**Internet Domain Name System**

Lab 6

*50.005 Computer System Engineering*

***Due: 15 Apr 08:30 AM (Week 12)***

[Overview and Learning Objectives](#_by3079ftoyhp)

[Submission](#_s8bsd4izj6cd)

[Part 1: Exploring DNS using dig](#_thq99knj2wzk)

[Part 2: DNS Hierarchy](#_con9u122eg2x)

[Part 3: DNS Caching](#_by60ahx6jhz8)

[Part 4: Tracing DNS using Wireshark](#_o7pxuf8heo9)

# Overview and Learning Objectives

In NS Module 4, we learnt about the role of the Domain Name System (DNS) in Internet naming and addressing. In this lab exercise, we will go deeper into DNS by using specialised network tools to perform and analyse DNS queries.

At the end of this lab exercise, you should be able to:

● Use dig to perform DNS queries (e.g. to look up an IP address)

● Read and interpret DNS records of different types

● Understand how a DNS query is resolved using hierarchy and recursion

● Observe and understand the effect of caching on DNS lookup times

● Use Wireshark to trace and read DNS packets sent to and from a machine

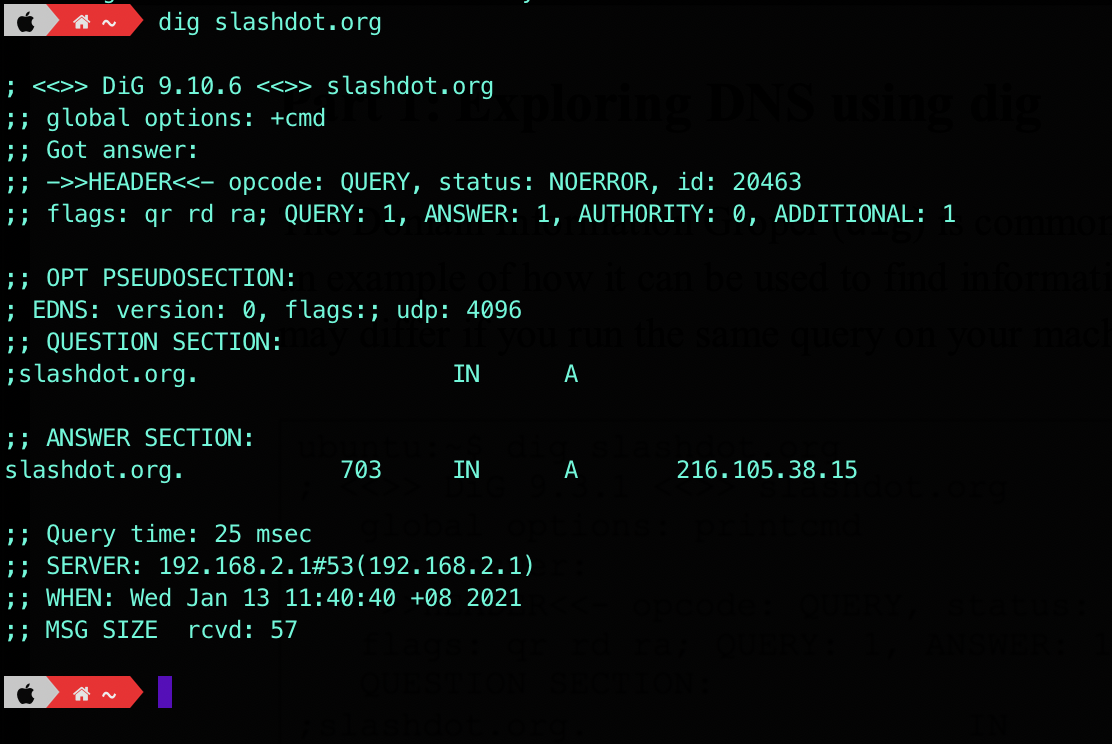
If you are using Ubuntu, dig should already be available on your system. To install Wireshark, run sudo apt-get install wireshark from the command line.

# Submission

* The total marks for this Lab is 25 (Part 1: 10, Part 2: 5, Part 3: 10)
* Complete the activities and answer the questions in the handout **denoted in blue.** As usual, export and edit this document.
* Export as pdf and **ZIP** it (not rar, or any other compression algorithm)
* **Upload** to @csesubmitbot telegram bot using the command /submitlab6
* **CHECK** your submission by using the command /checksubmission

# Part 1: Exploring DNS using dig

The Domain Information Groper (dig) is commonly used for performing DNS lookups. Here is an example of how it can be used to find information about the host slashdot.org. The results may differ if you run the same query on your machine.



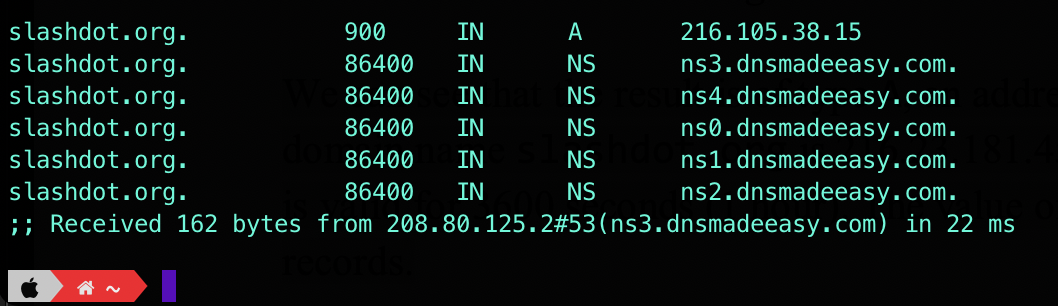
When the command dig slashdot.org is run, dig performs a DNS lookup and displays information about the request and the response it receives. At the bottom of the printout, we can see that the query was sent to the DNS server running on 192.168.2.1, and that the query took 25 ms to complete. Most of the information that we are interested in can be found in the **ANSWER SECTION**.

The answer section for this query contains a DNS record:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| slashdot.org | 703 | IN | A | 216.105.38.15 |
| *Server Name* | *Expiry* | *Class* | *Type* | *Data* |

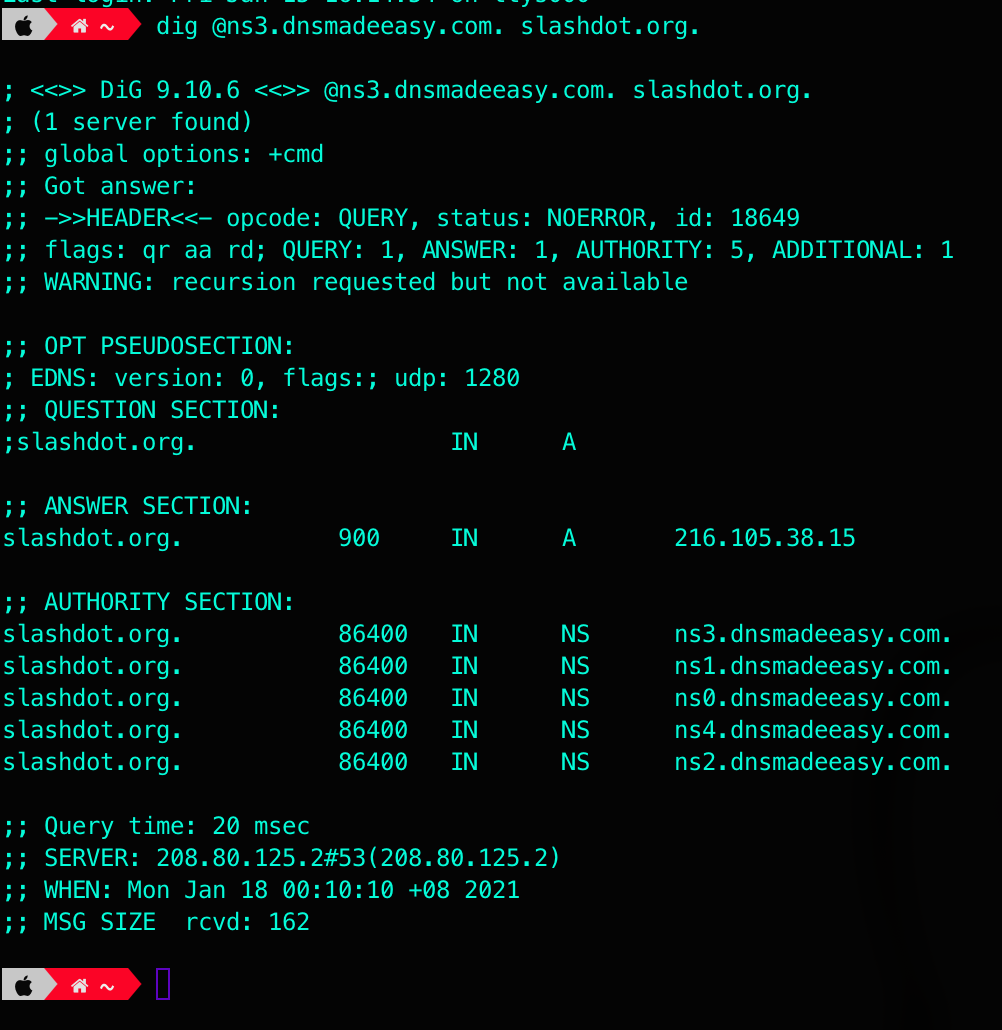
We can see that the result is of type A, an address record. It tells us that the IP address for the domain name slashdot.org is 216.105.38.15. The expiry time field indicates that this record is valid for 703 seconds. The value of the class field is usually **IN** (Internet) for all records.

If you’d like to know who’s the authoritative NS for the queried domain, you can add the **trace** option: dig slashdot.org +trace



The records of type NS indicate the names of the DNS servers storing records for a particular domain. Here, we can see that the hosts ns.3.dnsmadeeasy.com. and etc are responsible for providing authoritative responses to names in the slashdot.org domain.

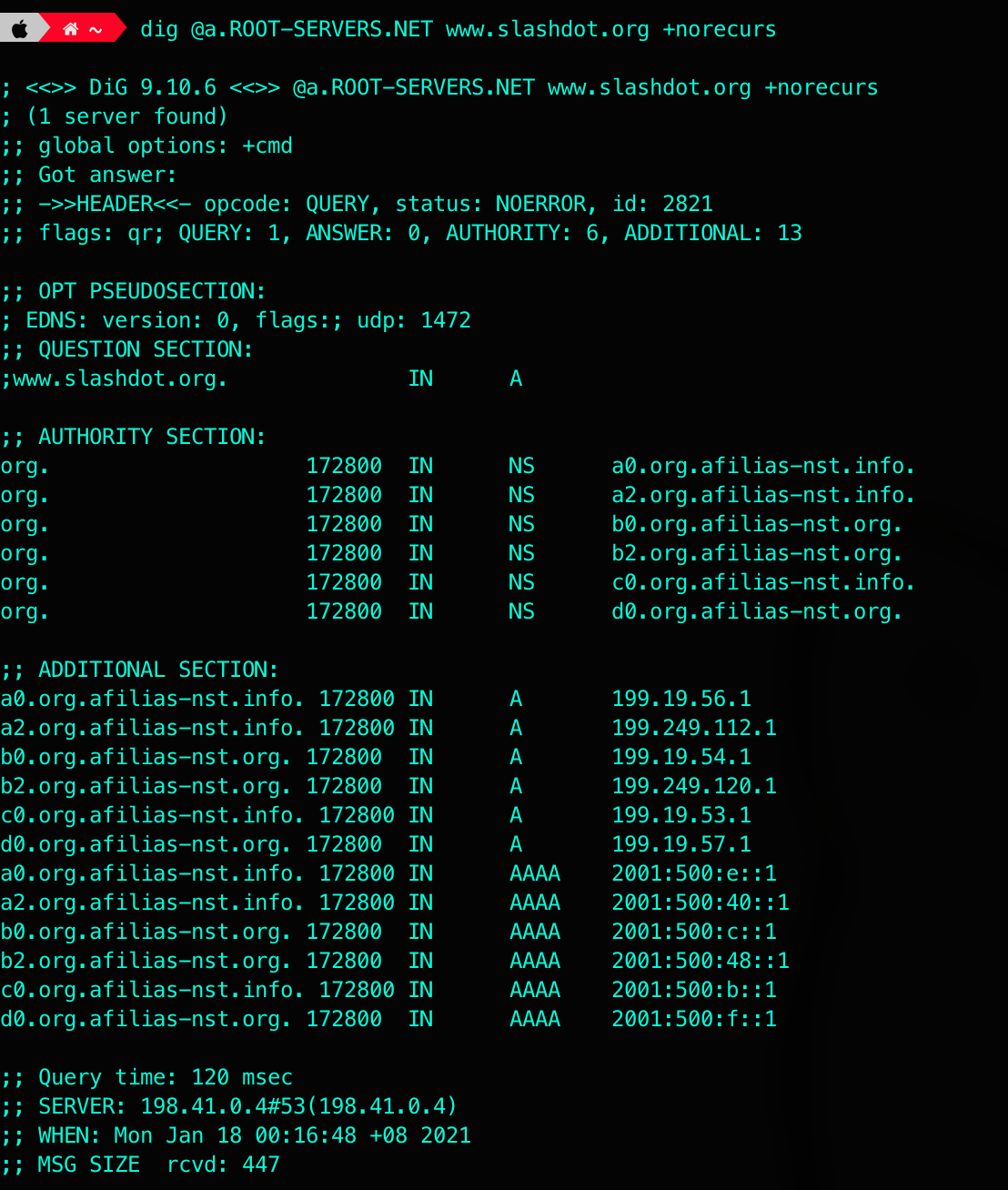
We can query a specific server for information about a host by using the @ option. For example, to perform a lookup using the DNS server ns3.dnsmadeeasy.com., we can run the command dig @ns3.dnsmadeeasy.com. slashdot.org.



There are three flags under the header: qr, aa, and rd.

* This means that the message is a query (qr), and dig is requesting a recursive lookup (rd stands for *‘recursion desired’*) and the server is the authoritative name server (aa stands for *‘authoritative answer’*).
* Not all servers perform recursive lookups due to the heavier load involved, and so you don’t see any ra flags here (ra stands for ‘*recursion available*’).

dig only prints the final result of a recursive search, but you can mimic the individual steps involved by making a query with the +norecurs option enabled. For example, to send a non- recursive query to one of the root servers:



As you can see, the server does not know the answer (there’s 0 ANSWER) and instead provides information about the servers most likely to be able to provide an authoritative answer for the question. In this case, the best that the root server knows is the identities of the servers for the org. top-level domain.

Now answer the questions below.

**Question 1 [1p]:** Using dig, find the IP address for thyme.lcs.mit.edu. What is the IP address?

**Your answer:**

**The IP address is 18.26.0.122**

**Question 2 [1p]:** The dig answer for the previous question includes a record of type CNAME. What does CNAME mean?

**Your answer:**

**CNAME refers to a canonical name, which is just another alias for the hostname.**

**Question 3 [1p]:** What is the expiration time for the CNAME record?

**Your answer:**

**The expiration time for the CNAME record is 1800s.**

**Question 4:** Run the following commands to find out what your computer receives when it looks up *‘ai’* and *‘ai.’* in the mit.edu domain. What are the two resulting IP addresses?

● dig +domain=mit.edu ai

**Your answer [1p]:**

**There is no IP address found.**

● dig +domain=mit.edu ai.

**Your answer [1p]:**

**The resulting IP address is 209.59.119.34.**

**Question 5 [1p]:** Why are the results for both queries different? Look up the manual for dig to find out what the +domain parameter does. Based on the output of the two commands, what is the difference between the DNS searches being performed for *‘ai’* and *‘ai.’*?

**Your answer:**

**The +domain parameter is used to set the search list to contain the domain. The reason for the difference between the DNS searches is due to the trailing dot in ‘ai’. With the trailing dot (‘ai.’), it’s an absolute path and will search this absolute path. However, without the trailing dot, it will search for ai within the mit.edu domain and in this case, there is no result of ai in the domain.**

# Part 2: DNS Hierarchy

In the previous section, you ran dig without changing the default options. This causes dig to perform a recursive lookup if the DNS server being queried supports it. In this part, you will trace the intermediate steps involved in a performing recursive query by beginning at a root server and manually going through the DNS hierarchy to resolve a host name. You can obtain a list of all the root servers by running the command dig . NS.

**Question 6 [1p]:** Use dig to query one of the DNS root servers for the IP address of lirone.csail.mit.edu without using recursion. What is the command that you use to do this?

**Your answer:**

**dig @m.root-servers.net. lirone.csail.mit.edu +norecurs**

**Question 7 [3p]:** Go through the DNS hierarchy from the root until you have found the IP address of lirone.csail.mit.edu. You should disable recursion and follow the referrals manually. Which commands did you use, and what addresses did you find? You can provide screenshots for each step.

**Your answer:**

1. dig @m.root-servers.net. lirone.csail.mit.edu +norecurs
2. dig @a.edu-servers.net. lirone.csail.mit.edu +norecurs
3. dig @asia1.akam.net. lirone.csail.mit.edu +norecurs
4. dig @auth-ns0.csail.mit.edu. lirone.csail.mit.edu +norecurs

**IP address: 128.52.129.186**

# Part 3: DNS Caching

**Question 8:** Without using recursion, query your default (local) DNS server for information about www.dmoz.org and answer the following questions.

* ● **[1p]** What is the command that you used?
* ● **[1p]** Did your default server have the answer in its cache? How did you know?
* ● **[1p]** How long did the query take?

*If the information was cached, find another host name that was not cached and complete all the questions in this section using that host.*

*You can set your default DNS server as your router (your ISP might do that too). Why can your router support DNS (layer 5, application layer protocol) when “routers” are supposed to support up to Network layer only?*

**Your answer:**

**The command I used was dig www.dmoz.org +norecurs.**

**My default server did not have the answer in its cache. This is seen in my terminal: ;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1 and as you can see ANSWER: 0.**

**The query took 17msec.**

**Question 9 [1p]:** Query your default DNS server for information about the host in the previous question, using the recursion option this time. How long did the query take?  
**Your answer:**

**With the recursion option, the query took 153msec.**

**Question 10 [1p]:** Query your default DNS server for information about the same host without using recursion. How long did the query take? Has the cache served its purpose? Explain why.

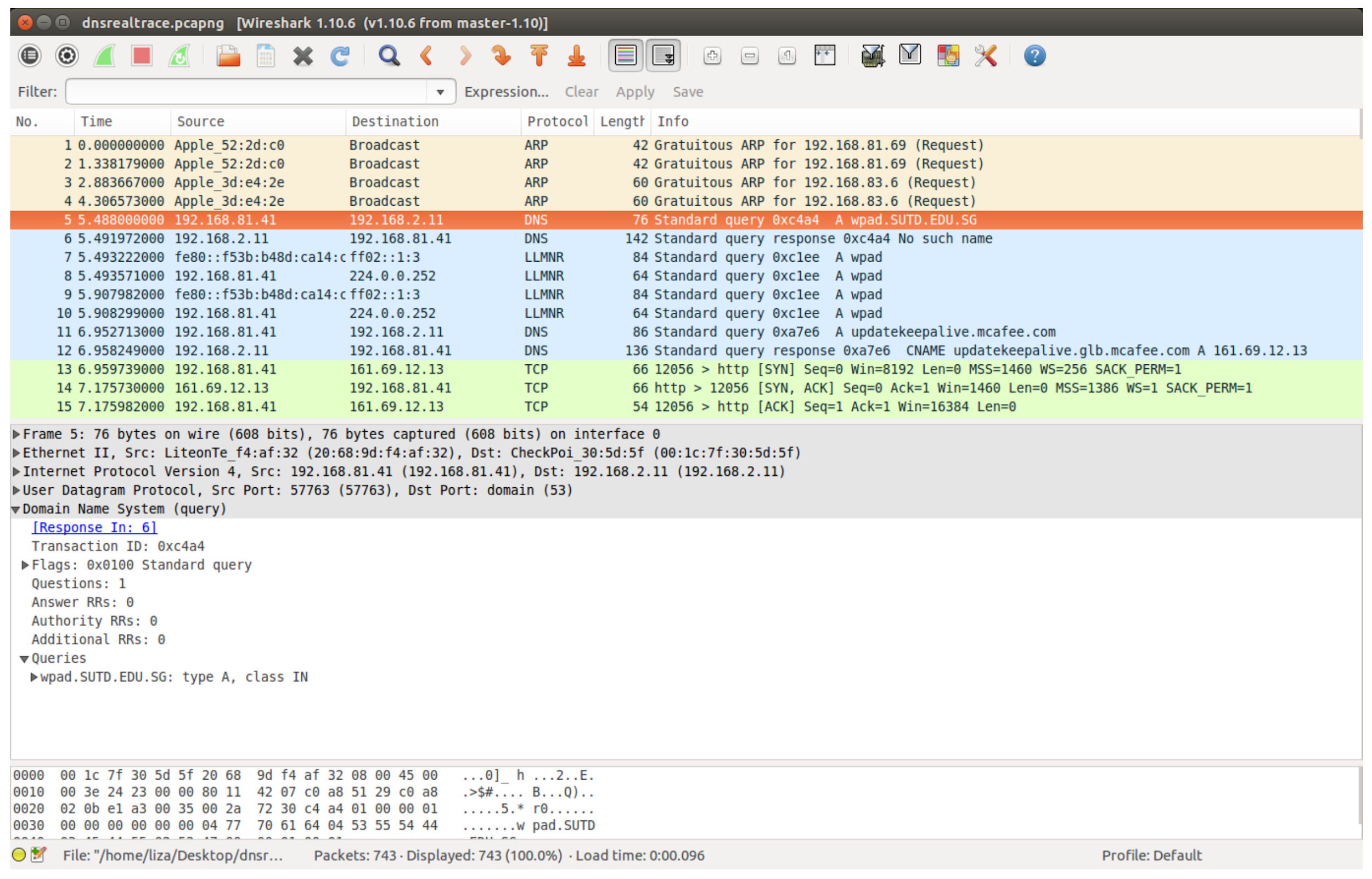
**Your answer:**

**The query took 11msec. Yes, the cache served its purpose as the DNS RR was cached in my default local DNS server, resulting in a faster query time.**

# Part 4: Tracing DNS using Wireshark

**Wireshark is a powerful tool used to capture packets sent over a network and analyse the content of the packets retrieved**. The file [*dnsrealtrace.pcapng*](https://drive.google.com/file/d/118Z03KnN7mNchsIs3G-DUdtf1zJV3NVI/view?usp=sharing)contains a trace of the packets sent and received when a web page is downloaded from a web server over the SUTD network. In the process of downloading the web page, DNS is used to find the IP address of the server.

Open the *dnsrealtrace.pcapng* in Wireshark and answer the following questions. You can refer to Wireshark tutorial [here](https://drive.google.com/file/d/12zi50lKYTf6ebXQNbUJsstc_BBSWO6X6/view?usp=sharing) before proceeding.



**Question 11 [1p]:** Locate the DNS query and response messages. Are they sent over UDP or TCP?

**Your answer:**

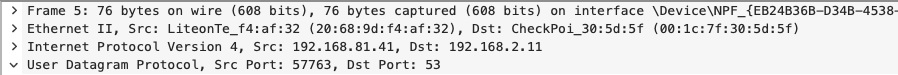
**They are sent over UDP.**

**Question 12 [2p]:** What is the destination port for the DNS query message? What is the source port of the DNS response message?

**Your answer:**

**The destination port is 53.**

**The source port is 57763.**



**Question 13 [2p]:** What is the IP address to which the DNS query message was sent? Use ifconfig to determine the IPv4 address of your local DNS server. Are these two addresses the same?

**Your answer:**

**The DNS query message was sent to 192.168.2.11. The two addresses are not the same when compared with my local DNS server.**

**Question 14 [2p]:** Examine the second DNS query message. What type of DNS query is it? Does the query message contain any answers?

**Your answer:**

**It is a standard query response. The query message does not contain any answers.**

**Question 15 [2p]:** Examine the second DNS response message. How many answers are provided? What does each of these answers contain?

**Your answer:**

**There are 2 answers. The answers contain the Name, Type, Class, Time, Data length and CNAME.**

**Graphical user interface, text, application, email

Description automatically generated**

**Question 16 [1p]:** Locate a TCP SYN packet sent by your host subsequent to the above DNS response. This packet opens a TCP connection between your host and the web server. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?

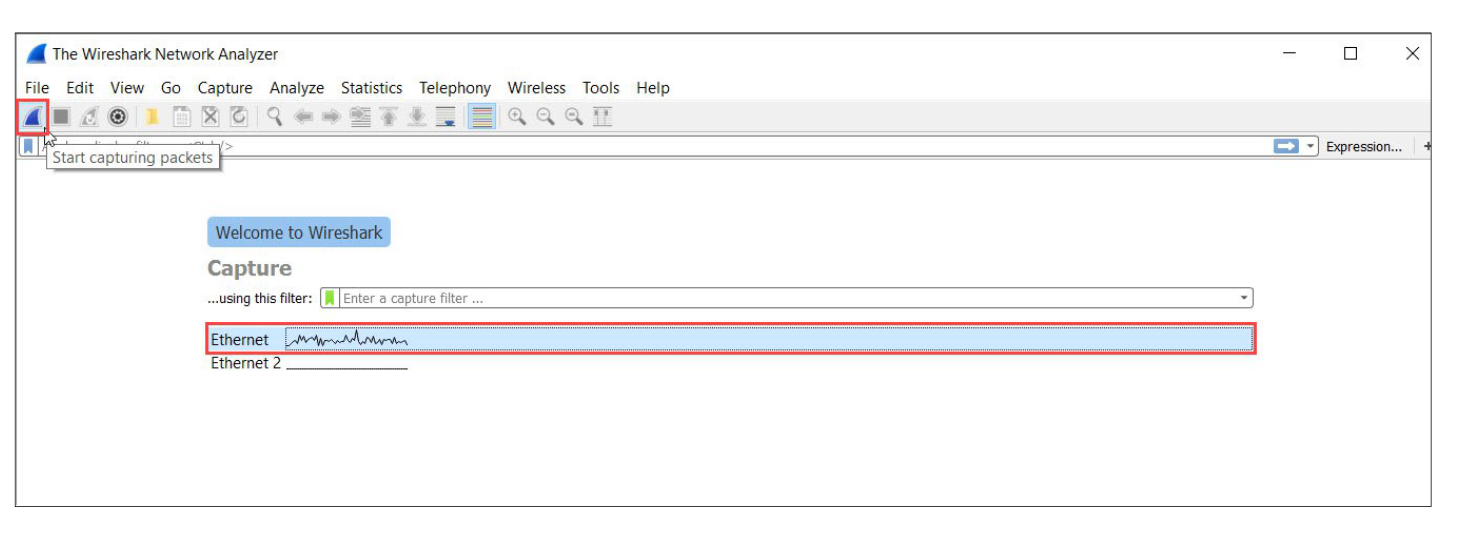
**Your answer:**

**Yes, the destination IP address is 161.69.12.13 which is found in the RR from the DNS response message.**

**Optional Activity:**

Capturing packets for packet analysis: Steps:

1. Once Wireshark is installed, launch the program to begin.
2. Once the program is launched, select the network interface to capture and click on the *sharkfin* icon at the top left of the application right under the menu bar to begin capturing packets.



1. To explore the interface, mention the interface (e.g. eth0, wlan) in the capture option.
2. There are display filters to analyse the packets.
   * Protocols: TCP, UDP, ARP, SMTP, etc.
   * Protocol Fields: port, src.addr, length, etc. (E.g. ip.src == 192.168.1.1)
3. For more detailed instructions on Wireshark, refer to https://www.wireshark.org/