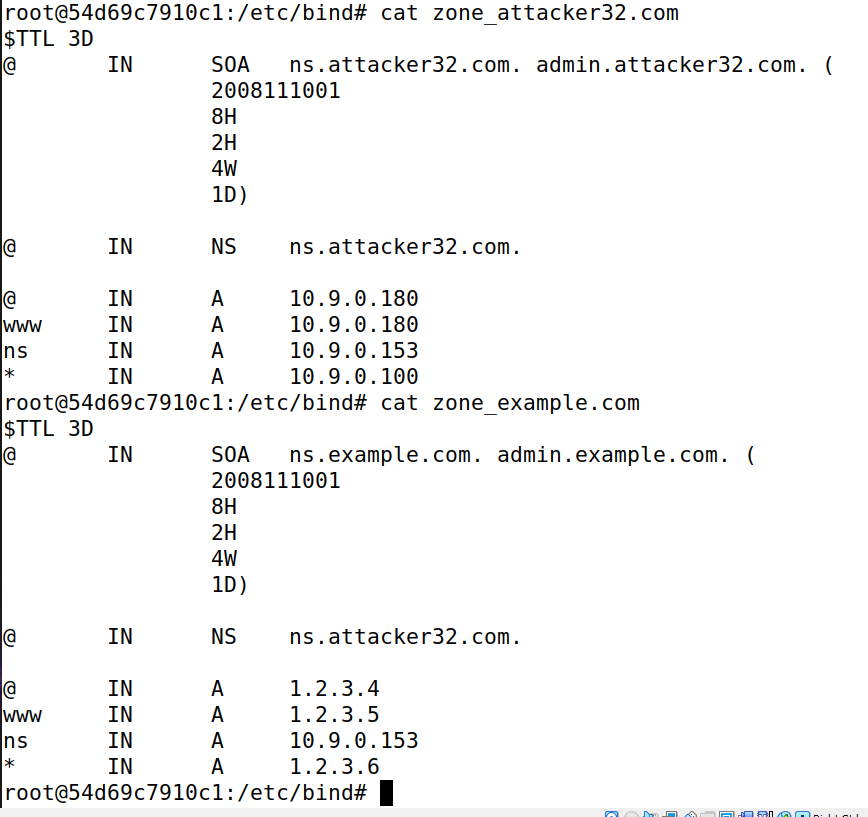
Now clearing the cache.



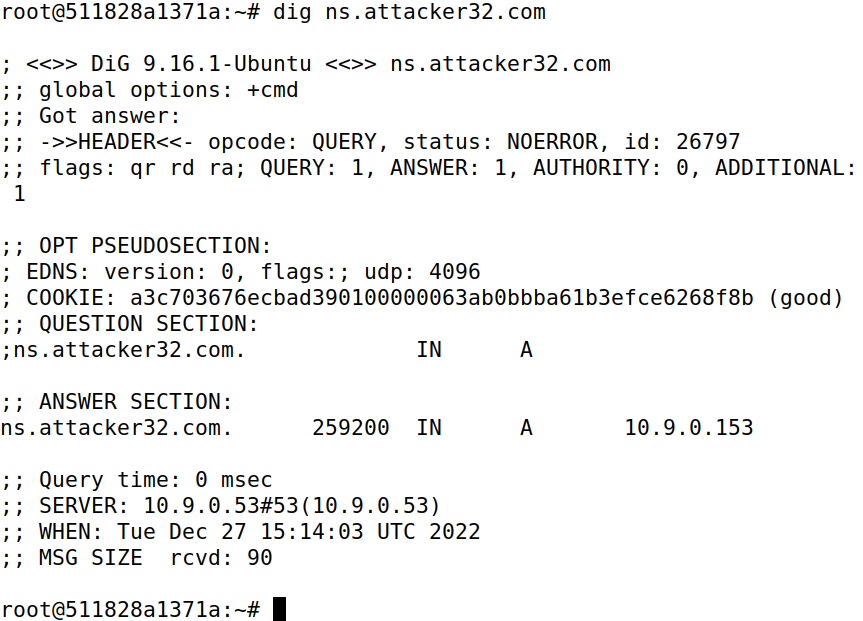
Checking zones again from Attacker-ns Docker.



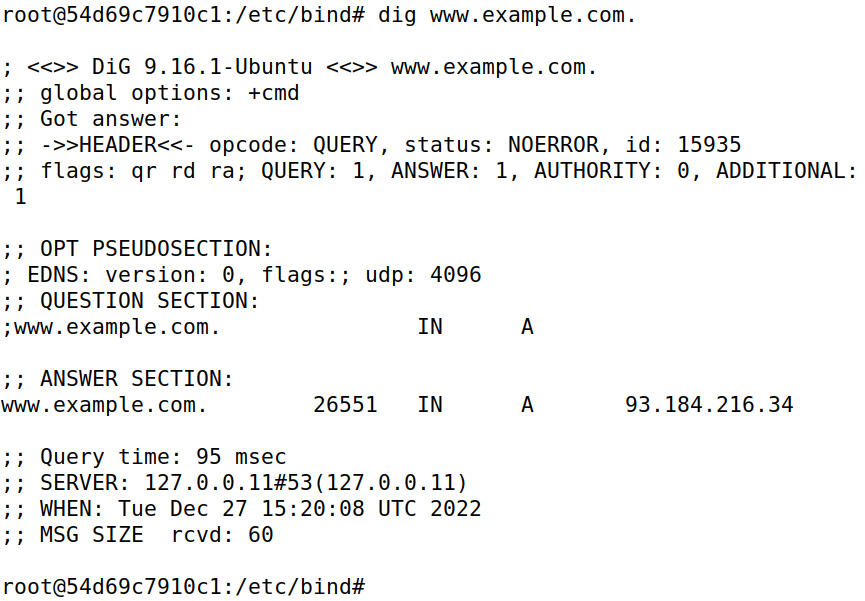
Checking the real and fake zones of the Attacker, respectively.



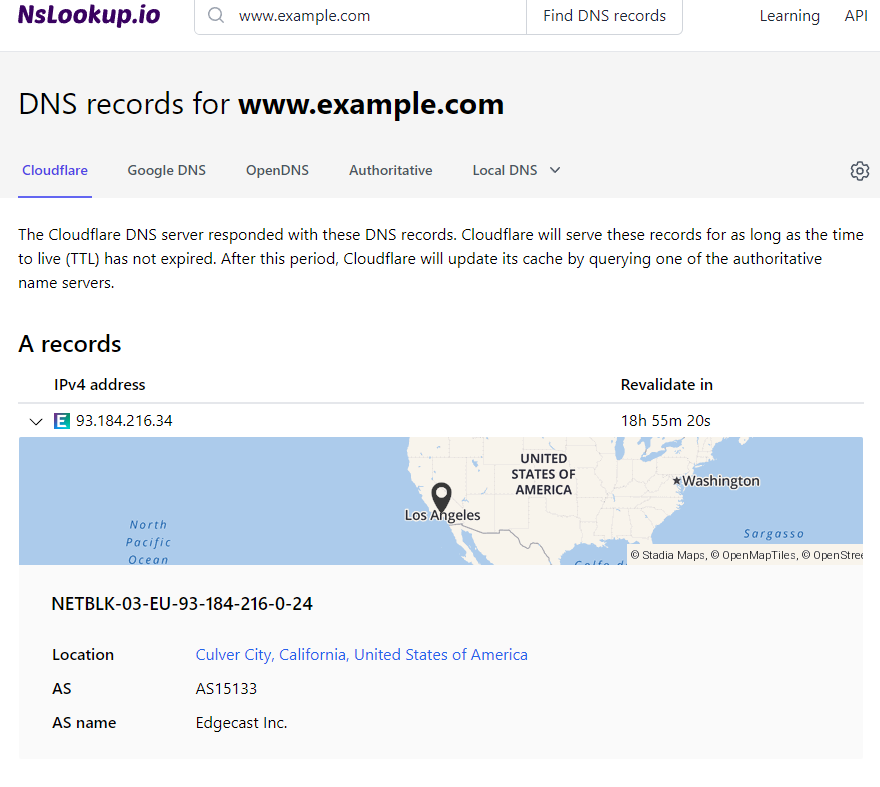
While testing the DNS Setup I tried grabbing the IP Address of the attacker32 in User’s terminal and got the response here in the **ANSWER** Section.



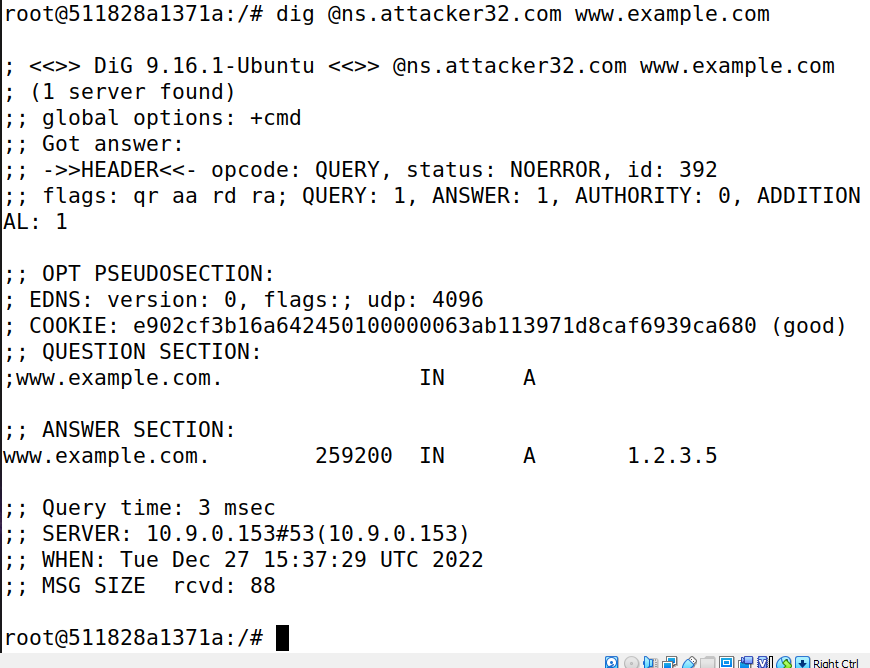
Checking the IP Address of the fake zone of the attacker which is visible in the **ANSWER** Section below.



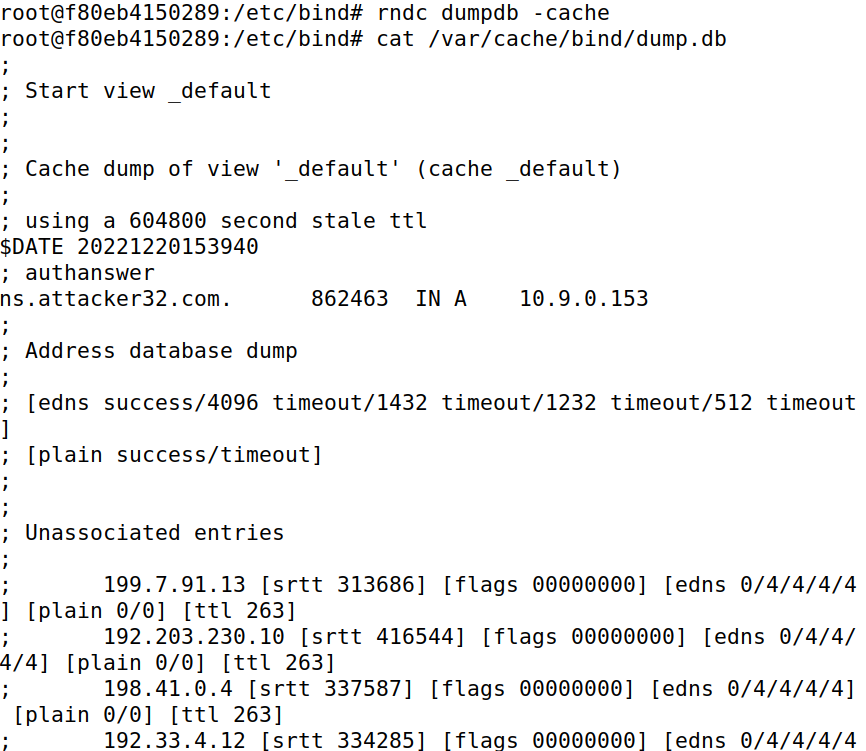
Cross verifying on <https://www.nslookup.io/>. And I noticed that the IP of the fake zone matches.



Upon performing the last test it is clear that when called the fake zone address will be displayed with this command.

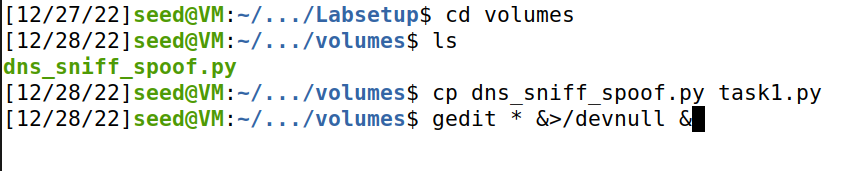


Checking the cache again in Local DNS Server’s terminal where ns.attacker32.com and other dumps are found

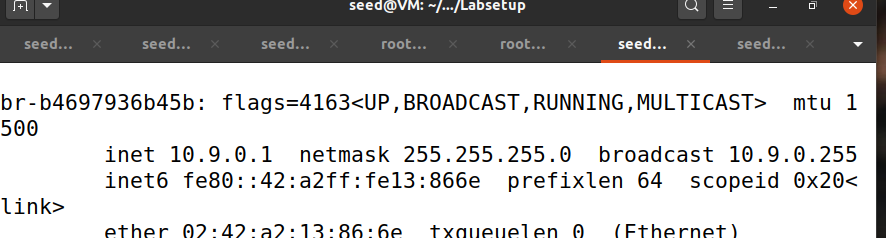


# Task 1

Copying the provided script in another file to begin the task and opening it in the editor.



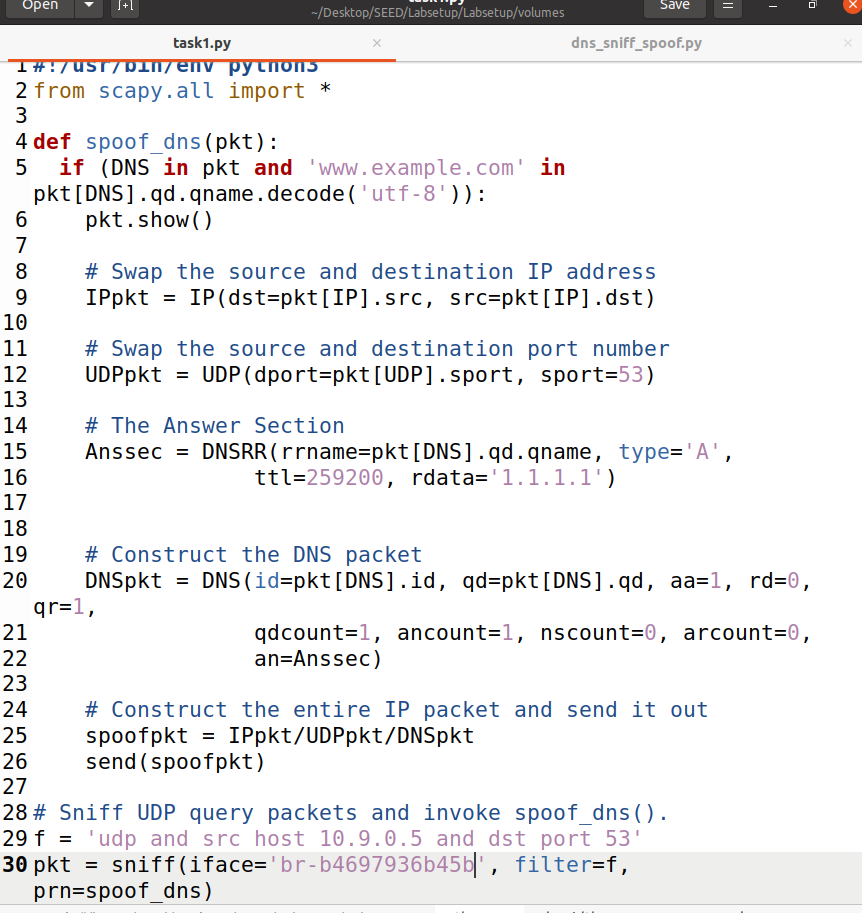
Checking Attacker’s Interface in Attacker’s terminal.



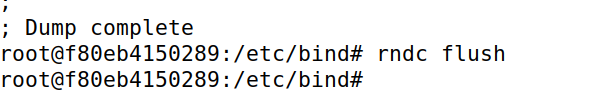
Provided Code looks like this which I will modify for the tasks ahead.



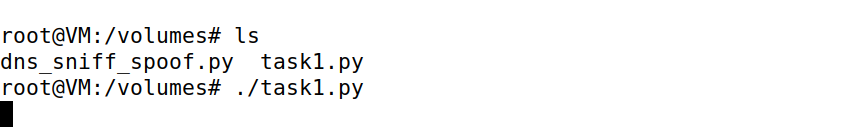
Modifying the code, where on line 6 I added the code to show packets. Spoofed the IP address and DNS packet while removing Authorization Server code lines as they are not required in this task. Made modifications on line 29 by adding source host and finally on line 30 the interface of Attacker checked from the Attacker’s terminal as shown in the screenshot above.



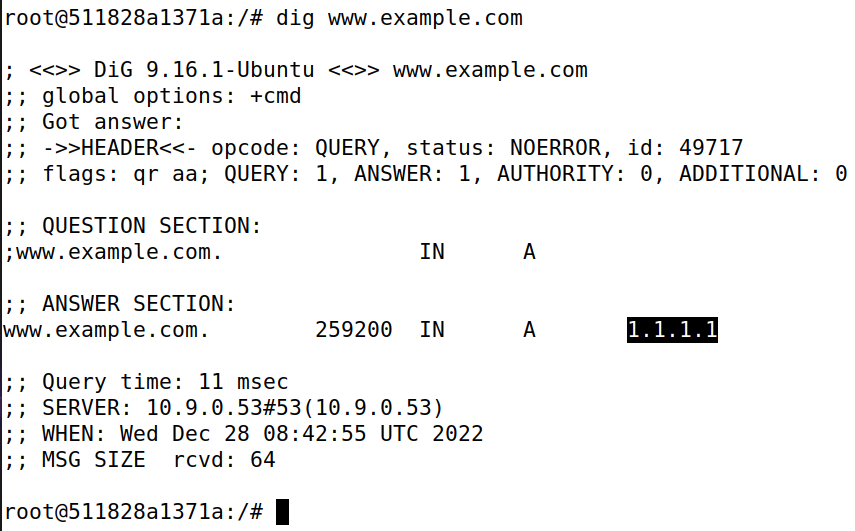
Now flushing the cache in Local DNS Server Docker.



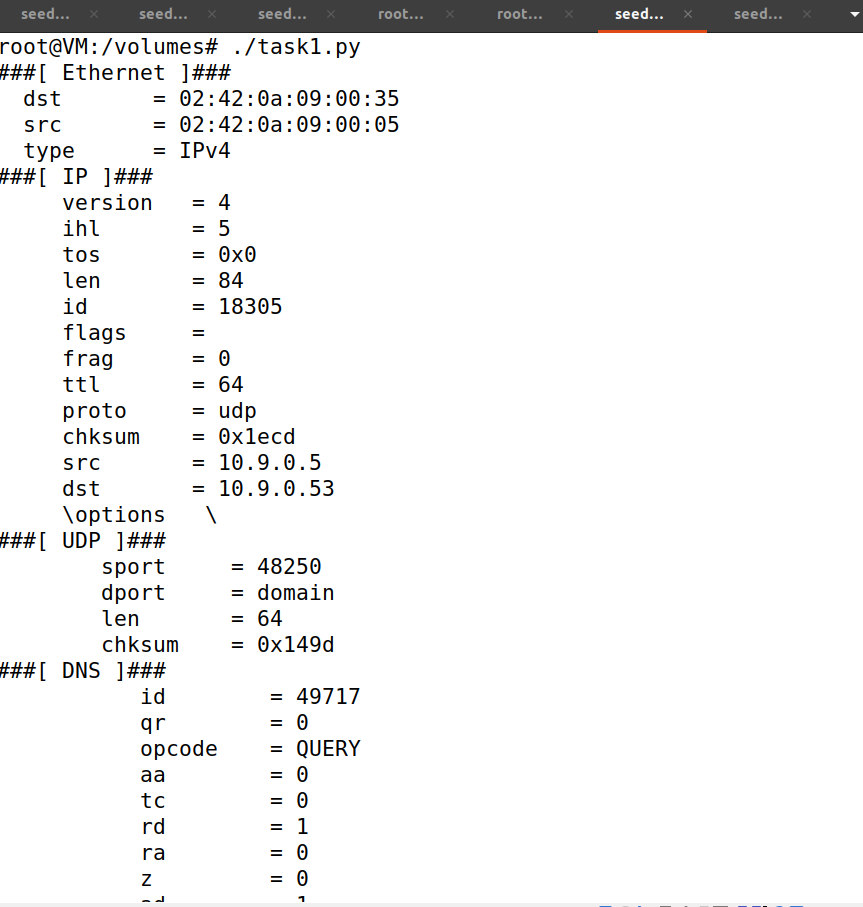
Launching the Attack.

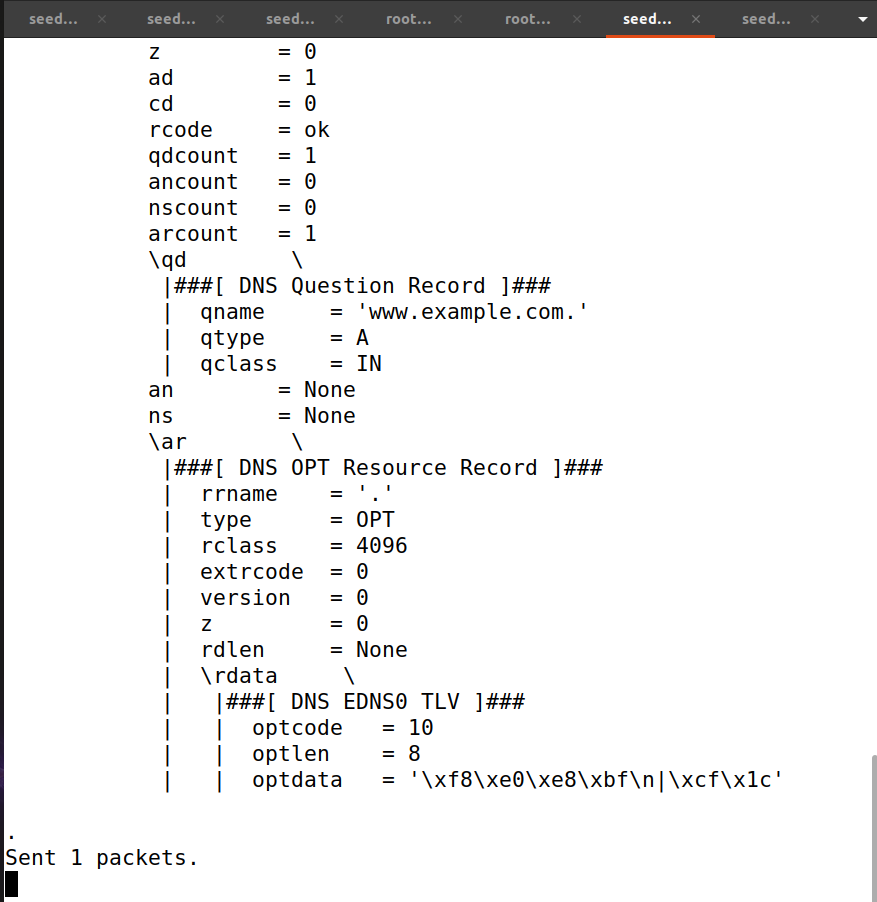


Now checking if the fake address was caught or not in User’s terminal, which is caught and that’s a success in spoofing but until placed with a packet delay the attack cannot be claimed to have succeeded.



This Packet has been sniffed from the User’s Machine in Attacker’s terminal.

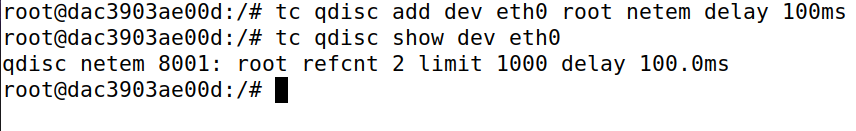




These are the entries found on Router’s terminal.

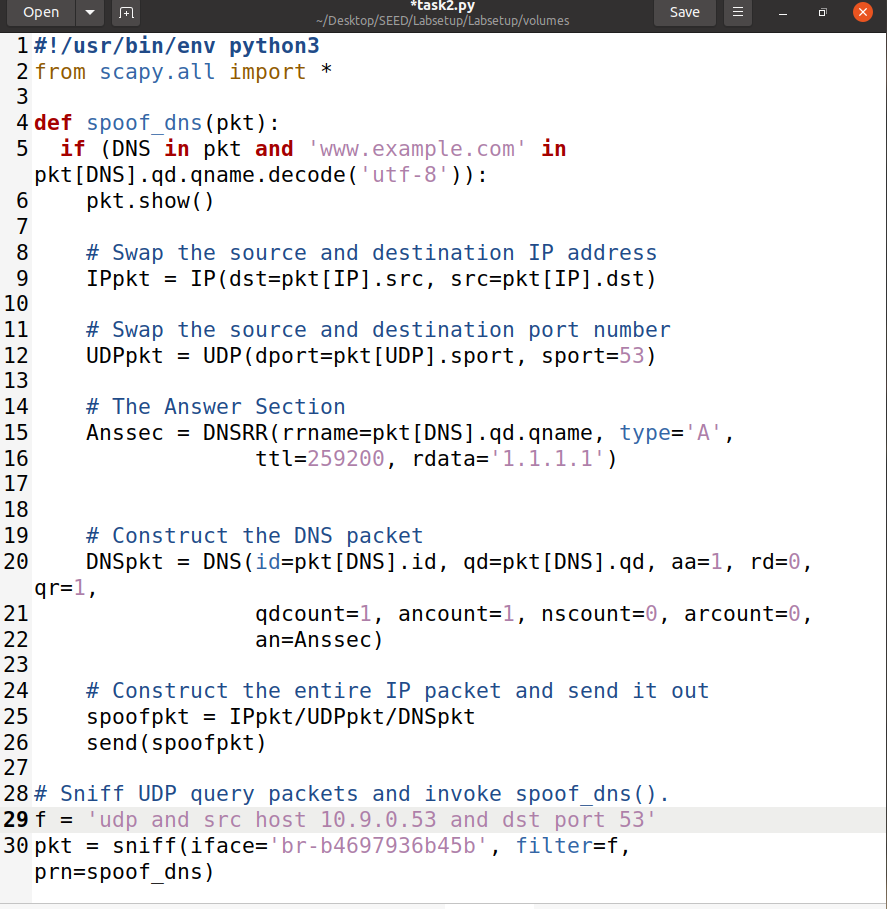


Adding an entry to delay traffic by 100ms. This delay is only needed if the attack doesn’t succeed which in my case did succeed.

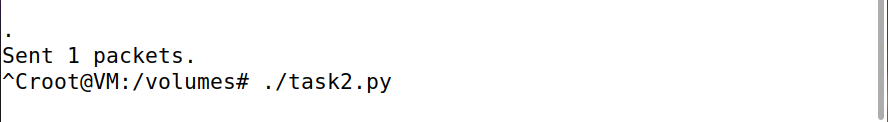


# Task 2

Now to intercept the query from Local DNS Server I have modified the code with the Local DNS Server’s IP Address on line 29.



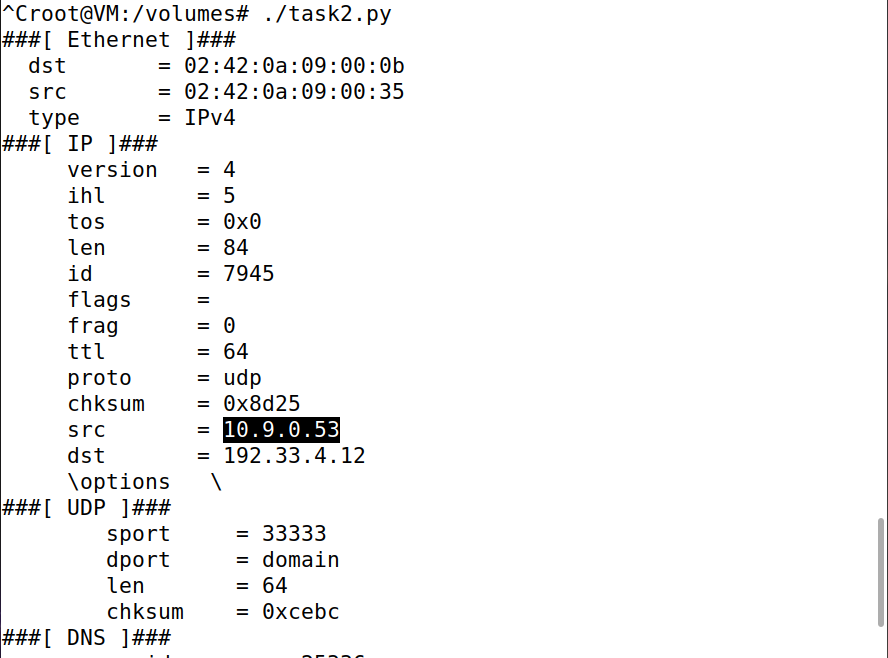
Now launching the attack from Attacker’s machine.



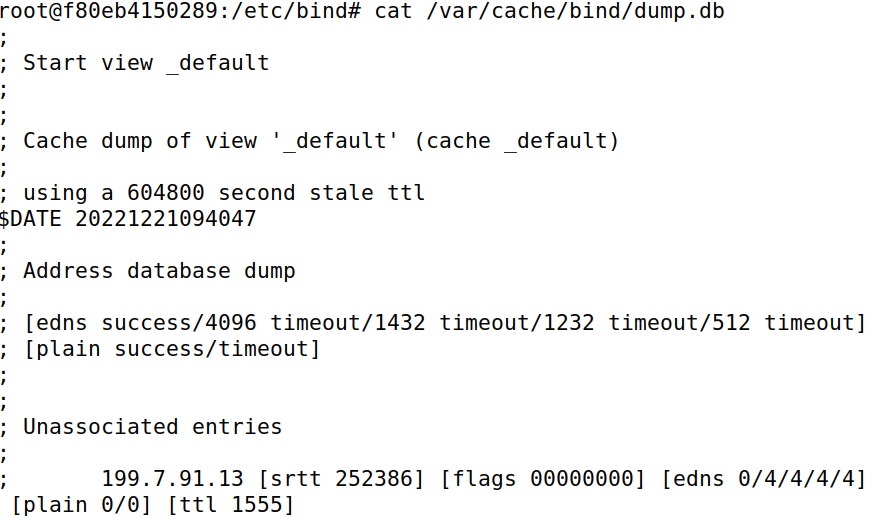
Now checking from the User’s machine.



And checking the sniffed packet from the Attacker’s machine which clearly shows that the packet was sniffed from the Local DNS Server.



Now while checking Local DNS Server nothing appeared like clear data which was needed.



But with this command “cat /var/cache/bind/dump.db | grep example”, I have obtained the information of the server cache being poisoned.

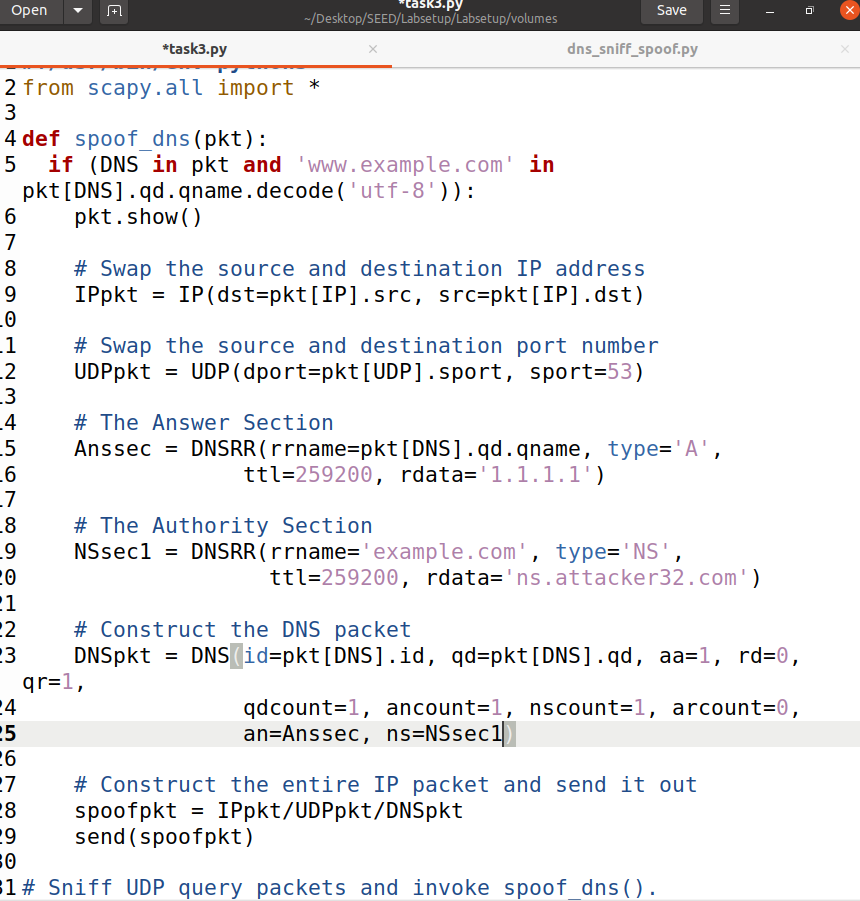


# Task 3

In order to make the attack work in this task I have copied this part from the provided code from the SEED Labs.



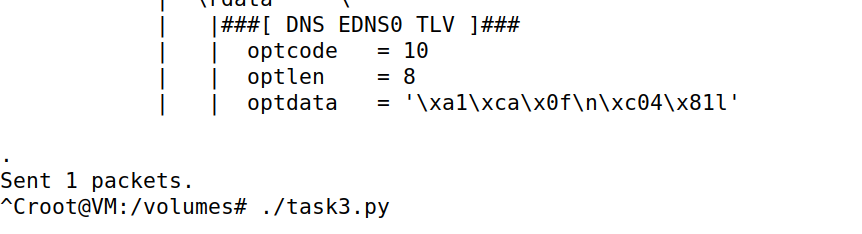
And I pasted the code here visible from line 18 to 20 and made some modifications including the name server count to 1 and ns=NSsec1 code addition on line 24.



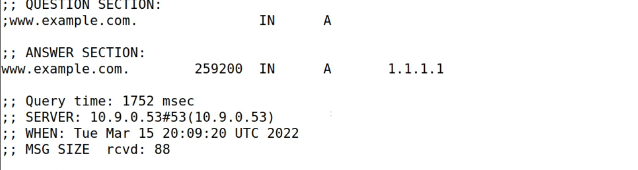
Now flushing the Local DNS Server cache.



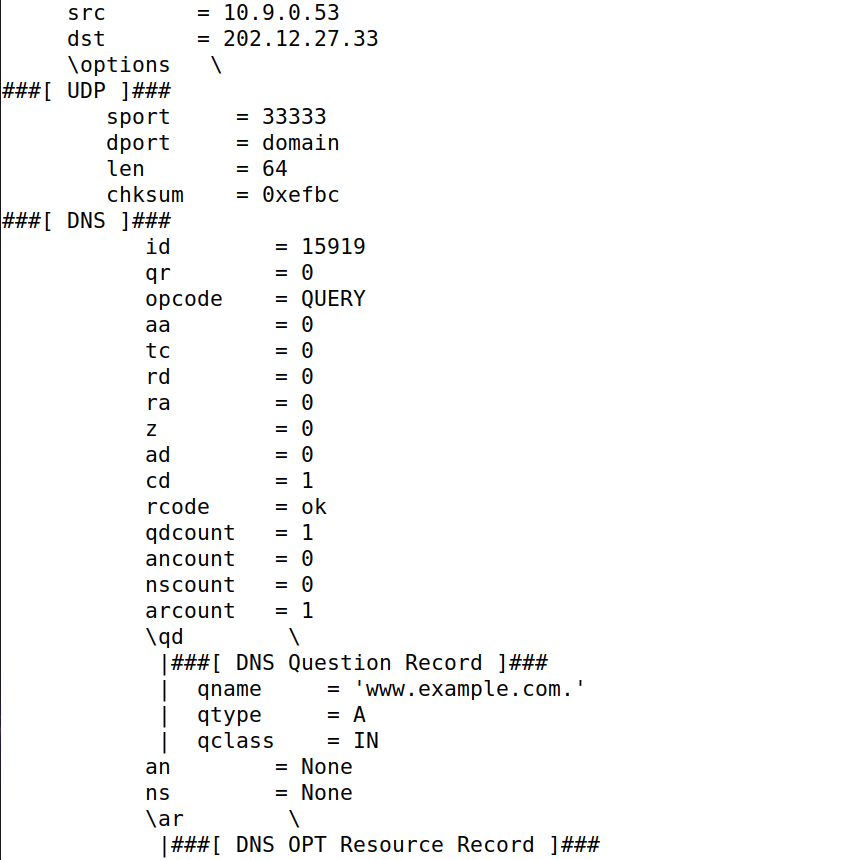
And launching the Attack.



And I again caught the spoofed the server in the Local DNS Server.



Same goes for the sniffed packet on Attacker’s machine shows the User’s activity on the target site and the packet intercepted from Local DNS Server going to Global DNS Servers.



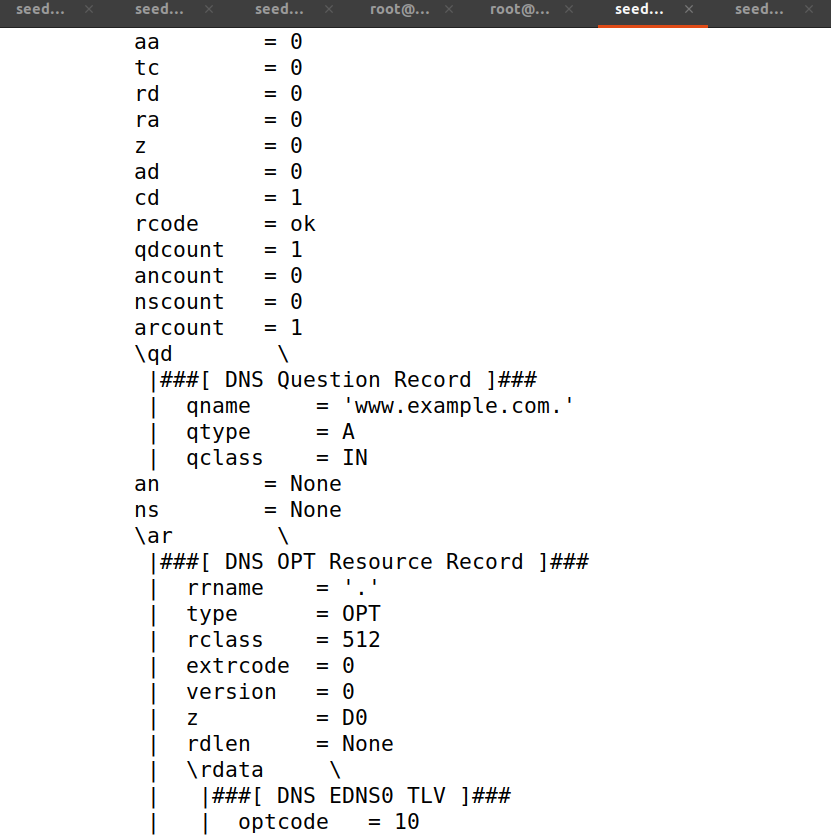
Moreover the Local DNS Server cached the fake address and the name server.



Just as an observation when I dig the malicious site from User’s terminal the IP address in not spoofed unlike the fake one.

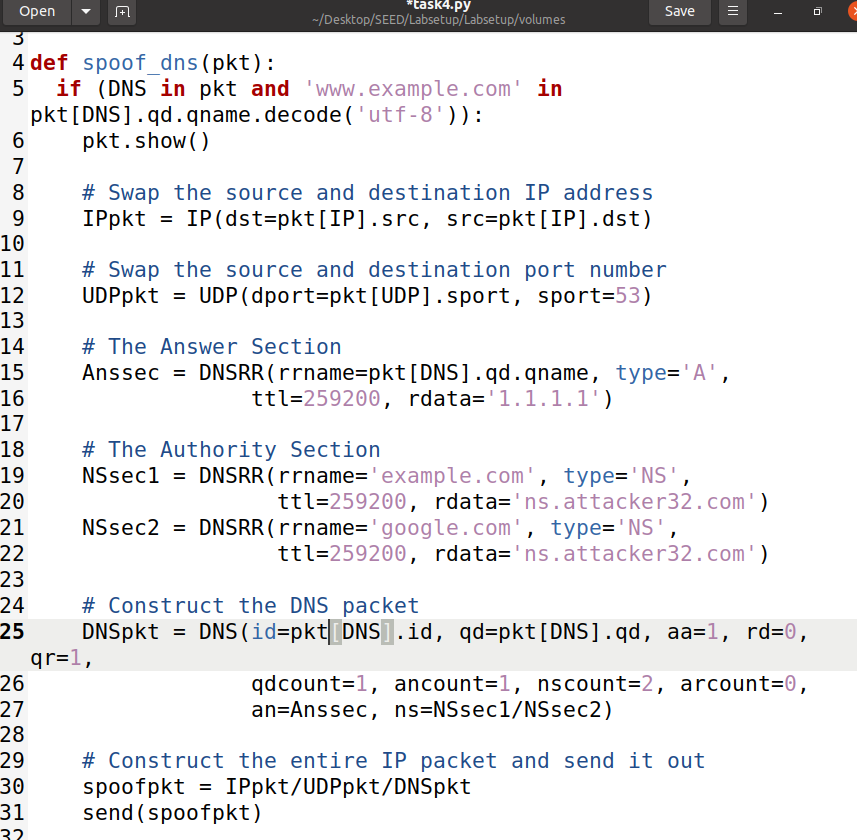


Moreover not even sniffed in the Attacker’s terminal which should be due to the configuration of the attack script and because it forwarded to the domain server.



# Task 4

To target the **google.com** I have added the code from line 21 to 22 and increased the nscount to 2 on line 26



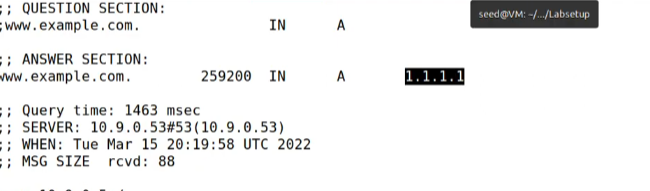
Now flushing the Local DNS Server cache before I launch the attack.



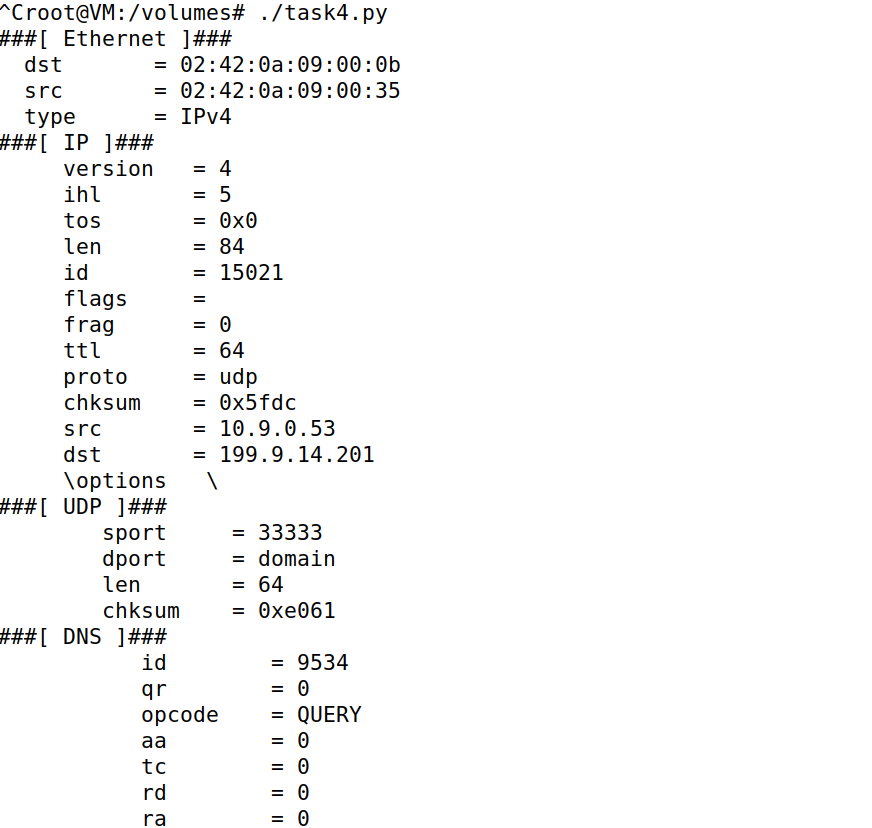
Now launching the attack script from Attacker’s machine.



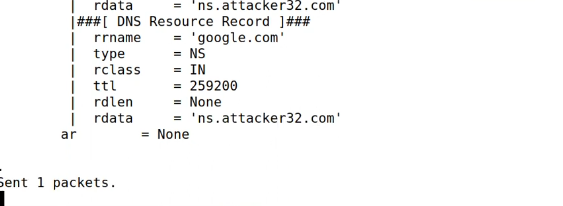
Again when digging [www.example.com](http://www.example.com) from the User’s terminal the IP address is spoofed.



This is the packet caught on Attacker’s terminal.



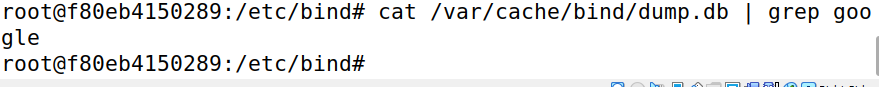
In the response packet caught this is visible.



And this being the response in the cache after it is dumped in the Local DNS Server it is visible that the attack didn’t work as it was not cached.



Not only that but no response if I try grep google.

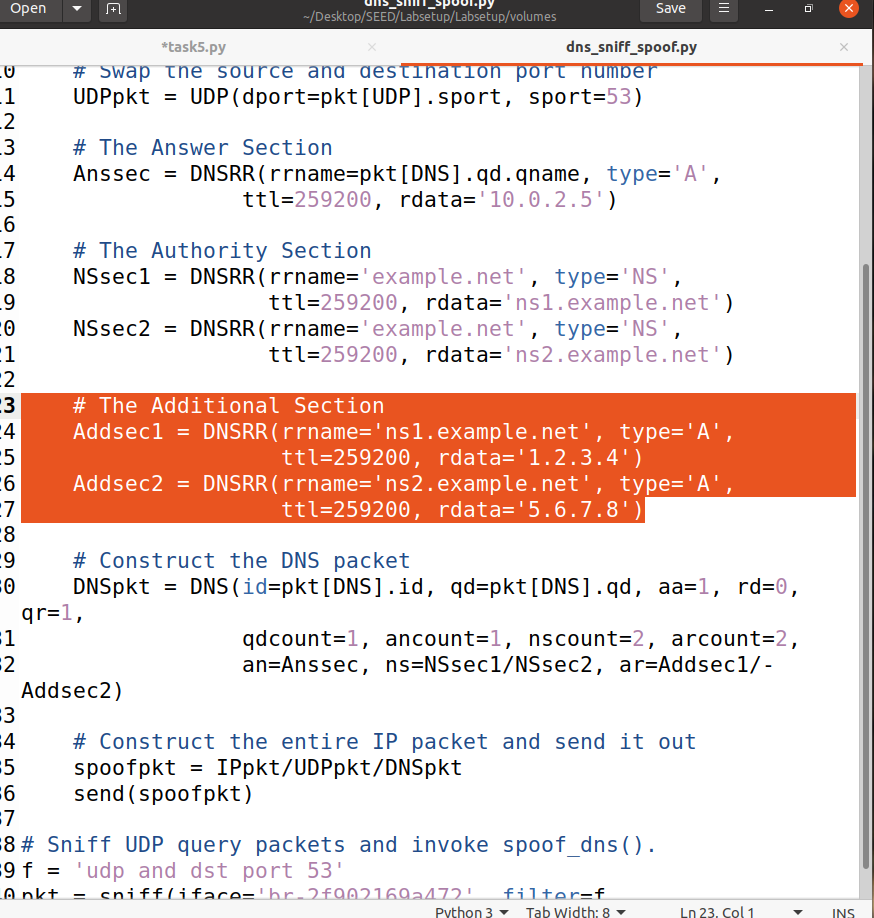


# Task 5

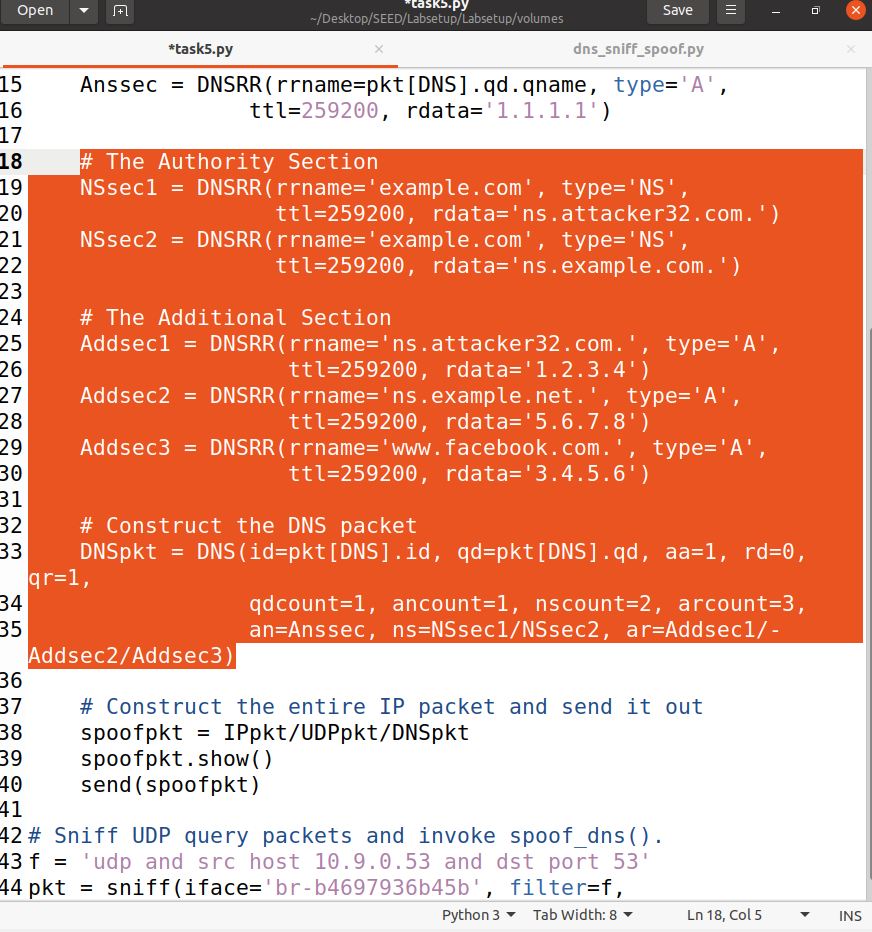
Flushing the cache after stopping the attack script.



Copying the Additional Section provided in the given script.



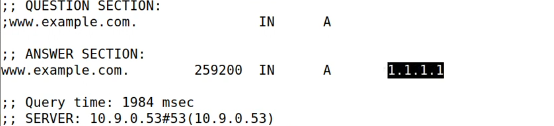
Then pasted and modified this highlighted area as per the need of the task given in the Lab Manual.



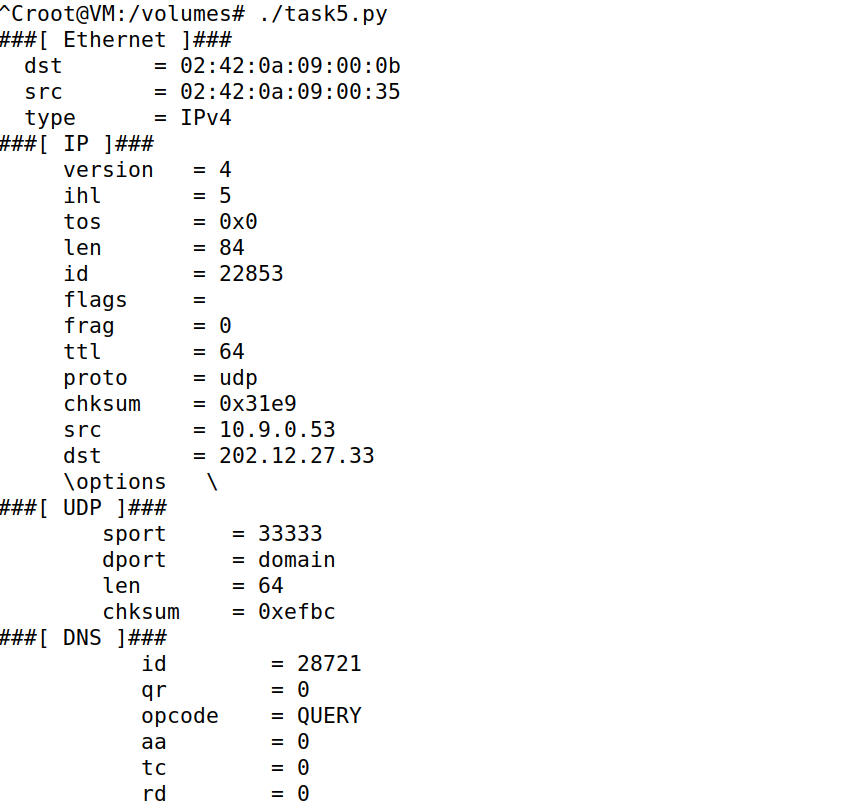
Launching the Attack.

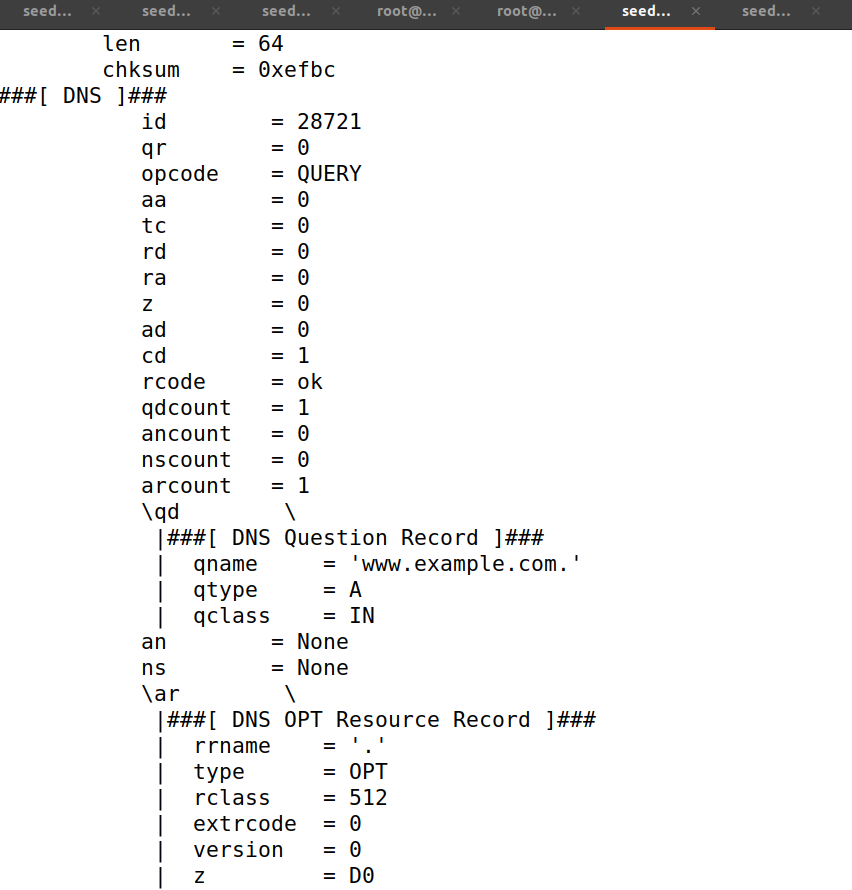


In User’s terminal by digging the spoofed IP Address is received.

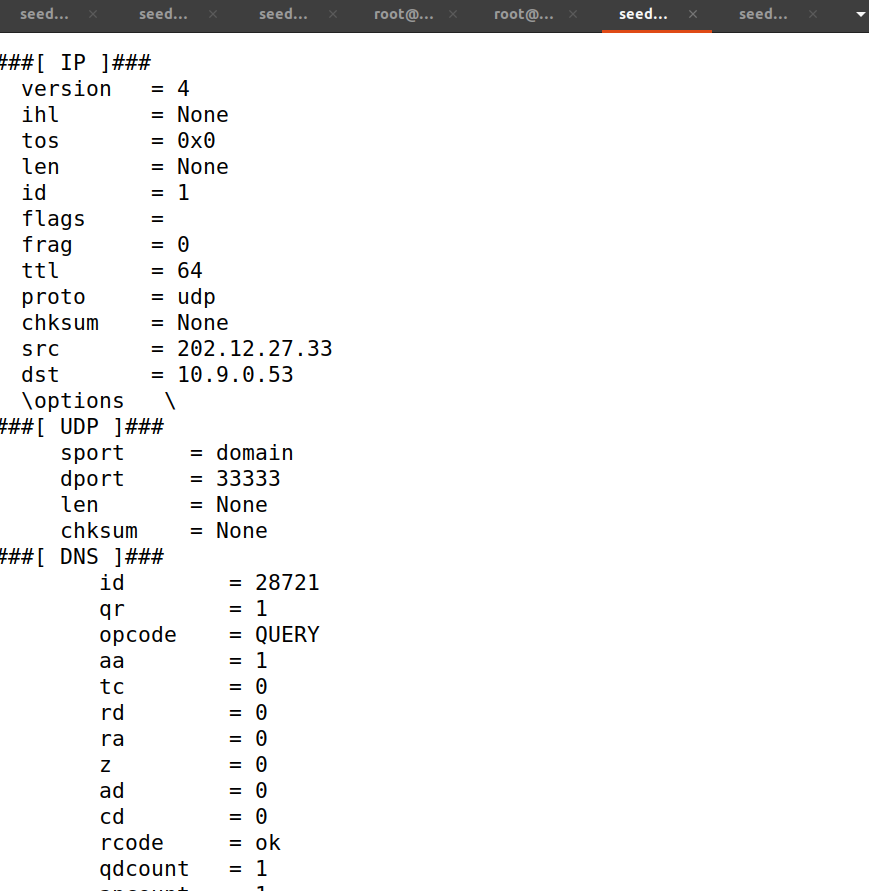


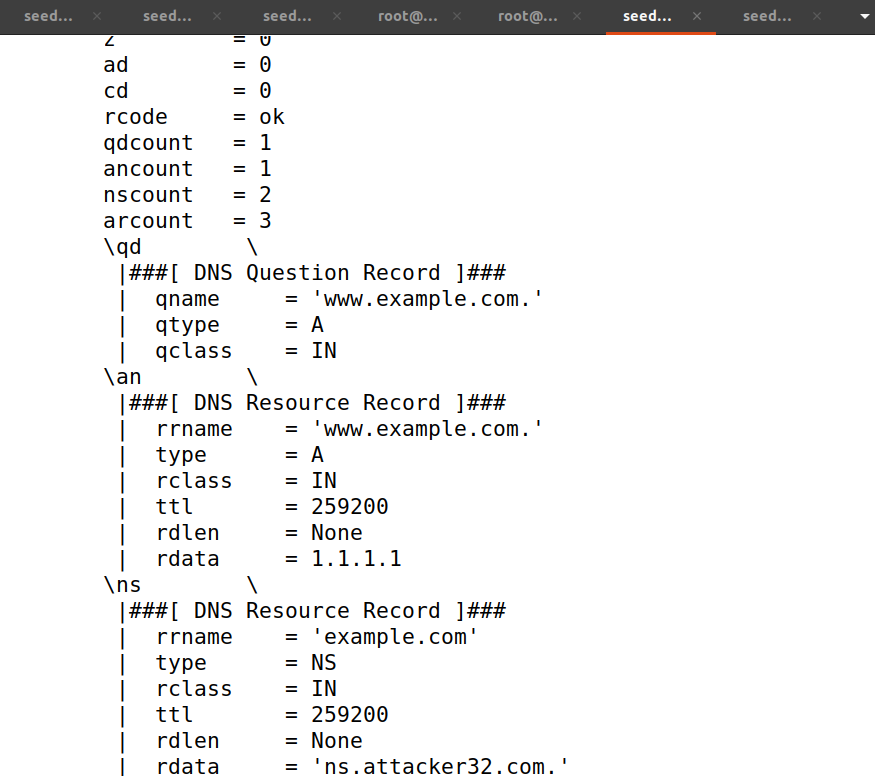
Now in the sniffed packets by the Attacker including the response packet. In the packet going from the Local DNS Server following information is present.

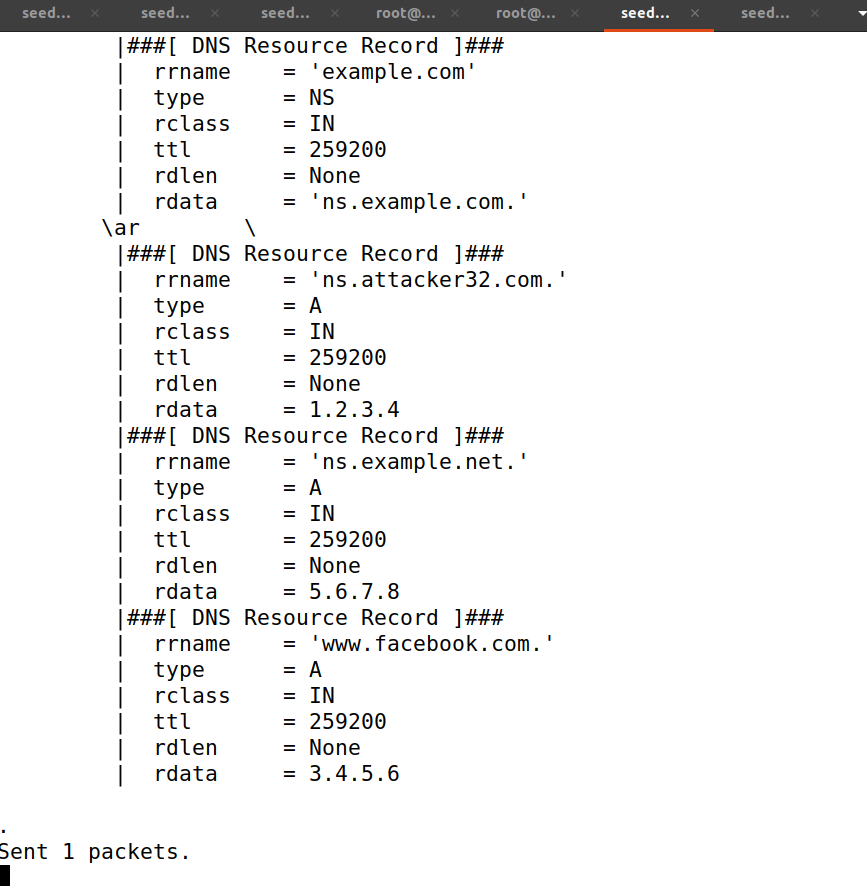




And in the packet coming back to the Local DNS Server shows this information.







Now while digging from the User’s terminal only this information was available from the Name Server and Authorization Server.

Using the command cat /var/cache/bind/dump.db | grep attack.



Using the command cat /var/cache/bind/dump.db | grep example.



And finally the Additional Servers which were not cached.

