

# Computer Networks Lab

UE19CS256

Week 4

Name: Sreenath Saikumar

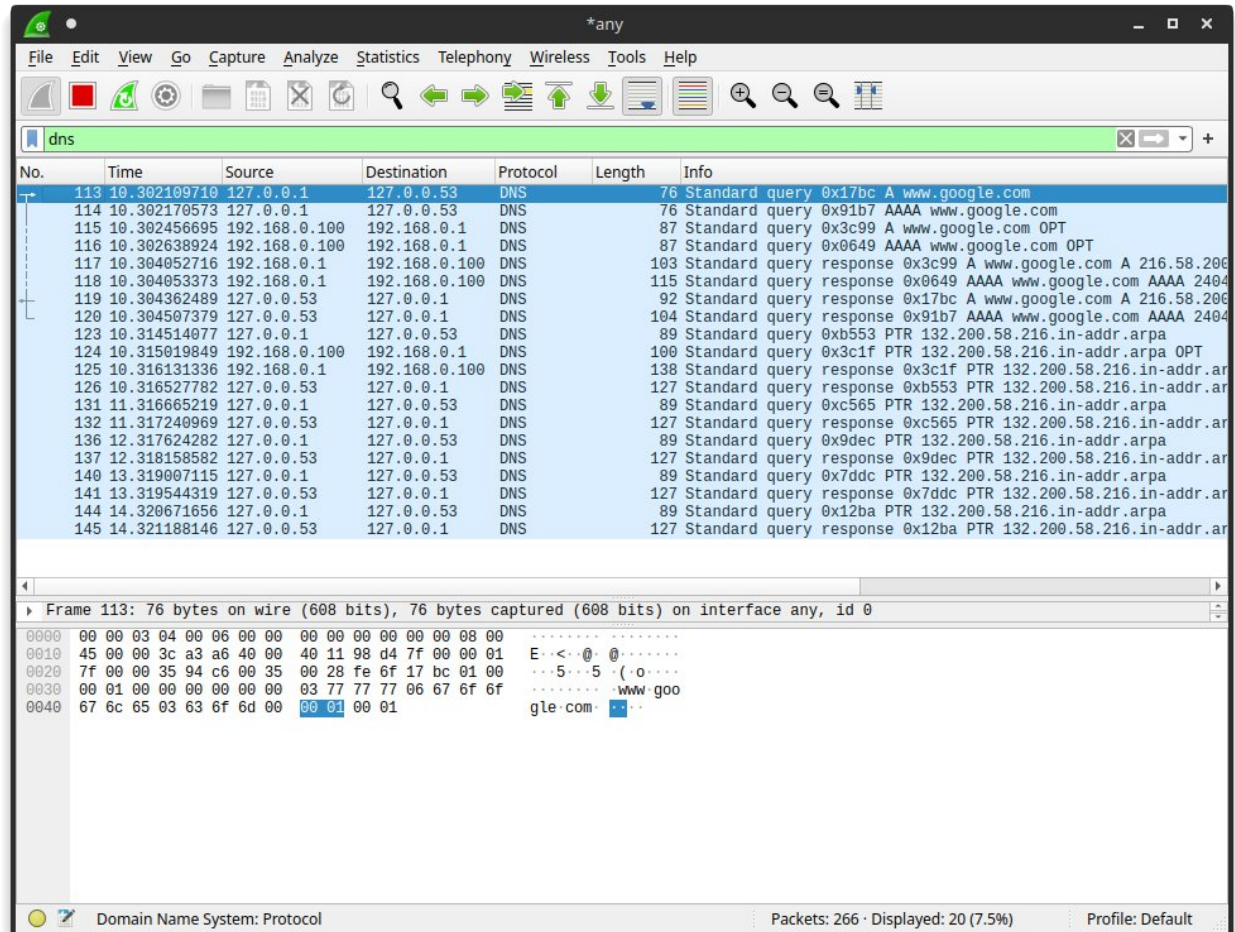
Semester: 4      Section :G

**SRN:** PES2UG19CS406

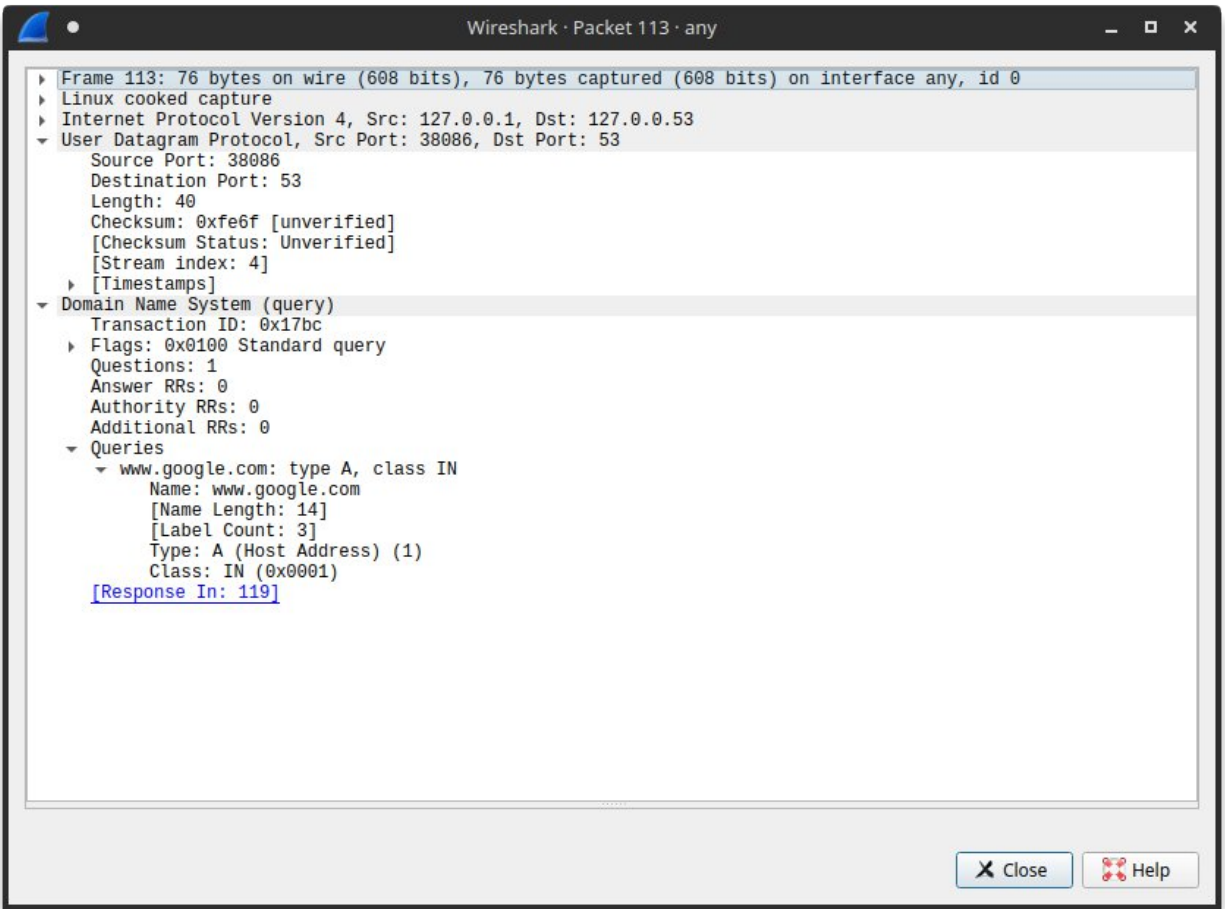
Date: 21/02/2021

## 1. Pinging Using default DNS

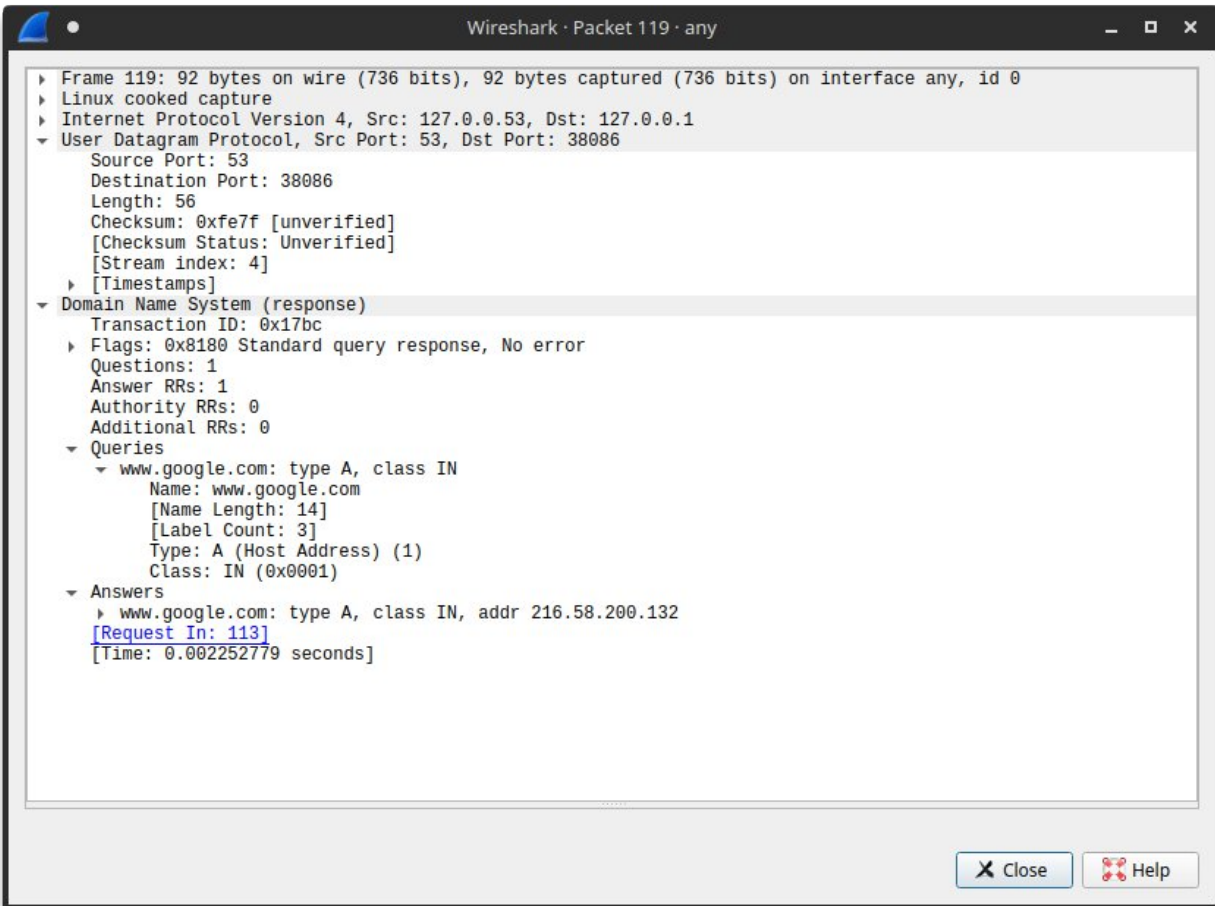
We use the `ping` command to connect to `www.google.com` and use Wireshark to capture the packets. The IP address of the default/ local DNS server is `127.0.0.53`.



Wireshark capture



DNS Query



### DNS Query Response

The query is type A (authoritative). The response has an answers field which contains the IP address of the destination (`www.google.com`) `216.58.200.132`.

## 2. Setting up a local DNS Server

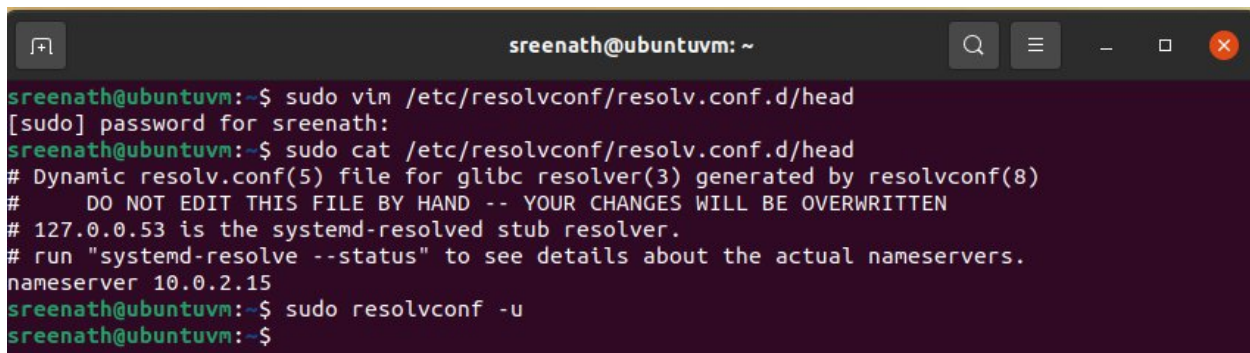
### a) Configure Client Machine

The IP address of the server is `192.168.0.100` and the IP address of the client used for this task is `192.168.0.104`.

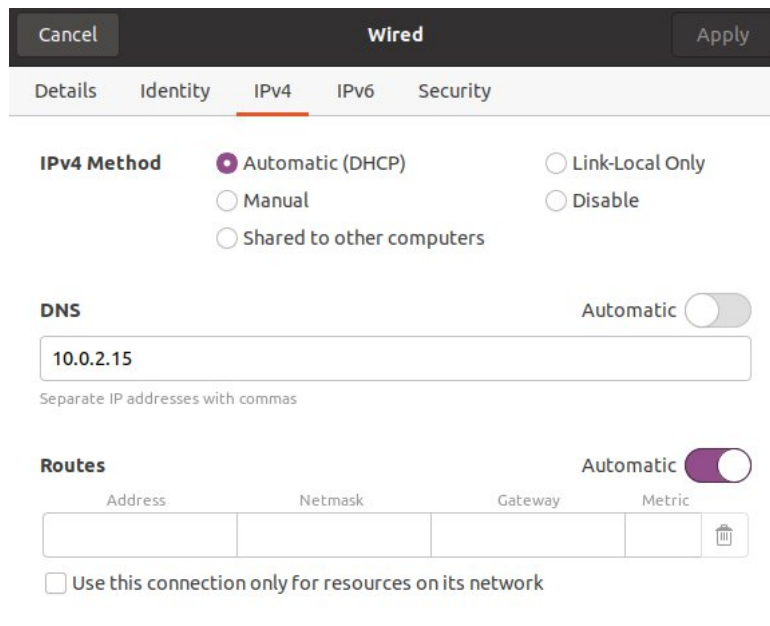
We can add a custom DNS Server by changing the resolver configuration in the `/etc/resolvconf/resolv.conf.d/head`.

We also change the DNS Server manually in the IPv4 Settings GUI.

To apply the changes, use the command `sudo resolvconf -u`

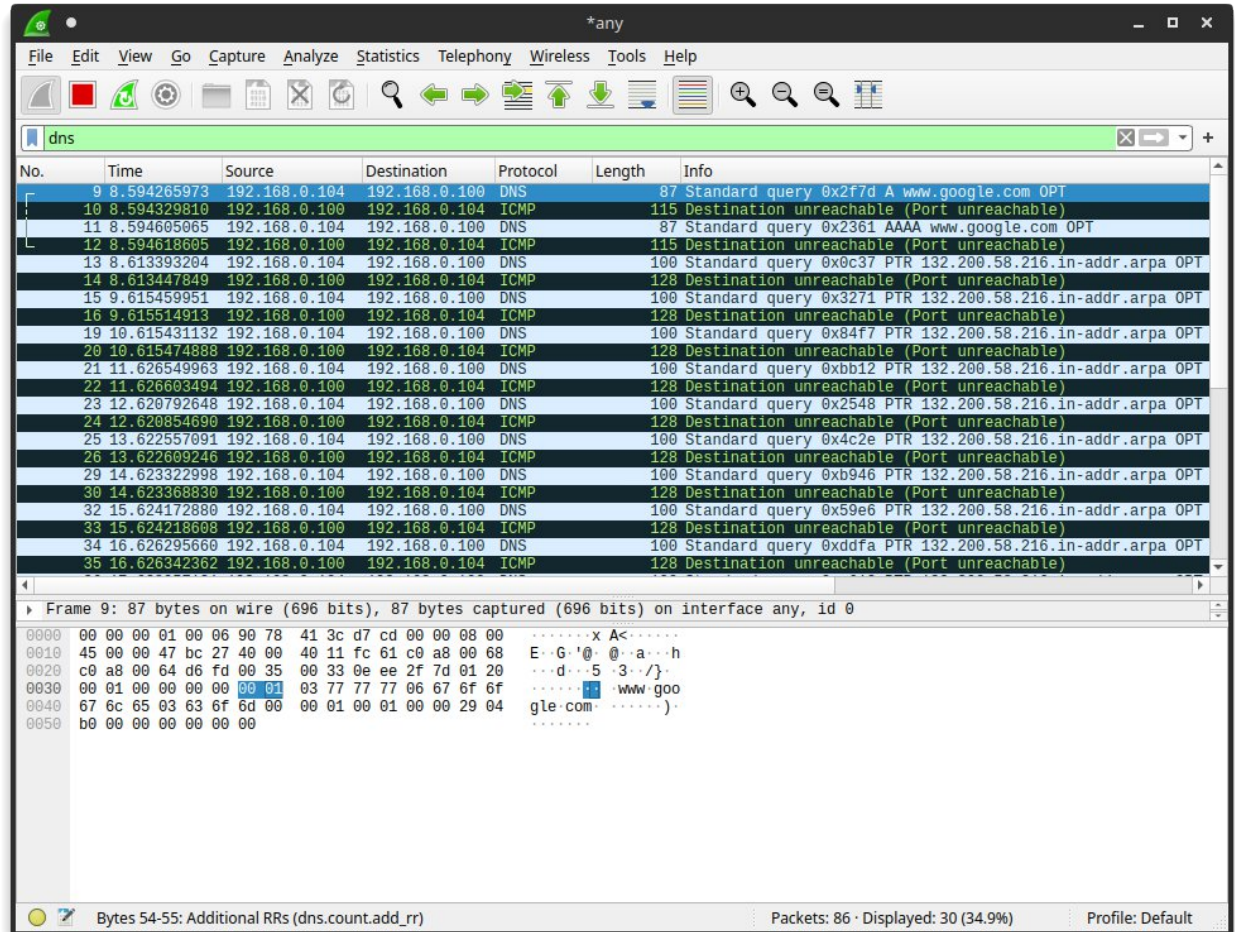
A terminal window titled 'sreenath@ubuntuvvm: ~' with standard Ubuntu window controls. The terminal shows the following commands and output:

```
sreenath@ubuntuvvm:~$ sudo vim /etc/resolvconf/resolv.conf.d/head
[sudo] password for sreenath:
sreenath@ubuntuvvm:~$ sudo cat /etc/resolvconf/resolv.conf.d/head
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
# 127.0.0.53 is the systemd-resolved stub resolver.
# run "systemd-resolve --status" to see details about the actual nameservers.
nameserver 10.0.2.15
sreenath@ubuntuvvm:~$ sudo resolvconf -u
sreenath@ubuntuvvm:~$
```

A screenshot of the 'Wired' network settings window. The 'IPv4' tab is selected. Under 'IPv4 Method', 'Automatic (DHCP)' is selected. Under 'DNS', the 'Automatic' toggle is turned off, and the text '10.0.2.15' is entered in the field below. Under 'Routes', the 'Automatic' toggle is turned on. At the bottom, there is a checkbox labeled 'Use this connection only for resources on its network' which is unchecked. The window has 'Cancel' and 'Apply' buttons at the top.

**Here, the IP is 10.0.2.15 but prior to this, it was `192.168.0.104`. This change was made since I went from using 2 physical machines to using 2 VM's.**

Now, we ping `www.google.com` again and observe the wireshark capture.



We get a **Destination Unreachable Error** since there is no record of Google's IP on the server i.e. our server machine and will try to fallback onto the default `127.0.0.53` IP.

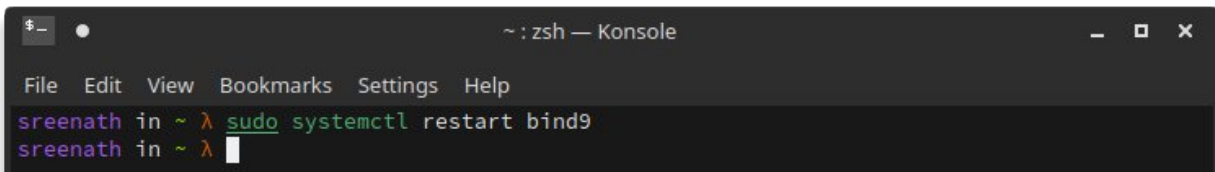


## b) Setting up a Local DNS Server

We use the `bind9` server as the DNS server on our local server machine. Install it using the command `sudo <packagemanager> install bind9`.

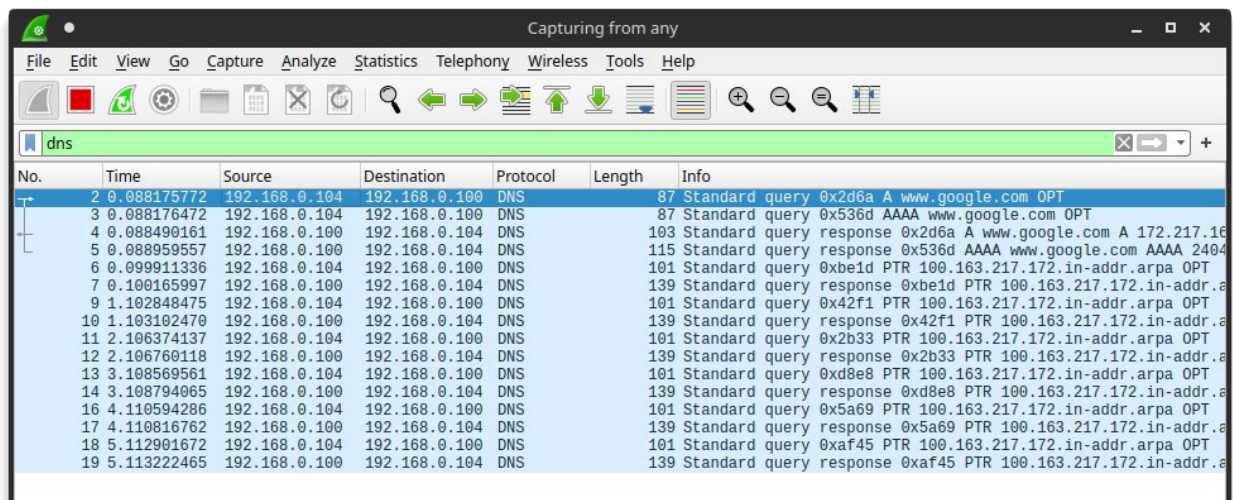
We can also edit the `named.conf.options` file in the `/etc/bind/` directory to specify where we want to store our DNS dump files but the default file stored in the `/var/cache/bind/named_dump.db` will be used.

Now restart the server using `sudo systemctl restart bind9` or `sudo service bind9 restart`.



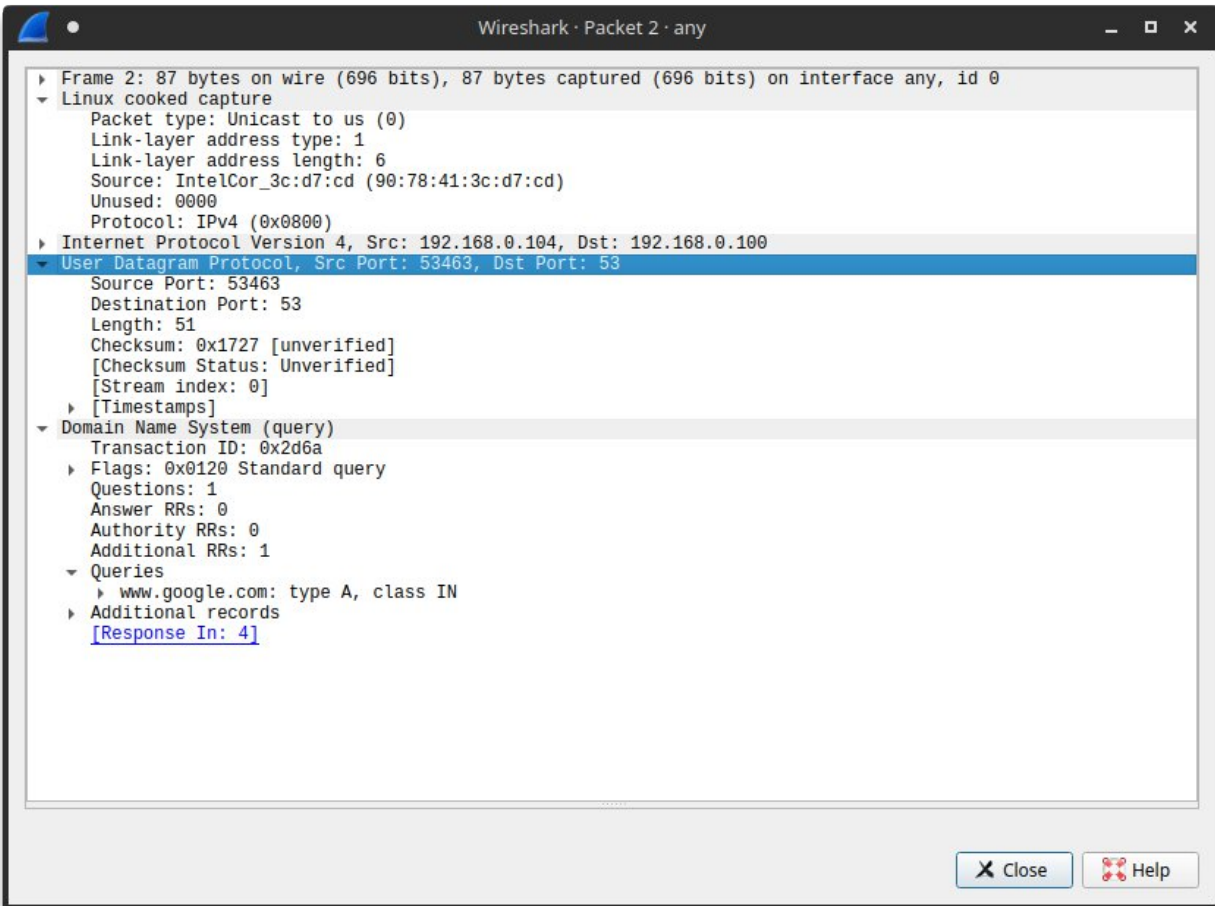
```
$ - ~: zsh — Konsole
File Edit View Bookmarks Settings Help
sreenath in ~ λ sudo systemctl restart bind9
sreenath in ~ λ
```

Now, we ping `www.google.com` again from our client machine( 192.168.0.104) and observe the Wireshark capture.



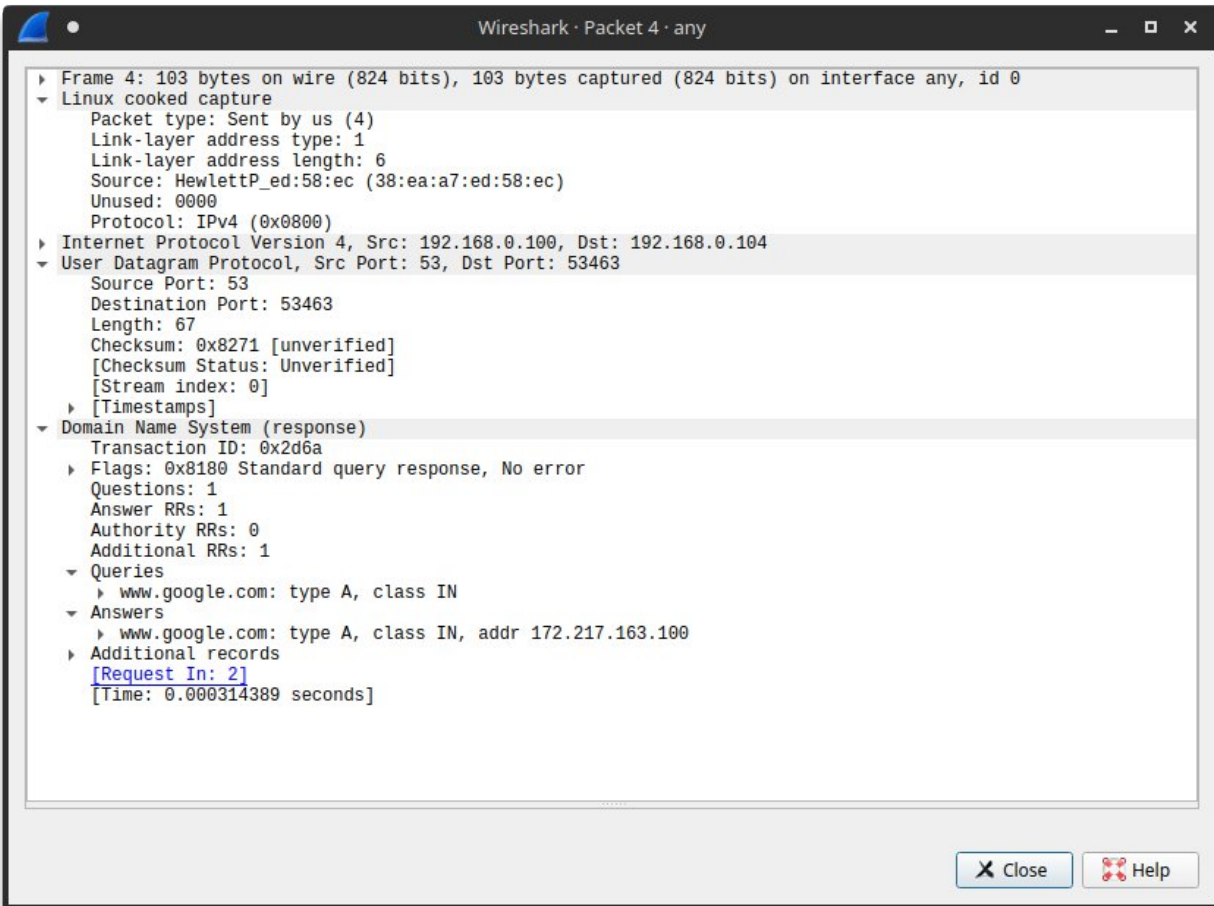
No.	Time	Source	Destination	Protocol	Length	Info
2	0.088175772	192.168.0.104	192.168.0.100	DNS	87	Standard query 0x2d6a A www.google.com OPT
3	0.088176472	192.168.0.104	192.168.0.100	DNS	87	Standard query 0x536d AAAA www.google.com OPT
4	0.088490161	192.168.0.100	192.168.0.104	DNS	103	Standard query response 0x2d6a A www.google.com A 172.217.16
5	0.088959557	192.168.0.100	192.168.0.104	DNS	115	Standard query response 0x536d AAAA www.google.com AAAA 2404
6	0.099911336	192.168.0.104	192.168.0.100	DNS	101	Standard query 0xbe1d PTR 100.163.217.172.in-addr.arpa OPT
7	0.100165997	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0xbe1d PTR 100.163.217.172.in-addr.a
9	1.102848475	192.168.0.104	192.168.0.100	DNS	101	Standard query 0x42f1 PTR 100.163.217.172.in-addr.arpa OPT
10	1.103102470	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0x42f1 PTR 100.163.217.172.in-addr.a
11	2.106374137	192.168.0.104	192.168.0.100	DNS	101	Standard query 0x2b33 PTR 100.163.217.172.in-addr.arpa OPT
12	2.106760118	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0x2b33 PTR 100.163.217.172.in-addr.a
13	3.108569561	192.168.0.104	192.168.0.100	DNS	101	Standard query 0xd8e8 PTR 100.163.217.172.in-addr.arpa OPT
14	3.108794065	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0xd8e8 PTR 100.163.217.172.in-addr.a
16	4.110594286	192.168.0.104	192.168.0.100	DNS	101	Standard query 0x5a69 PTR 100.163.217.172.in-addr.arpa OPT
17	4.110816762	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0x5a69 PTR 100.163.217.172.in-addr.a
18	5.112901672	192.168.0.104	192.168.0.100	DNS	101	Standard query 0xaf45 PTR 100.163.217.172.in-addr.arpa OPT
19	5.113222465	192.168.0.100	192.168.0.104	DNS	139	Standard query response 0xaf45 PTR 100.163.217.172.in-addr.a

We now see that our client machine clearly uses the local DNS Server that we have set up with A type records.



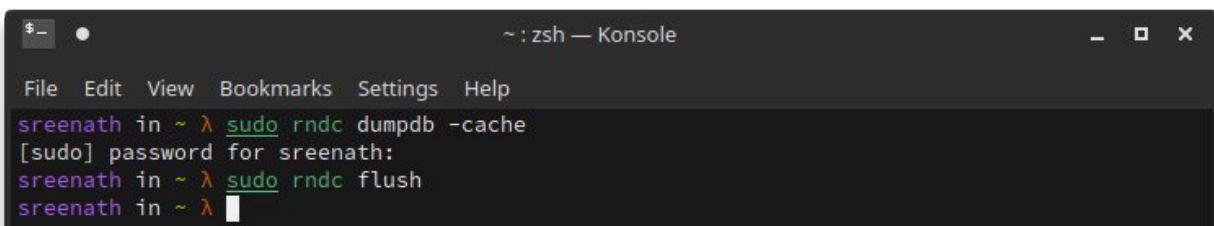
DNS Query





## DNS Response

We now use the commands `sudo rndc dumpdb -cache` & `sudo rndc flush` to dump the cache to the `named_dump.db` file and then flush the DNS cache.



```
$ -  ~: zsh — Konsole
File Edit View Bookmarks Settings Help
sreenath in ~ λ cat /var/cache/bind/named_dump.db | grep google
217.172.in-addr.arpa. 129374 NS ns1.google.com.
                      129374 NS ns2.google.com.
                      129374 NS ns3.google.com.
                      129374 NS ns4.google.com.
; 163.217.172.in-addr.arpa. SOA ns1.google.com. dns-admin.google.com. 358138008 900 900 1
800 60
google.com.          215771 NS ns1.google.com.
                      215771 NS ns2.google.com.
                      215771 NS ns3.google.com.
                      215771 NS ns4.google.com.
ns1.google.com.       215771 A 216.239.32.10
ns2.google.com.       215771 A 216.239.34.10
ns3.google.com.       215771 A 216.239.36.10
ns4.google.com.       215771 A 216.239.38.10
www.google.com.       43271  A 172.217.163.100
sreenath in ~ λ
```

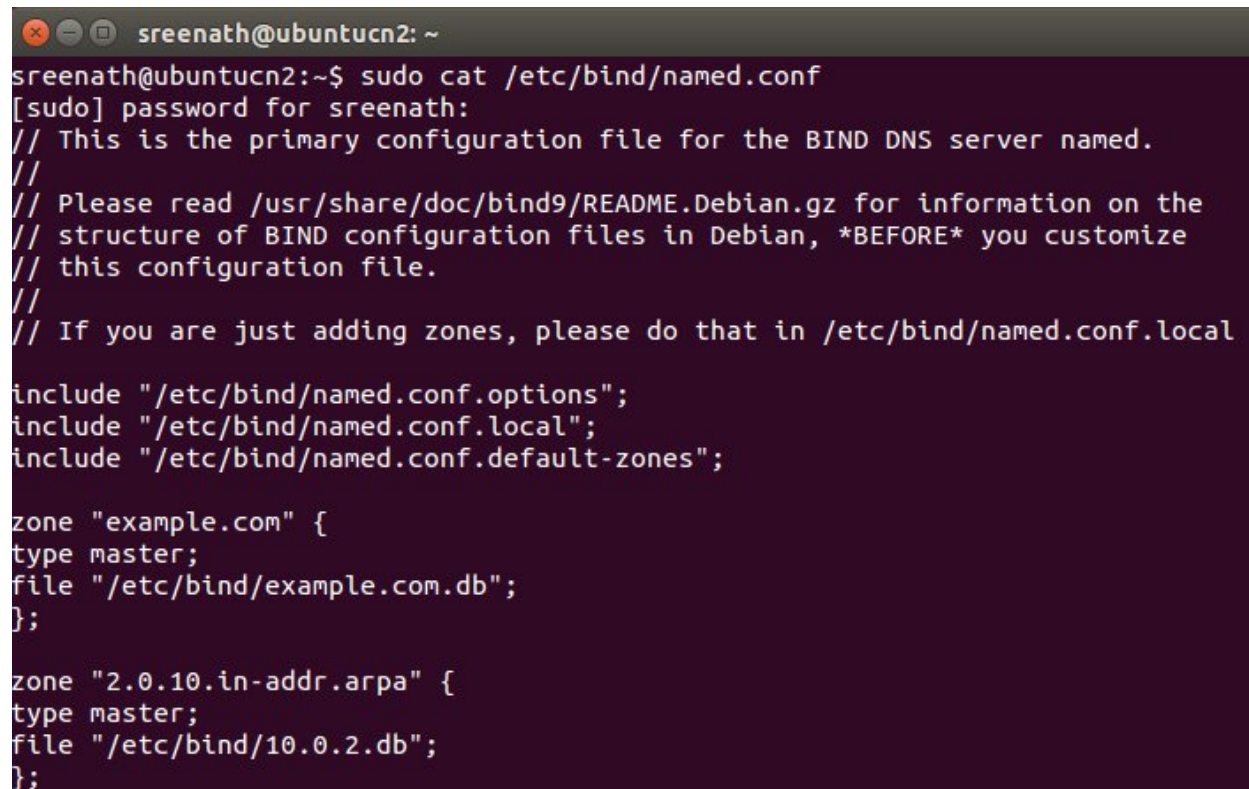
On grepping for google in the DNS cachedump

### 3. Setting up an Authoritative Nameserver for the example.com Domain

#### a) Creating a Zone in the local DNS Server

We add 2 zone entries to the `/etc/bind/named.conf` files on the server to create 2 zones. The 1st zone corresponds to the forward lookup (from hostname to IP) and the 2nd zone is for reverse lookup (from IP to hostname).

**The IP's used from now onwards are `10.0.2.15` for the Server and `10.0.2.4` for the Client machine as they are running on VM's.**

A terminal window titled 'sreenath@ubuntucn2: ~' shows the command 'sudo cat /etc/bind/named.conf' being executed. The output displays the contents of the file, which includes comments about the primary configuration file and instructions to read the README. It also shows the inclusion of 'named.conf.options', 'named.conf.local', and 'named.conf.default-zones'. Two zone definitions are visible: 'example.com' as a master zone pointing to '/etc/bind/example.com.db', and '2.0.10.in-addr.arpa' as a master zone pointing to '/etc/bind/10.0.2.db'.

```
sreenath@ubuntucn2:~$ sudo cat /etc/bind/named.conf
[sudo] password for sreenath:
// 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 "example.com" {
    type master;
    file "/etc/bind/example.com.db";
};

zone "2.0.10.in-addr.arpa" {
    type master;
    file "/etc/bind/10.0.2.db";
};
```

10.0.2.0 is the subnet mask of our IP address, hence the reverse of that is used in the reverse lookup.

## Forward and Reverse Lookup

The forward lookup file is located in `/etc/bind/example.com.db`.

The `@` symbol is used to specify the origin, which is `example.com` in this case. There are 7 resource records in the lookup file, including an SOA (Start of Authority) RR, an NS (Name Server) RR, an MX (Mail Exchanger) RR and 4 A (host Addresses) RR's.

```
sreenath@ubuntucn2: ~  
sreenath@ubuntucn2:~$ sudo cat /etc/bind/example.com.db  
$TTL 3D  
@      IN      SOA  ns.example.com. admin.example.com. (  
                2008111001  
                8H  
                2H  
                4W  
                1D)  
  
@      IN      NS   ns.example.com.  
@      IN      MX   10 mail.example.com.  
  
www    IN      A    10.0.2.101  
mail   IN      A    10.0.2.102  
ns     IN      A    10.0.2.10  
*.example.com. IN  A  10.0.2.100  
sreenath@ubuntucn2:~$
```

`example.com.db` file

The reverse lookup file is located in `/etc/bind/10.0.2.db`.

This is used to translate IP addresses to hostnames for a given domain.

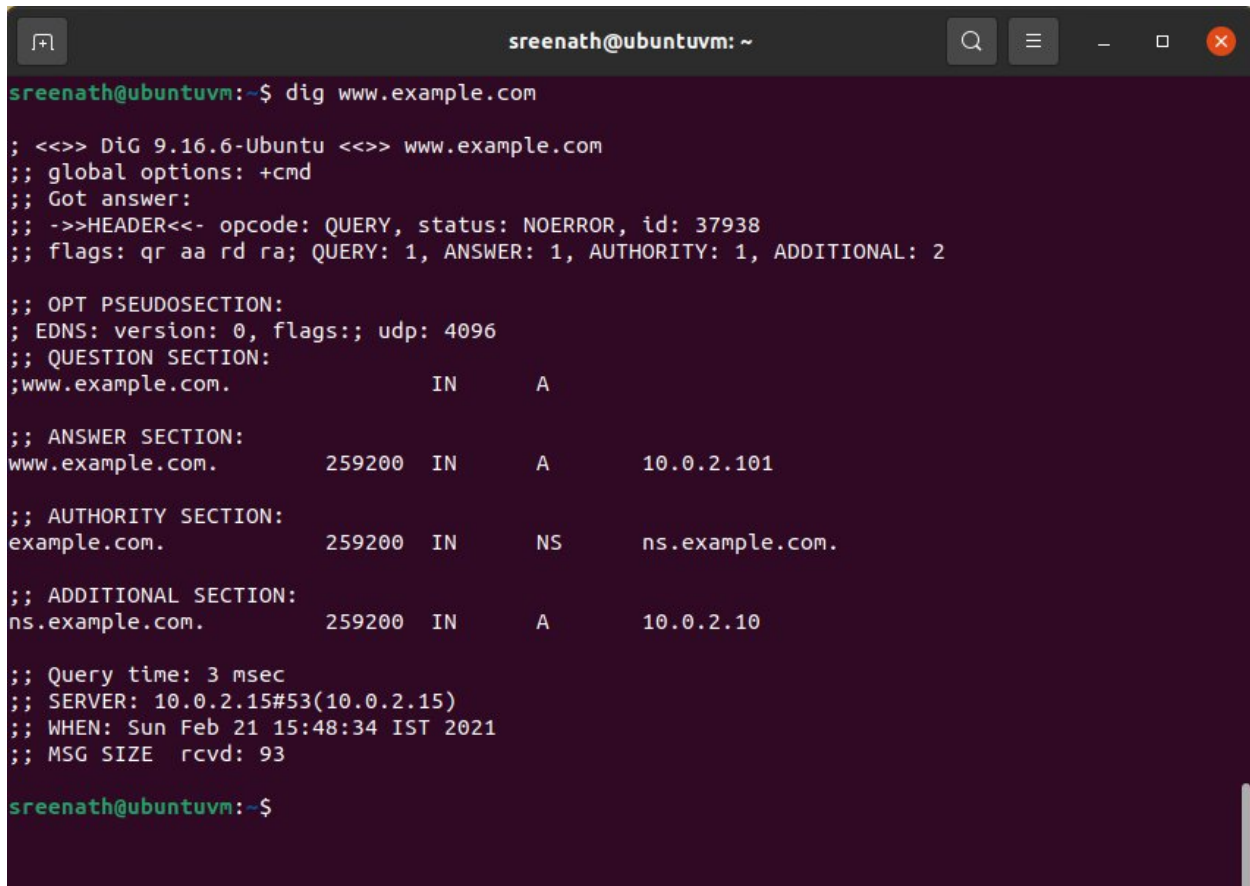
For each IP address defined in the forward lookup file, a corresponding hostname is specified here. The record type here is PTR or DNS Pointer Record.

```
sreenath@ubuntucn2: ~  
sreenath@ubuntucn2:~$ sudo cat /etc/bind/10.0.2.db  
$TTL 3D  
@      IN      SOA  ns.example.com. admin.example.com (  
                2008111001  
                8H  
                2H  
                4W  
                1D)  
  
@      IN      NS   ns.example.com.  
  
101    IN      PTR  www.example.com.  
102    IN      PTR  mail.example.com.  
10     IN      PTR  ns.example.com.  
sreenath@ubuntucn2:~$
```

Once the files have been created, restart the `bind9` server.

## Testing the DNS Server using DIG

On the client machine, use the `dig` command which is used to lookup name servers stored in the `/etc/resolv.conf` file. We use Wireshark on the server to capture the packets received when we run the command `dig www.example.com`

A terminal window titled 'sreenath@ubuntuvm: ~' showing the output of the 'dig www.example.com' command. The output is a detailed DNS query and response. The query section shows 'www.example.com. IN A'. The answer section shows 'www.example.com. 259200 IN A 10.0.2.101'. The authority section shows 'example.com. 259200 IN NS ns.example.com.'. The additional section shows 'ns.example.com. 259200 IN A 10.0.2.10'. The query time is 3 msec, the server is 10.0.2.15#53(10.0.2.15), and the message size received is 93 bytes.

```
sreenath@ubuntuvm:~$ dig www.example.com

; <<>> DiG 9.16.6-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37938
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.com.                IN      A

;; ANSWER SECTION:
www.example.com.                259200  IN      A      10.0.2.101

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

;; ADDITIONAL SECTION:
ns.example.com.                 259200  IN      A      10.0.2.10

;; Query time: 3 msec
;; SERVER: 10.0.2.15#53(10.0.2.15)
;; WHEN: Sun Feb 21 15:48:34 IST 2021
;; MSG SIZE rcvd: 93

sreenath@ubuntuvm:~$
```

`dig www.example.com`

On inspecting the output of the `dig` command, we can see that the ANSWER section contains the DNS mapping of our query which is `10.0.2.101`, which corresponds to the IP that we had specified in the lookup files.



Wireshark interface showing a DNS capture. The packet list displays two DNS packets: a query (No. 3) and a response (No. 4). The packet details pane shows the structure of the DNS query, including the Linux cooked capture, IP header, UDP header, and the DNS query itself. The packet bytes pane shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
3	2.200548533	10.0.2.4	10.0.2.15	DNS	100	Standard query 0xc40f A www.example.com OPT
4	2.200868830	10.0.2.15	10.0.2.4	DNS	137	Standard query response 0xc40f A www.example.com A 1...

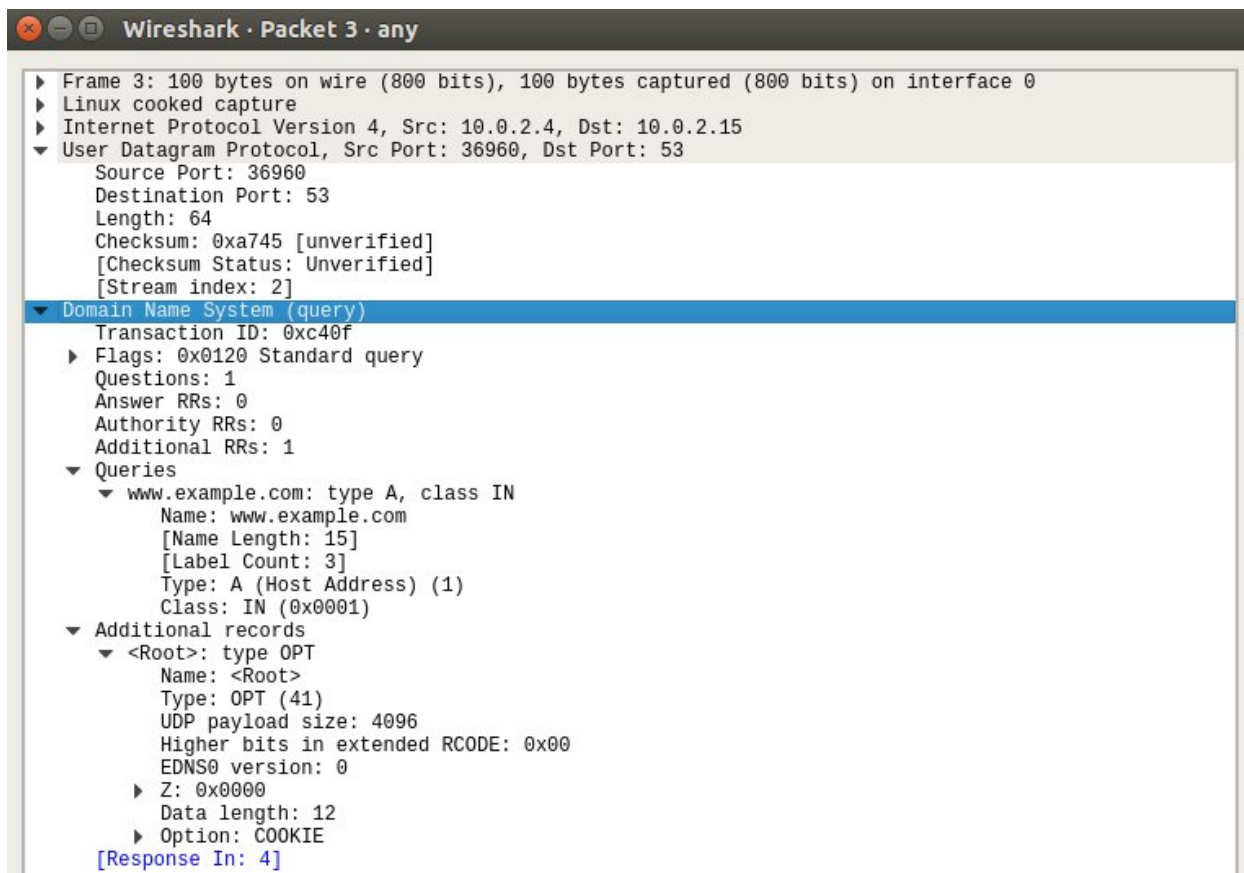
Frame 3: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0  
 Linux cooked capture  
 Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15  
 User Datagram Protocol, Src Port: 36960, Dst Port: 53  
 Domain Name System (query)

```

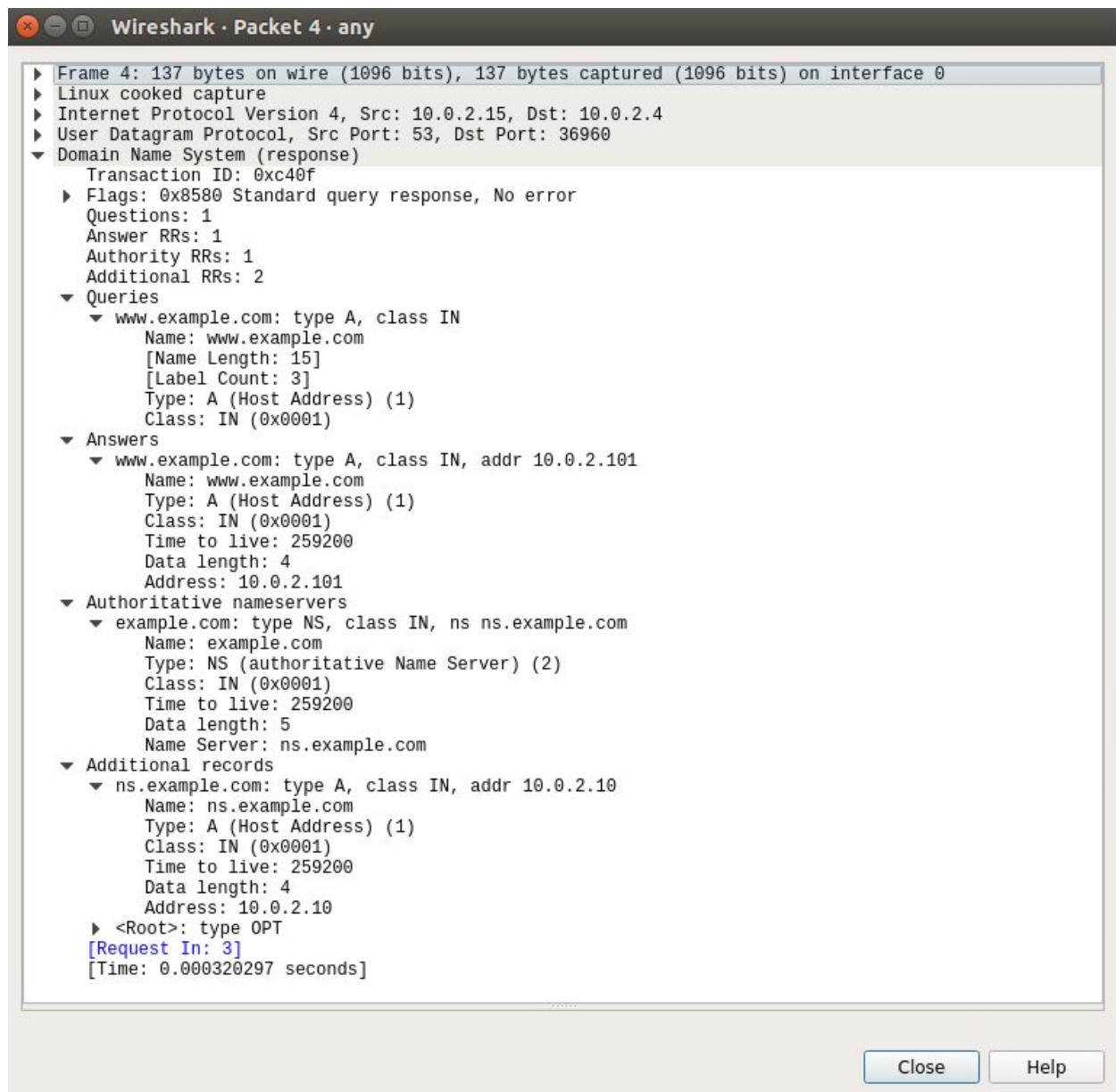
0000  00 00 00 01 00 06 08 00 27 34 e3 3f 00 00 08 00  ..... '4.?....
0010  45 00 00 54 2f b0 00 00 40 11 32 d7 0a 00 02 04  E..T/... @.2....
0020  0a 00 02 0f 90 60 00 35 00 40 a7 45 c4 0f 01 20  .....5 @.E....
0030  00 01 00 00 00 00 00 01 03 77 77 77 07 65 78 61  .....www.exa
0040  6d 70 6c 65 03 63 6f 6d 00 00 01 00 01 00 00 29  mple.com .....)
0050  10 00 00 00 00 00 00 0c 00 0a 00 08 98 58 dd 55  .....X.U
0060  49 72 d1 8b                                Ir..
  
```

Domain Name System: Protocol      Packets: 8 · Displayed: 2 (25.0%) · Dropped: 0 (0.0%)      Profile: Default

## Wireshark Capture



## DNS Query



## DNS Query Response

Unfortunately, on checking the DNS cache after this task, there was only garbage information, so it has been omitted.

## Questions:

1. Locate the DNS query and response messages. Are they sent over TCP or UDP?

We notice from the above packet captures that they are sent over UDP.

2. What is the destination port for the DNS query message? What is the source port of the DNS response message?

Both ports are the same. The port used for DNS is 53.

3. To what IP address is the DNS query message sent? Using ipconfig, determine the IP address of your local DNS server. Are these 2 IP's the same?

The DNS query is sent to 10.0.2.15 which is what is configured.

4. Examine the DNS query message. What 'Type' of DNS query is it? Does the query message contain any 'answers'?

The DNS query is of type A since it is authoritative. There is no answer section since queries won't have answers.

5. Examine the DNS response message. How many 'answers' are provided? What do each of these answers contain?

The DNS response contained 1 RR: an A type RR for `www.google.com` with its IP address and canonical hostname.

6. Consider the subsequent TCP SYN packet sent by your host. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?

The destination IP address corresponds to the IP address of the hostname `www.google.com` obtained from the response message.

