50.005 Computer System Engineering (Spring 2020)

Lab 6: Internet Domain Name System

Overview

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.

Learning objectives

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

Preparation

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.

Deliverables

- Complete the activities and answer the questions in the handout. Submit a report containing your name, student ID and answers to eDimension before **16th April, 2020 9 AM** . Late submissions will be dealt with according to the standard course policy.

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.

```
ubuntu:~$ dig slashdot.org
; <<>> DiG 9.3.1 <<>> slashdot.org
   global options: printcmd
   Got answer:
   ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 997
   flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 3
   QUESTION SECTION:
;slashdot.org.                  IN      A
;; ANSWER SECTION:
slashdot.org.           3600    IN      A       216.34.181.45           (*)
;; AUTHORITY SECTION:
slashdot.org.          86399   IN      NS      ns-2.ch3.sourceforge.com.
slashdot.org.          86399   IN      NS      ns-1.ch3.sourceforge.com.
slashdot.org.          86399   IN      NS      ns-1.sourceforge.com.

;; ADDITIONAL SECTION:
ns-1.ch3.sourceforge.com. 172800 IN    A       216.34.181.21
ns-1.sourceforge.com.  172800  IN      A       208.122.22.23
ns-2.ch3.sourceforge.com. 172800 IN    A       216.34.181.22
   Query time: 69 msec
   SERVER: 127.0.0.1#53(127.0.0.1)
   WHEN: Wed Mar 11 17:32:51 2009
   MSG SIZE rcvd: 170
```

Figure 1: Output from dig

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 on `127.0.0.1`, and that the query took 69 ms to complete. Most of the information that we are interested in can be found in the answer section, marked with a bold asterisk **(*)** above.

The answer section for this query contains a DNS record.

`slashdot.org.`	`3600`	`IN`	`A`	`216.34.181.45`
server name	*expiry*	*class*	*type*	*data*

Figure 2: An 'A' record with fields annotated

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 216.23.181.45. The expiry time field indicates that this record is valid for 3600 seconds (1 hour). The value of the class field is usually `IN` (Internet) for all records.

The authority section contains records of type `NS`, indicating the names of the DNS servers storing records for a particular domain. Here, we can see that the hosts `ns-2.ch3.sourceforge.com.`, `ns-1.ch3.sourceforge.com.` and `ns-1.sourceforge.com.` 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 `dns1.maxias.net`, we can run the command `dig @dns1.maxias.net. slashdot.org`.

```
Dimas-MacBook-Pro:~ damdoom$ dig @Dns1.maxias.net. slashdot.org

; <<>> DiG 9.8.3-P1 <<>> @Dns1.maxias.net. slashdot.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 14259
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;slashdot.org.                  IN      A

;; ANSWER SECTION:
slashdot.org.           300     IN      A       216.34.181.45

;; Query time: 363 msec
;; SERVER: 123.100.236.66#53(123.100.236.66)
;; WHEN: Mon Apr  4 20:52:56 2016
;; MSG SIZE  rcvd: 46
```

Figure 3: Output from dig using the @ option

The header section indicates that the `rd` and `ra` flags have been enabled for the query. This means that `dig` is requesting a recursive lookup (`rd` stands for *'recursion desired'*) and the server performs recursive lookups (`ra` stands for *'recursion available'*). Not all servers perform recursive lookups due to the heavier load involved.

`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:

```
Dimas-MacBook-Pro:~ damdoom$ dig @a.ROOT-SERVERS.NET www.slashdot.org +norecurs

; <<>> DiG 9.8.3-P1 <<>> @a.ROOT-SERVERS.NET www.slashdot.org +norecurs
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17663
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 12

;; QUESTION SECTION:
;www.slashdot.org.              IN      A

;; AUTHORITY SECTION:
org.                    172800  IN      NS      a0.org.afilias-nst.info.
org.                    172800  IN      NS      a2.org.afilias-nst.info.
org.                    172800  IN      NS      b0.org.afilias-nst.org.
org.                    172800  IN      NS      b2.org.afilias-nst.org.
org.                    172800  IN      NS      c0.org.afilias-nst.info.
org.                    172800  IN      NS      d0.org.afilias-nst.org.

;; ADDITIONAL SECTION:
a0.org.afilias-nst.info. 172800 IN      A       199.19.56.1
a2.org.afilias-nst.info. 172800 IN      A       199.249.112.1
b0.org.afilias-nst.org. 172800  IN      A       199.19.54.1
b2.org.afilias-nst.org. 172800  IN      A       199.249.120.1
c0.org.afilias-nst.info. 172800 IN      A       199.19.53.1
d0.org.afilias-nst.org. 172800  IN      A       199.19.57.1
a0.org.afilias-nst.info. 172800 IN      AAAA    2001:500:e::1
a2.org.afilias-nst.info. 172800 IN      AAAA    2001:500:40::1
b0.org.afilias-nst.org. 172800  IN      AAAA    2001:500:c::1
b2.org.afilias-nst.org. 172800  IN      AAAA    2001:500:48::1
c0.org.afilias-nst.info. 172800 IN      AAAA    2001:500:b::1
d0.org.afilias-nst.org. 172800  IN      AAAA    2001:500:f::1

;; Query time: 52 msec
;; SERVER: 198.41.0.4#53(198.41.0.4)
;; WHEN: Mon Apr  4 21:03:21 2016
;; MSG SIZE  rcvd: 436
```

Figure 4: Output from dig using the +norecurs option

As you can see, the server does not know the 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.

DNS basics

Question 1: Using dig, find the IP address for `thyme.lcs.mit.edu`. What is the IP address?

Question 2: The dig answer for the previous question includes a record of type `CNAME`. What does `CNAME` mean?

Question 3: What is the expiration time for the `CNAME` record?

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`
- `dig +domain=mit.edu ai.`

Question 5: 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.'*?

Understanding 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.

Now, 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: 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?

Question 7: 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 address did you find?

Understanding caching

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

- What is the command that you used?
- Did your default server have the answer in its cache? How did you know?
- How long did the query take?

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

Question 9: 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?

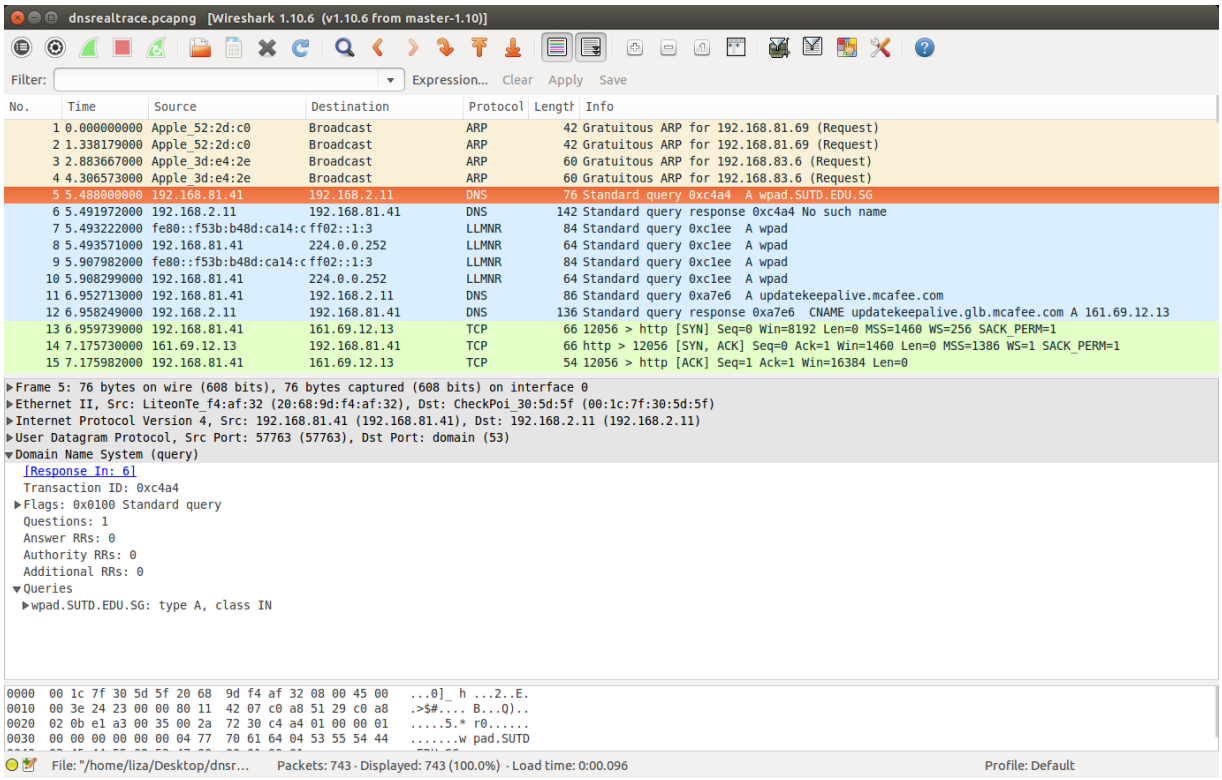
Question 10: Query your default DNS server for information about the same host without using recursion. How long did the query take? Has the cache served its purpose? Explain why.

Part 2: 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* 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.



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

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

Question 3: What is the IP address to which the DNS query message was sent? Use ifconfig to determine the IP address of your local DNS server. Are these two addresses the same?

Question 4: Examine the second DNS query message. What type of DNS query is it? Does the query message contain any answers?

Question 5: Examine the second DNS response message. How many answers are provided? What does each of these answers contain?

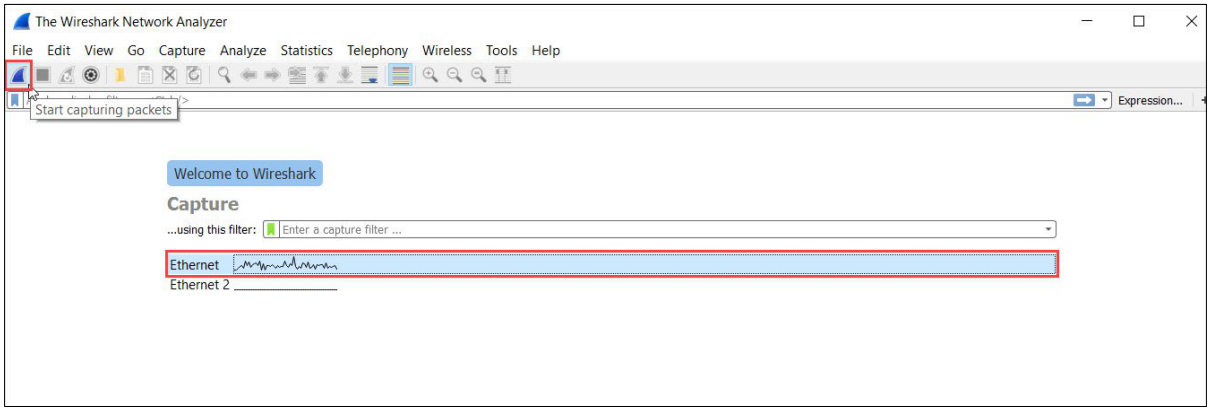
Question 6: Locate a TCP SYN packet sent by your host subsequent to the above DNS response. This packet opens a TCP connection between your host and the web server. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?

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* at the top left of the application right under the menu bar to begin capturing packets.



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