UE18CS335 – COMPUTER NETWORK SECURITY

Assignment – 4 REMOTE DNS CACHE POISONING ATTACK

Date: 04/04/2021

By:

Nitish S

PES2201800368

6 'A'

Objective: - To understand the remote DNS cache poisoning attack, also called the Kaminsky DNS attack.

Lab Setup: Attacker VM: 10.0.2.5 ; **Victim VM:** 10.0.2.4 ; **DNS Server VM:** 10.0.2.14

TASK 1: CONFIGURE THE LOCAL DNS SERVER

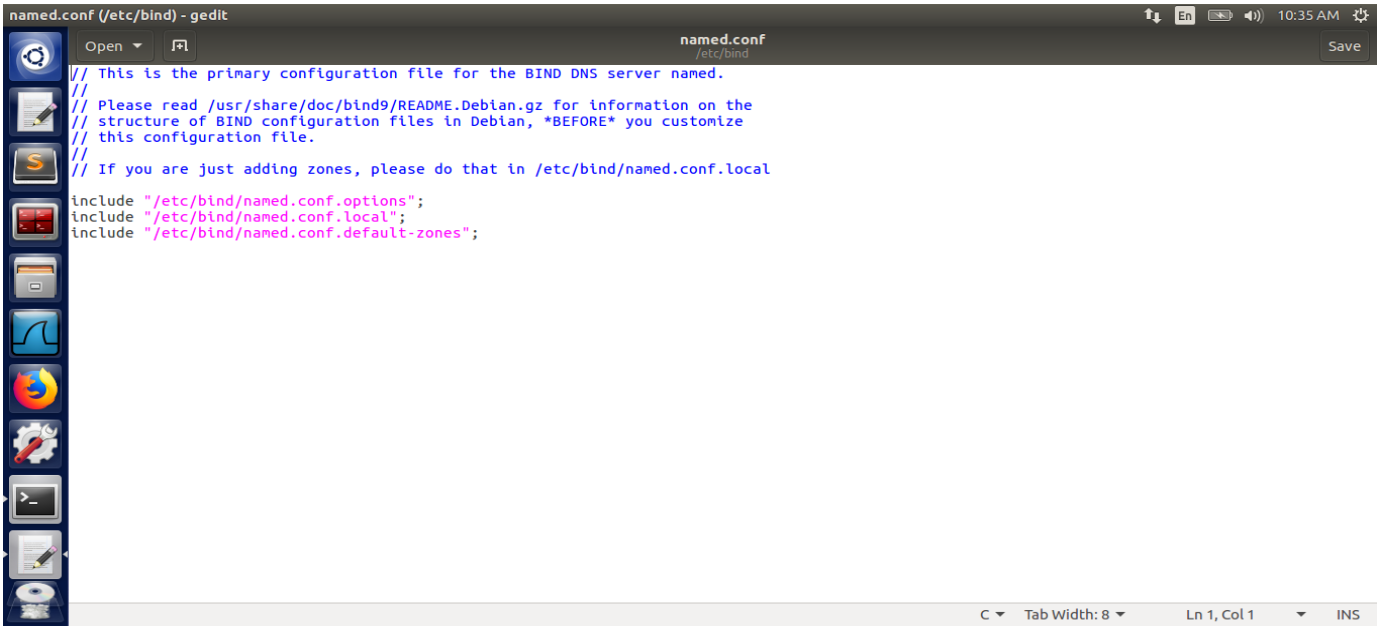
Step 1: Configure the BIND9 Server.

Step 2: Turnoff DNSSEC

Step 3: Fix the Source Ports

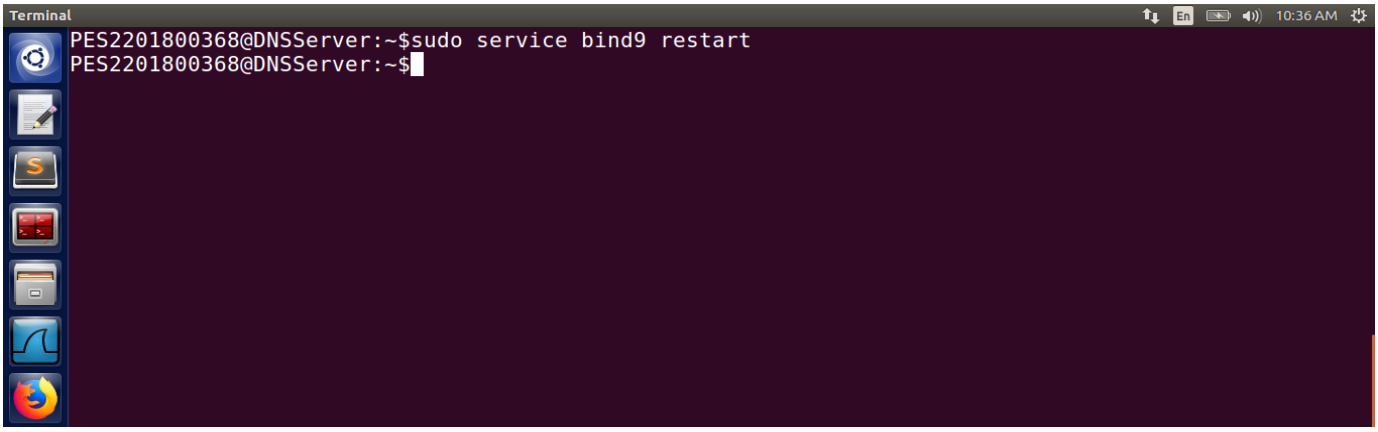
SCREENSHOT SHOWING THE CONFIGURATIONS MADE TO named.conf.options FILE

Step 4: Remove the example.com zone



SCREENSHOT SHOWING THE named.conf FILE AND THE example.com ZONE REMOVED

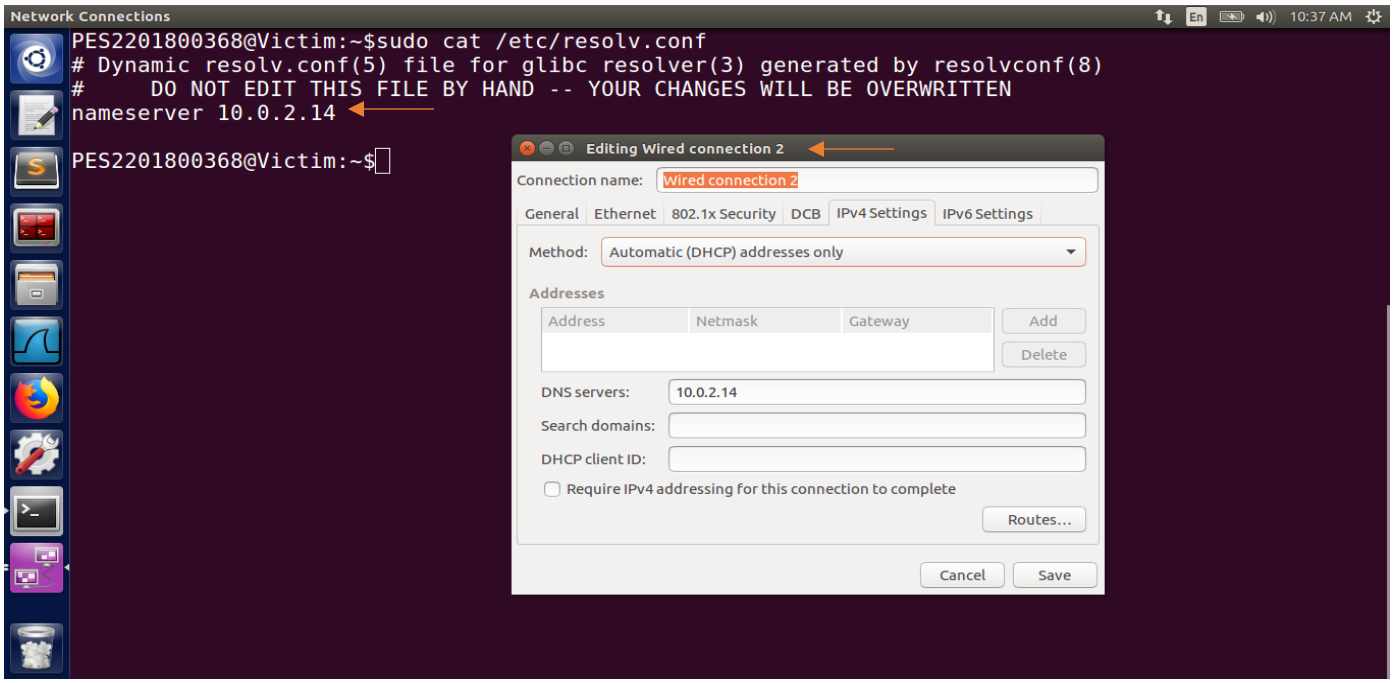
Step 5: Start DNS server



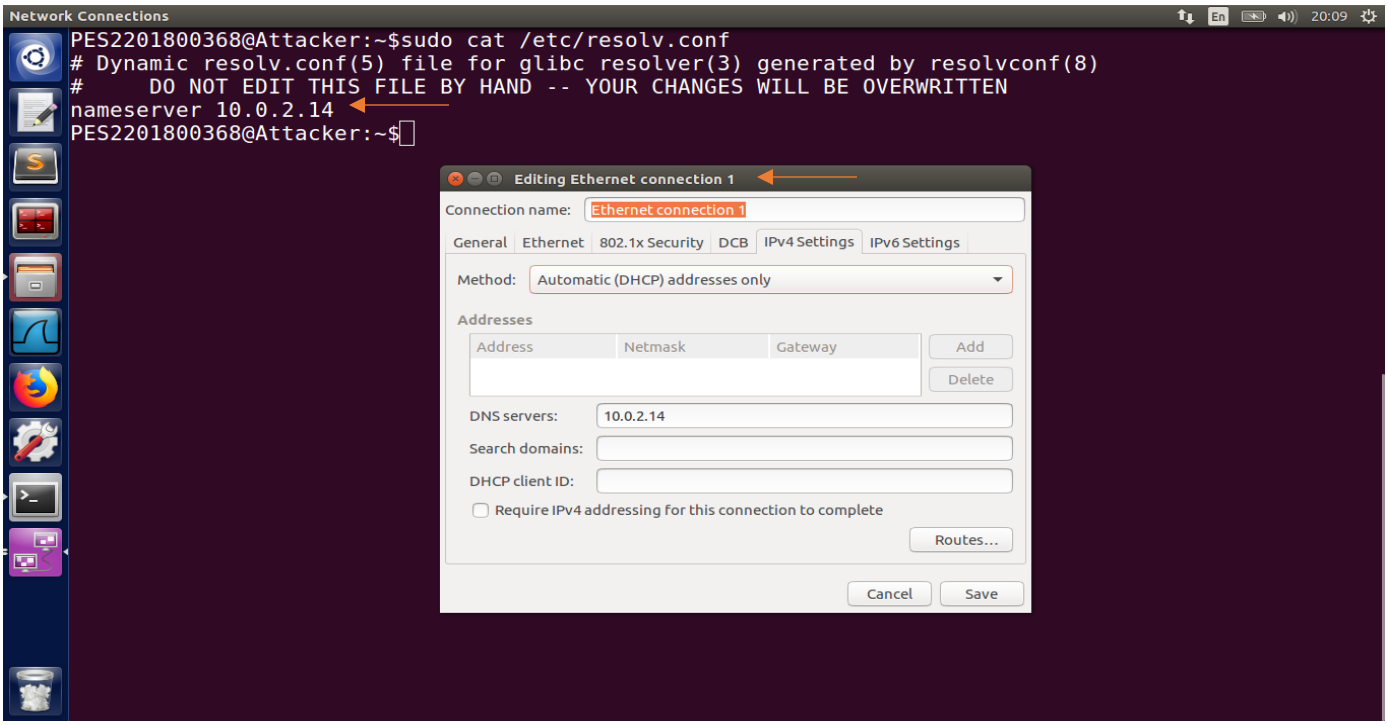
SCREENSHOT SHOWING THE SERVER RESTART

TASK 2: CONFIGURE THE VICTIM AND ATTACKER MACHINE

1. Open Edit Connection
2. Select IPv4 Settings
3. Choose Method as Automatic (DHCP) addresses only
4. Enter the IP Address of YOUR DNS Server in the DNS server field



SCREENSHOT SHOWING THE VICTIM MACHINE CONFIGURATION



SCREENSHOT SHOWING THE ATTACKER MACHINE CONFIGURATION

TASK 3.1 THE KAMINSKY ATTACK

Objective: To redirect the user to another machine B when the user tries to get to machine A using A's host name.

The Kaminsky attack:
We configure the attacker machine, so it uses the targeted DNS server as its defaults DNS Server as its default DNS server. The attacker machine is on the same NAT network.

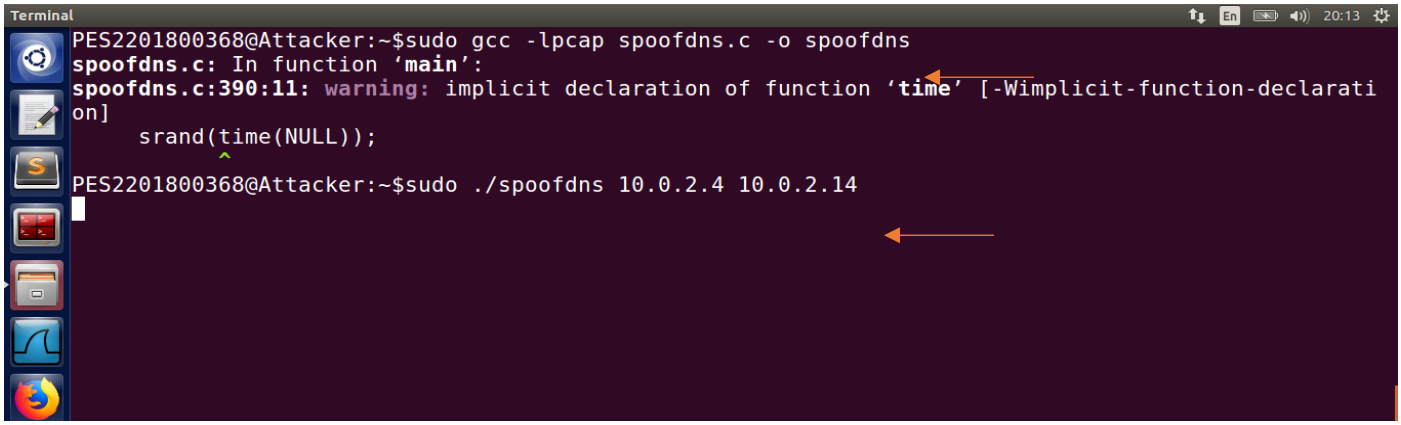
Task 1.1: Spoofing DNS Request

In this task, we will spoof DNS Requests that trigger the target DNS server to send out DNS queries, so we can spoof DNS replies.

Task 1.2: Spoofing DNS Replies

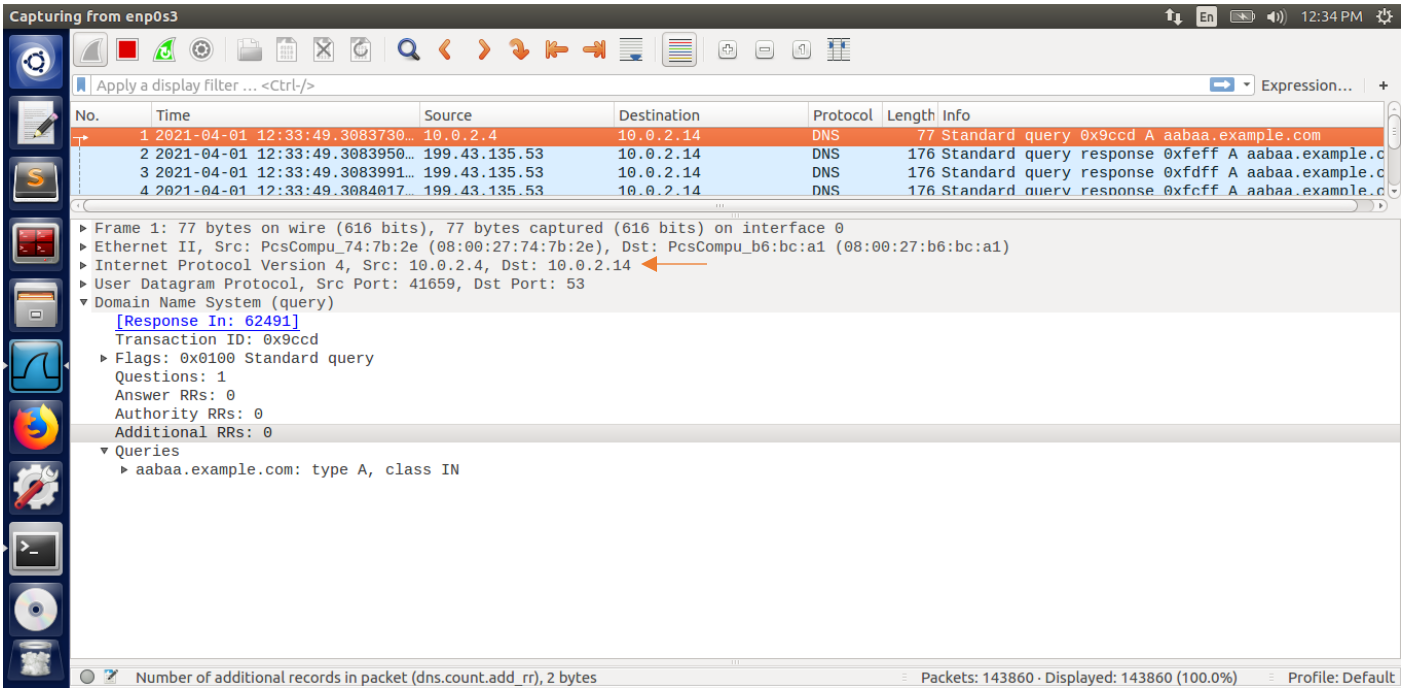
In this task, we will spoof DNS Responses to the local DNS Server for each query. We will create a DNS Header with DNS Payload with the Answer, Authority and Additional section. The answer section will give the IP address of the query domain, the authoritative section fills the authoritative nameserver for the query domain. So, after the attack is successful, any query with the domain name will be directed to the Attacker's nameserver "**ns.dnslabattacker.com**".

The C Code for both the above tasks have been included in the file named spoofdns.c and is run as a part of the attack from the Attacker VM. The two files can however be separated as request and response with a small correction made in the 'while' loop which sends out responses. The line while(count<100) sends out 100 spoofed responses; in the request file this can be set to while(count<0) thereby making it never execute during the request and for the response file it can be retained as while(count<100) and run.
The C Code along with the comments for all the steps can be found here:
https://drive.google.com/drive/folders/1CfGykfgH0GAVO-6CvomH2vi_P6S_teDx?usp=sharing

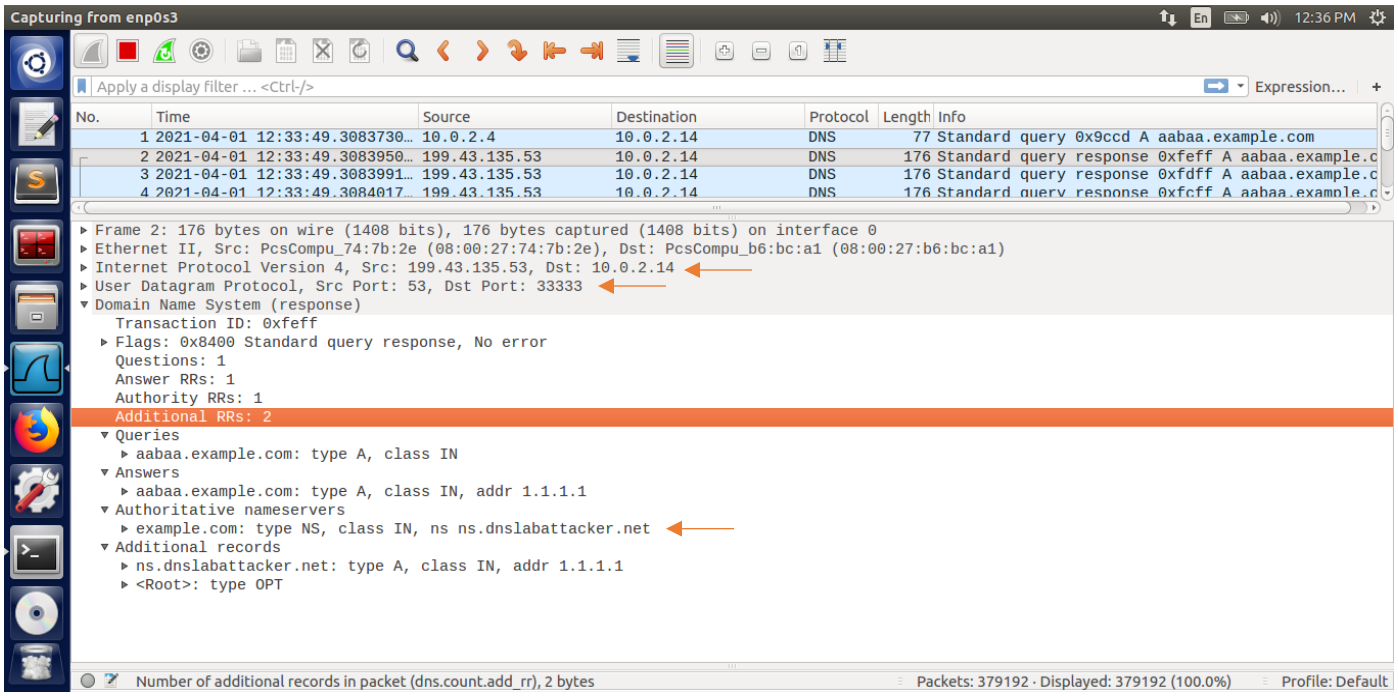


SCREENSHOT SHOWING THE RUNNING OF THE ATTACK FROM THE ATTACKER VM

WIRESHARK CAPTURE WHEN THE ATTACK IS RUNNING:



WIRESHARK CAPTURE SHOWING THE INITIAL DNS REQUEST GOING FROM THE VICTIM MACHINE(10.0.2.4) TO THE DNS SERVER MACHINE(10.0.2.14)



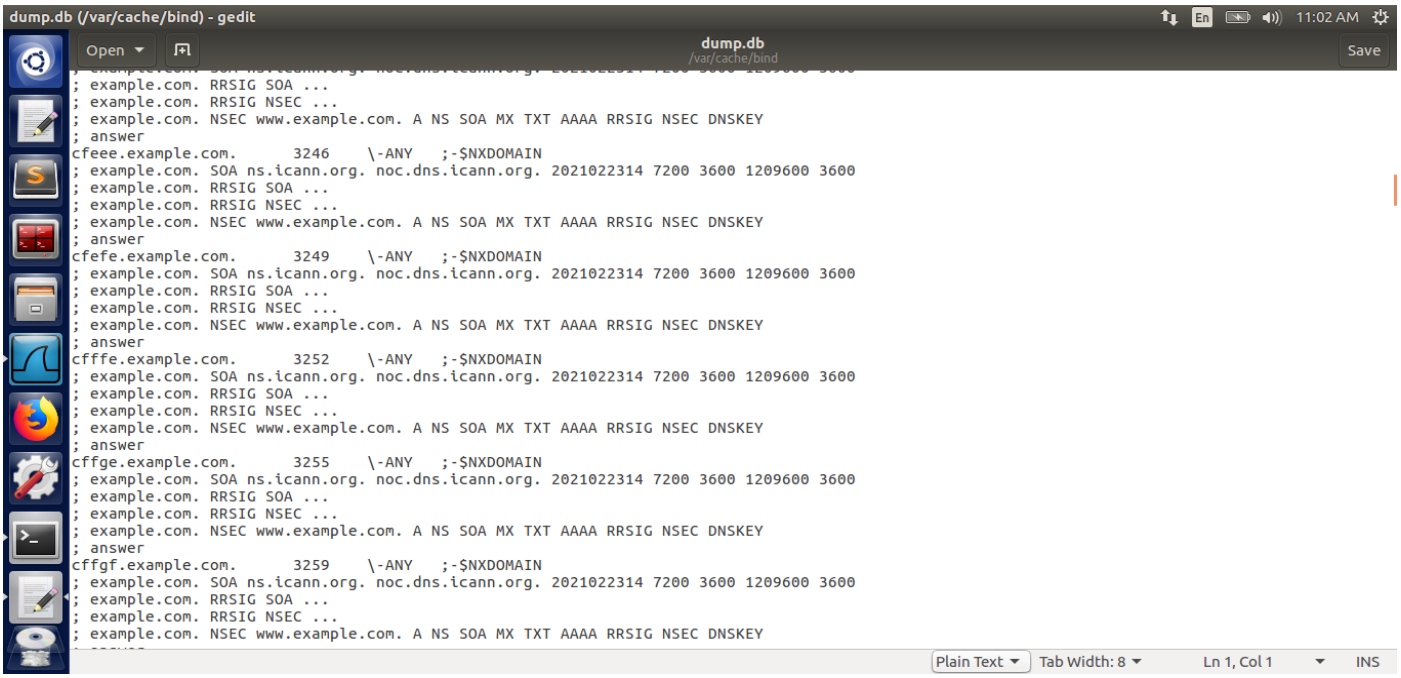
WIRESHARK CAPTURE SHOWING THE AUTHORITATIVE NAMESERVER AS ns.dnslabattacker.com(malicious site) FOR example.com SENT AS A SPOOFED RESPONSE TO THE DNS SERVER SO THAT THE RESPONSE IS CACHED

TASK 3.2: THE KAMINSKY ATTACK

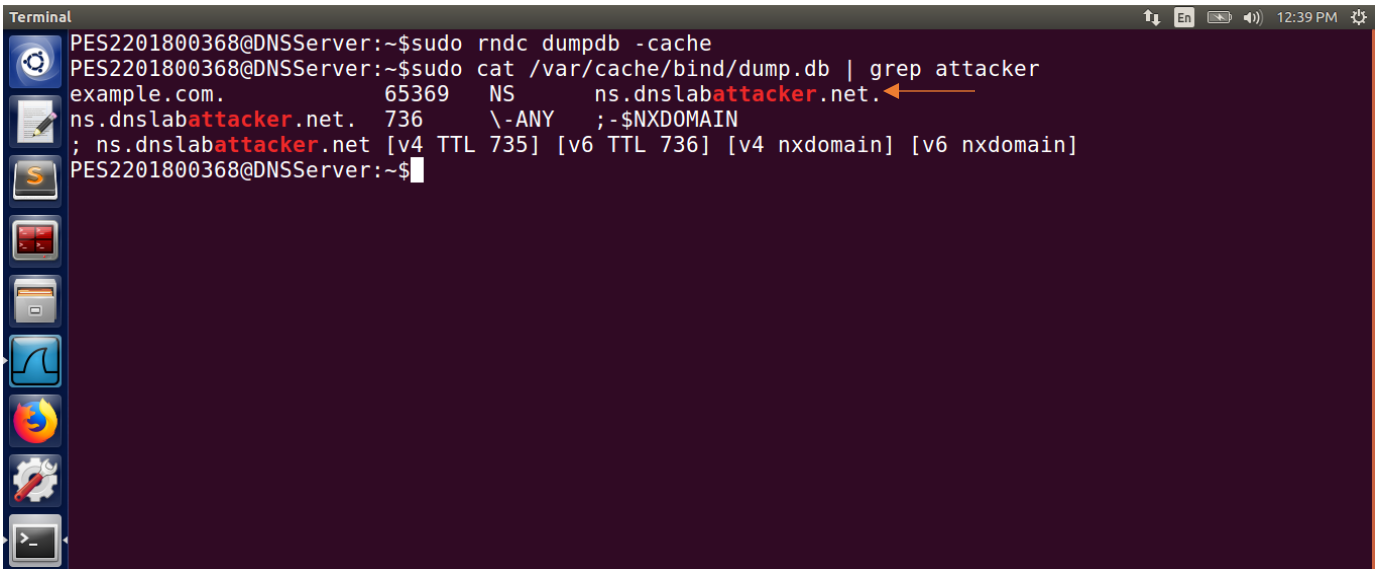
Aim: - To combine the above two tasks to perform Kaminsky Attack.

Check the DNS Cache in the DNS server machine

SCREENSHOT SHOWING THE COMMANDS TO CHECK THE CONTENTS OF THE CACHE



SCREENSHOT SHOWING THE CACHE CONTENTS



SCREENSHOT OF A BETTER LOOK INTO THE CACHE CONTENTS WHICH SHOWS ns.dnslabattacker.com TO BE THE NS for example.com AND HENCE PROVING THE ATTACK TO BE SUCCESSFUL

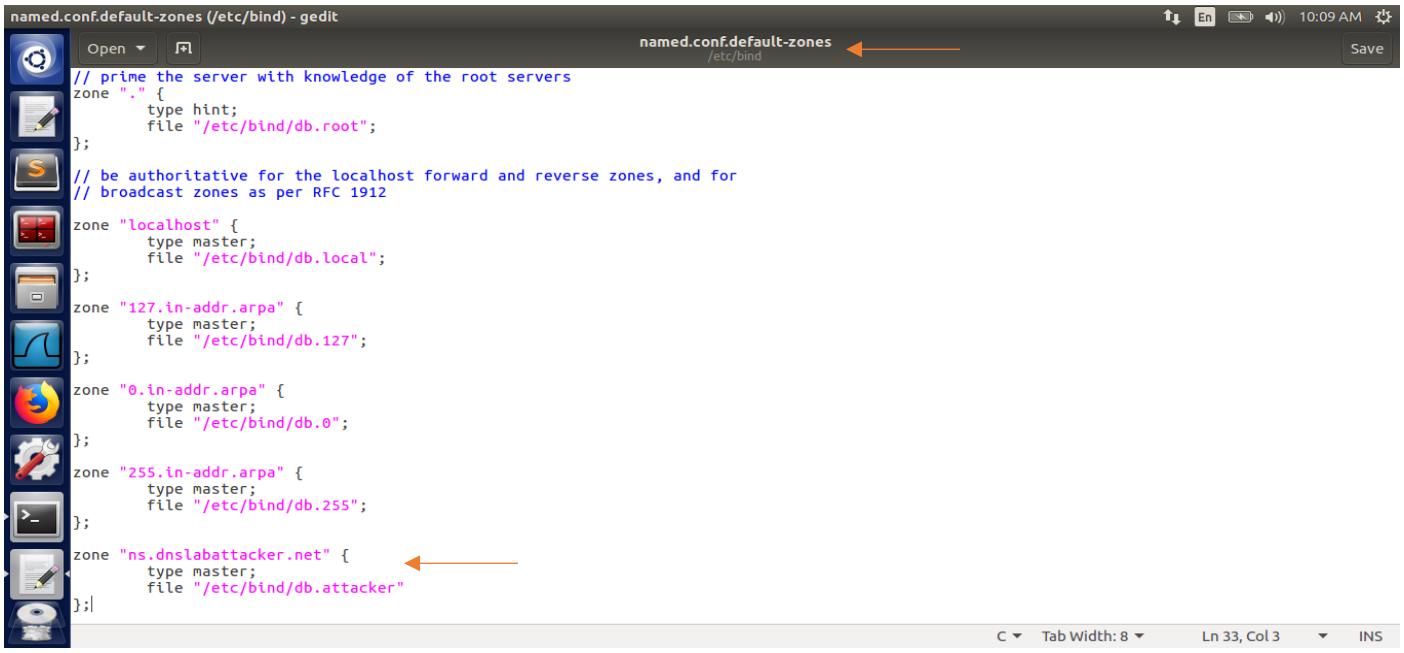
SCREENSHOT OF THE DIG COMMAND SHOWING A FAILURE AS ONLY THE QUESTION SECTION IS VISIBLE

In order to obtain the ns.dnslabattacker.net as the NS record for example.com in the dig command, we need to first configure the victim's DNS Server.

In order for the attack to work, the attacker needs their own domain name (reasons for this will become clearer after you see the explanation below). Since we do not own a real domain name, we can demonstrate the attack using our fake domain name ns.dnslabattacker.net and some extra

configuration on Target. We will basically add the **ns.dnslabattacker.net's** IP address to Target's DNS configuration, so Target does not need to go out asking for the IP address of this hostname from a non-existing domain. In a real-world setting, the Target's query would resolve to the attacker's server, which would be registered with a DNS registrar.

We first configure the victim's DNS server and add entries into the **named.conf.default-zones** in the /etc/bind/ folder:

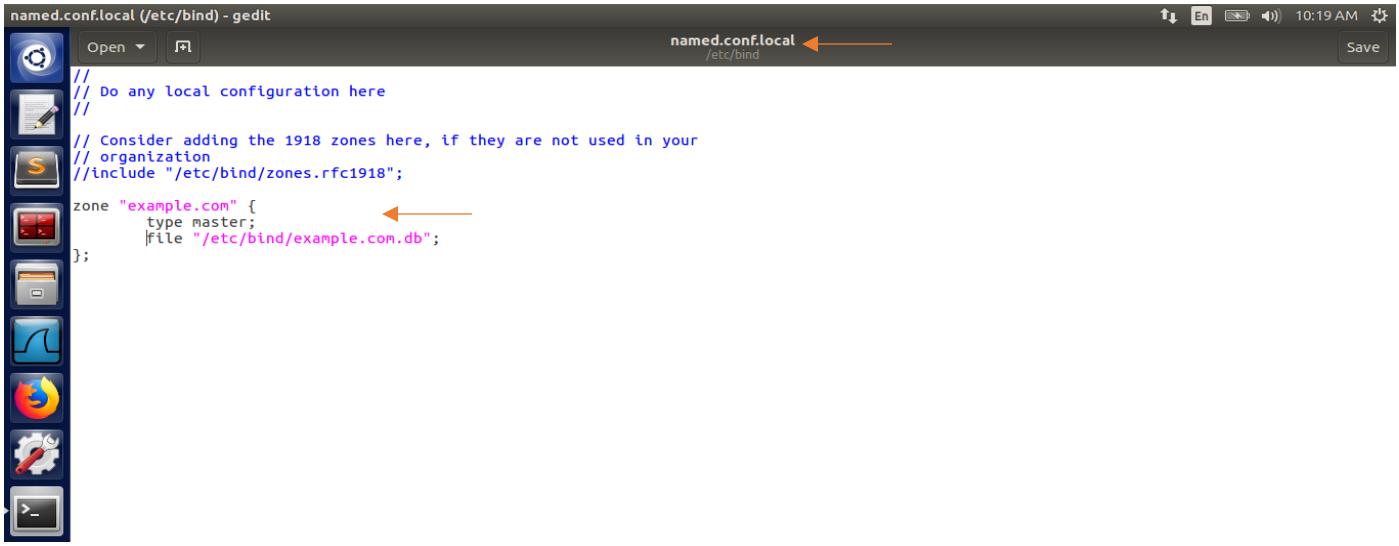


SCREENSHOT OF THE CONFIGURATIONS TO THE named.conf.default-zones FILE

We next Create the file /etc/bind/db.attacker, and place the following contents in it. We let the attacker's machine and ns.dnslabattacker.net share the machine (10.0.2.5).

SCREENSHOT SHOWING THE CREATION OF THE FILE NAMED db.attacker

We need to configure the DNS server, so it answers the queries for the domain example.com and hence the following entries are added in /etc/bind/named.conf.local

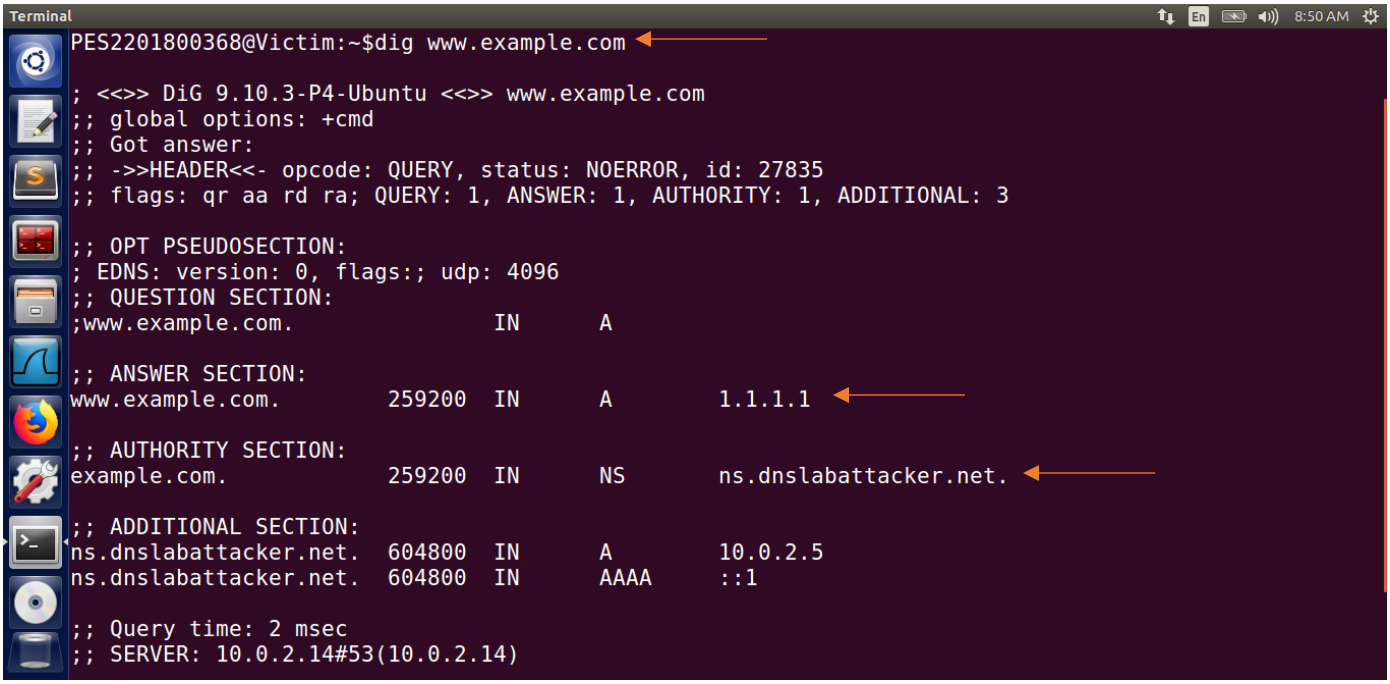


SCREENSHOT OF THE CONFIGURATIONS MADE TO THE named.conf.local FILE

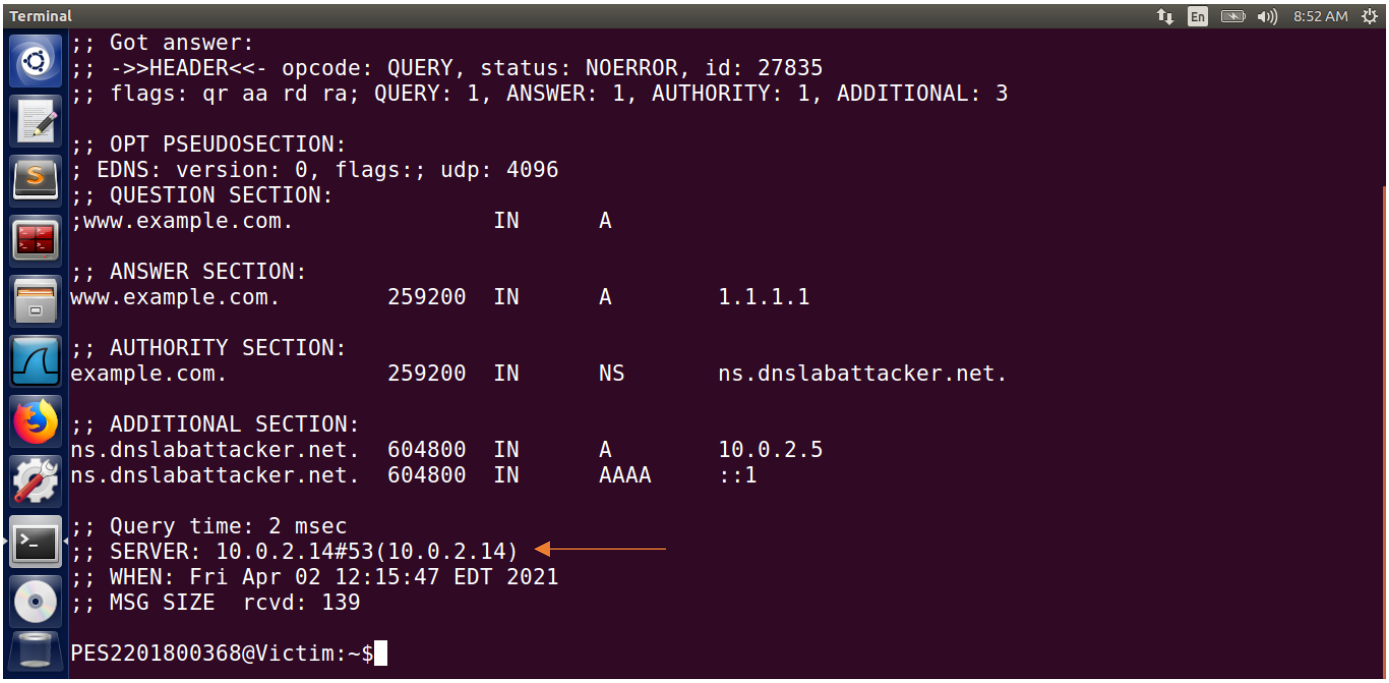
We now Create a file called /etc/bind/example.com.db, and fill it with the following contents.

SCREENSHOT SHOWING THE CREATION OF FILE example.com.db WITH THE ABOVE SEEN CONTENTS

When the configurations are finished, restart the DNS Server of the DNS Server VM; otherwise, the modification will not take effect. If everything is done properly, you can use the command "dig www.example.com" on the user machine. The reply would be 1.1.1.1, which is exactly we put in the above file.



```
Terminal                                                                    8:50 AM
PES2201800368@Victim:~$dig www.example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27835
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.dnslabattacker.net.

;; ADDITIONAL SECTION:
ns.dnslabattacker.net.  604800  IN      A       10.0.2.5
ns.dnslabattacker.net.  604800  IN      AAAA    ::1

;; Query time: 2 msec
;; SERVER: 10.0.2.14#53(10.0.2.14)
```



```
Terminal                                                                    8:52 AM
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27835
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.dnslabattacker.net.

;; ADDITIONAL SECTION:
ns.dnslabattacker.net.  604800  IN      A       10.0.2.5
ns.dnslabattacker.net.  604800  IN      AAAA    ::1

;; Query time: 2 msec
;; SERVER: 10.0.2.14#53(10.0.2.14)
;; WHEN: Fri Apr 02 12:15:47 EDT 2021
;; MSG SIZE  rcvd: 139

PES2201800368@Victim:~$
```

SCREENSHOTS SHOWING THE EXECUTION OF THE DIG COMMAND

Observations:

The Kaminsky Attack is also called as the remote DNS cache poisoning attack. To overcome the challenges of time of spoofing and cache effect which made remote DNS attack not feasible, Kaminsky's idea was to:

- Ask a different question every time, so caching the answer does not matter, and the local DNS server will send out a new query each time.

- Provide forged answer in the Authority section (Hijack the entire domain - Nameserver)
- To achieve the above ideas, we used the C code in order to trigger the DNS Server to send out queries and then keep sending out spoofed responses continuously, so that if one attempt fails and the actual DNS reply gets cached, the attacker need not wait for the cache to timeout for the next attempt.

The following observations can be made from the dig command:

1) We see that the IP address of example.com is given as 1.1.1.1 in the ANSWER SECTION according to the configurations we made in the example.com.db FILE.

2) We see that the NS(NameServer) for the example.com domain is seen to be ns.dnslabattacker.net in the AUTHORITY SECTION which was the main intent of our attack.

3) We also see that the ADDITIONAL SECTION contains the IP address 10.0.2.5 (our attacker's IP) as the IP address for ns.dnslabattacker.com so that the queries can be redirected to the attacker machine with a malicious website.

4) Finally all these results are coming from our victim's local DNS Server which was poisoned on account of our attack.