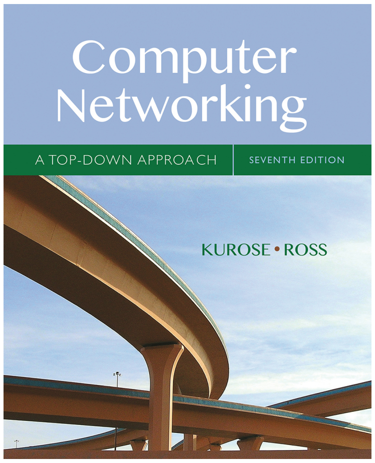
Wireshark Lab: DNS v7.2

Supplement to *Computer Networking: A Top-Down Approach, 7th ed.,* J.F. Kurose and K.W. Ross

"Tell me and I forget. Show me and I remember. Involve me and I understand." Chinese proverb



As described in Section 2.4 of the text[1], the Domain Name System (DNS) translates hostnames to IP addresses, fulfilling a critical role in the Internet infrastructure. In this lab, we'll take a closer look at the client side of DNS. Recall that the client's role in the DNS is relatively simple – a client sends a *query* to its local DNS server, and receives a *response* back. As shown in Figures 2.19 and 2.20 in the textbook, much can go on "under the covers," invisible to the DNS clients, as the hierarchical DNS servers communicate with each other to either recursively or iteratively resolve the client's DNS query. From the DNS client's standpoint, however, the protocol is quite simple – a query is formulated to the local DNS server and a response is received from that server.

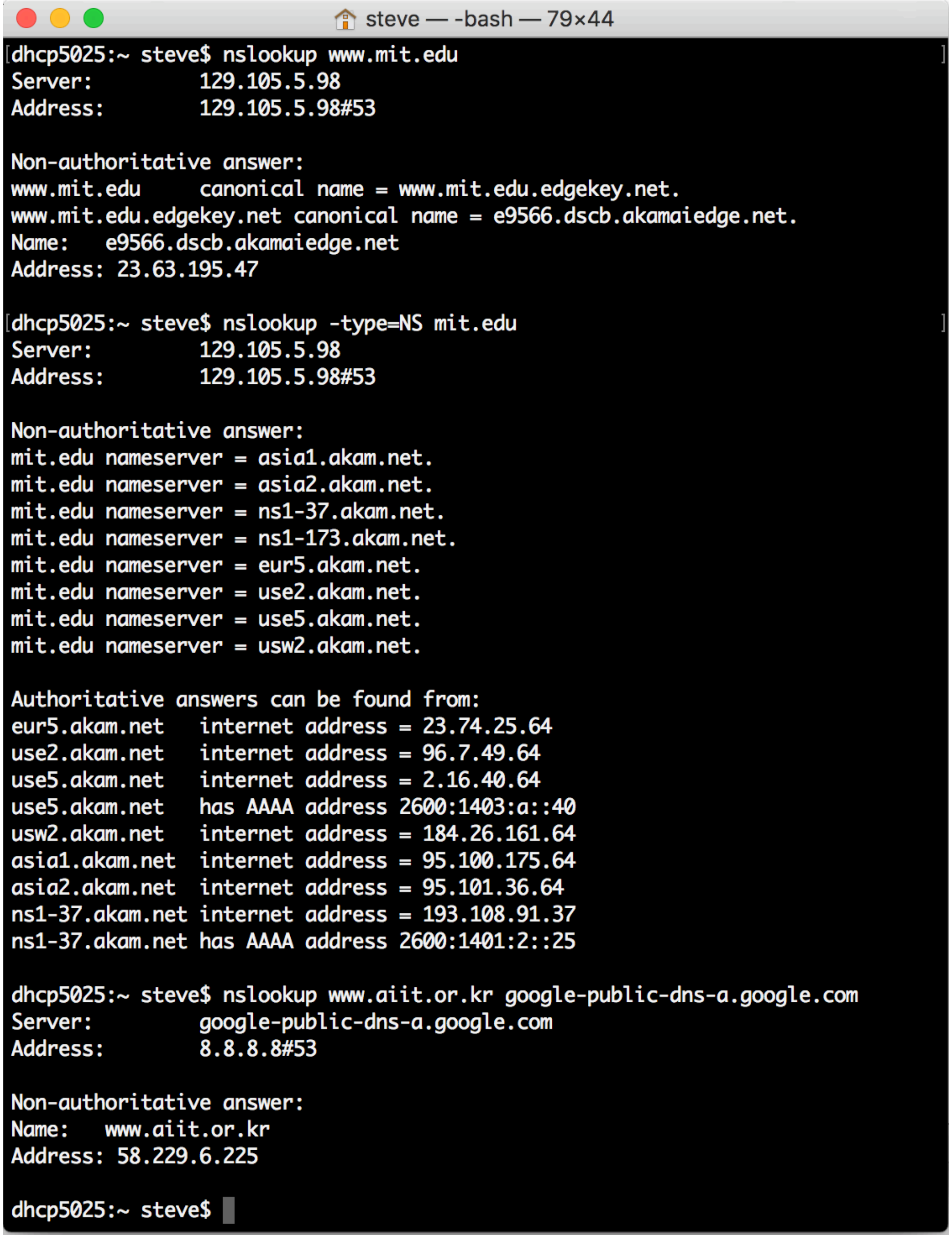
Before beginning this lab, you'll probably want to review DNS by reading Section 2.4 of the text. In particular, you may want to review the material on **local DNS servers**, **DNS caching**, **DNS records and messages**, and the **TYPE field** in the DNS record.

1. nslookup

In this lab, we'll make extensive use of the *nslookup* tool, which is available in most Linux/Unix and Microsoft platforms today. To run *nslookup* in Linux/Unix, you just type the *nslookup* command on the command line. To run it in Windows, open the Command Prompt and run *nslookup* on the command line.

In it is most basic operation, *nslookup* tool allows the host running the tool to query any specified DNS server for a DNS record. The queried DNS server can be a root DNS server, a top-level-domain DNS server, an authoritative DNS server, or an intermediate DNS server (see the textbook for definitions of these terms). To accomplish this task, *nslookup* sends a DNS query to the specified DNS server, receives a DNS reply from that same DNS server, and displays the result.

[1] References to figures and sections are for the 7th edition of our text, *Computer Networks, A Top-down Approach, 7th ed.,* J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2016.



```
[dhcp5025:~ steve$ nslookup www.mit.edu
Server:         129.105.5.98
Address:        129.105.5.98#53

Non-authoritative answer:
www.mit.edu     canonical name = www.mit.edu.edgekey.net.
www.mit.edu.edgekey.net canonical name = e9566.dscb.akamaiedge.net.
Name:   e9566.dscb.akamaiedge.net
Address: 23.63.195.47

[dhcp5025:~ steve$ nslookup -type=NS mit.edu
Server:         129.105.5.98
Address:        129.105.5.98#53

Non-authoritative answer:
mit.edu nameserver = asia1.akam.net.
mit.edu nameserver = asia2.akam.net.
mit.edu nameserver = ns1-37.akam.net.
mit.edu nameserver = ns1-173.akam.net.
mit.edu nameserver = eur5.akam.net.
mit.edu nameserver = use2.akam.net.
mit.edu nameserver = use5.akam.net.
mit.edu nameserver = usw2.akam.net.

Authoritative answers can be found from:
eur5.akam.net   internet address = 23.74.25.64
use2.akam.net   internet address = 96.7.49.64
use5.akam.net   internet address = 2.16.40.64
use5.akam.net   has AAAA address 2600:1403:a::40
usw2.akam.net   internet address = 184.26.161.64
asia1.akam.net  internet address = 95.100.175.64
asia2.akam.net  internet address = 95.101.36.64
ns1-37.akam.net internet address = 193.108.91.37
ns1-37.akam.net has AAAA address 2600:1401:2::25

dhcp5025:~ steve$ nslookup www.aiit.or.kr google-public-dns-a.google.com
Server:         google-public-dns-a.google.com
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   www.aiit.or.kr
Address: 58.229.6.225

dhcp5025:~ steve$ 
```

The above screenshot shows the results of three independent *nslookup* commands (displayed in the Mac Terminal.app). In this example, the client host is located on the campus of Northwestern University, where the default local DNS server is 129.105.5.98.

When running *nslookup*, if no DNS server is specified, then *nslookup* sends the query to the default DNS server, which in this case is 129.105.5.98. Consider the first command:

```
nslookup www.mit.edu
```

In words, this command is saying "please send me the IP address for the host www.mit.edu". As shown in the screenshot, the response from this command provides two pieces of information: (1) the name and IP address of the DNS server that provides the answer; and (2) the answer itself, which is the host name and IP address of www.mit.edu. Although the response came from the local DNS server on campus, it is quite possible that this local DNS server iteratively contacted several other DNS servers to get the answer, as described in Section 2.4 of the textbook.

Now consider the second command:

```
nslookup -type=NS mit.edu
```

In this example, we have provided the option "-type=NS" and the domain "mit.edu". This causes *nslookup* to send a query for a type-NS record to the default local DNS server. In words, the query is saying, "please send me the host names of the authoritative DNS for mit.edu". (When the -type option is not used, *nslookup* uses the default, which is to query for type A records.) The answer, displayed in the above screenshot, first indicates the DNS server that is providing the answer (which is the default local DNS server) along with eight MIT nameservers. Each of these servers is indeed an authoritative DNS server for the hosts on the MIT campus. However, *nslookup* also indicates that the answer is "non-authoritative," meaning that this answer came from the cache of some server rather than from an authoritative MIT DNS server. Finally, the answer also includes the IP addresses of the nine authoritative DNS servers for MIT (which happen to be provided by the company Akamai – akam.net). Even though the type-NS query generated by *nslookup* did not explicitly ask for these IP addresses, the local DNS server returned these "for free" and *nslookup* displays the result.

Now finally consider the third command:

```
nslookup www.aiit.or.kr google-public-dns-a.google.com
```

In this example, we indicate that we want to the query sent to the DNS server "google-public-dns-a.google.com" rather than to the default DNS server (dns-prime.poly.edu). Thus, the query and reply transaction takes place directly between our querying host and google-public-dns-a.google.com. In this example, the Google DNS server provides the IP address of the host www.aiit.or.kr, which is a web server at the Advanced Institute of Information Technology (in Korea). Note that we could have supplied an IP address (instead of a hostname) for the DNS server:

```
nslookup www.aiit.or.kr 8.8.8.8
```

Now that we have gone through a few illustrative examples, you are perhaps wondering about the general syntax of *nslookup* commands. The syntax is:

```
nslookup -option1 -option2 host-to-find dns-server
```

In general, *nslookup* can be run with zero, one, two or more options. And as we have seen in the above examples, the dns-server is optional as well; if it is not supplied, the query is sent to the default local DNS server.

The basic syntax for *nslookup* is the same on all operating systems, but the advanced options generally have different syntax on different OSes.

Now that we have provided an overview of *nslookup*, it is time for you to test drive it yourself. Do the following (and write down the results):

1. Run *nslookup* to obtain the IP address of a Web server in Asia. What is the IP address of that server?
2. Run *nslookup* to determine the authoritative DNS servers for a university in Europe.
3. Let's say that we want to get the mail servers that accept email for Yahoo! Mail accounts. What nslookup command gives this result?
4. Modify the query above to send the same query for Yahoo! Mail servers to the European DNS server you gave as an answer in Question 2.

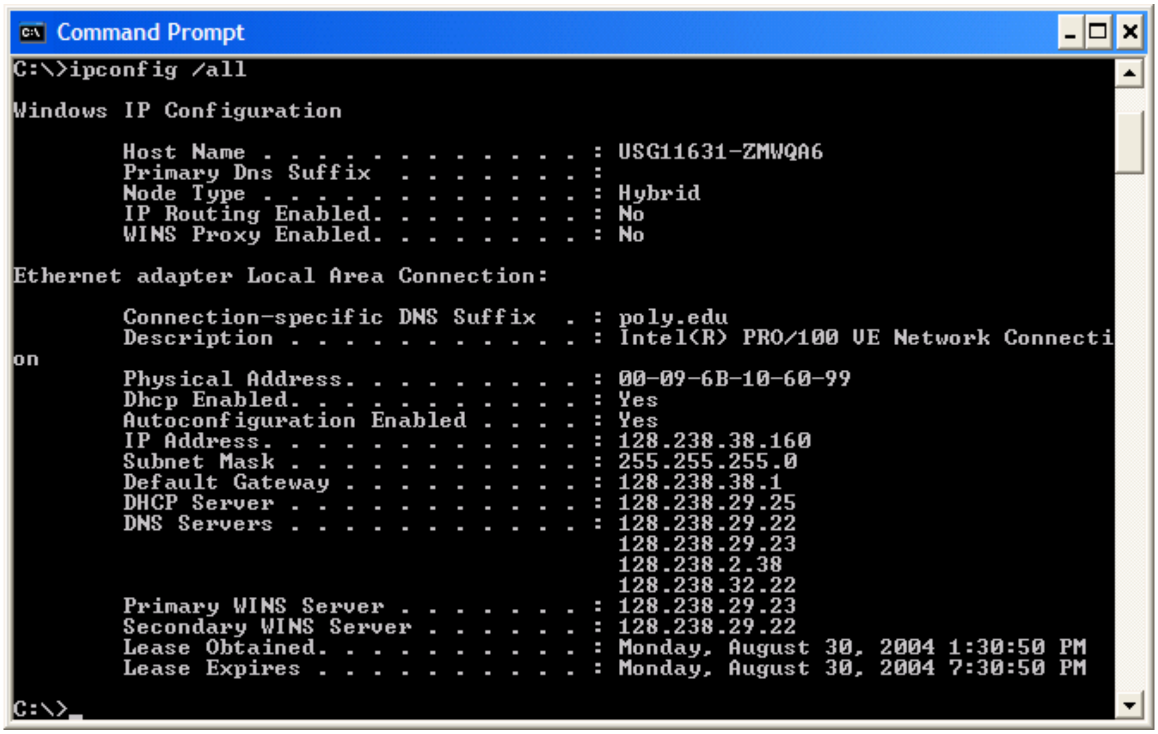
   NOTE: The query above will likely be *refused* by the server. Most DNS servers are configured to give responses only when the query comes from a certain range of client IP addresses or when asked about a domain for which it is an authoritative name server. The DNS server for the European university is not an authoritative namserver for yahoo.com, so it would have to make a query itself to answer your query. It refused your query because it does not want to do all this work for you. On the other hand, if the request came from an IP address within the university campus, then it would probably do the work necessary to answer it.

2. ipconfig

ipconfig (for Windows) and *ifconfig* (for Unix/Mac) are useful for debugging network issues. Here we'll only describe *ipconfig*, although the Unix/Mac *ifconfig* is very similar. *ipconfig* can be used to show your current TCP/IP information, including your address, DNS server addresses, adapter type and so on. The examples below apply to Windows, but there are similar commands available for ifconfig on other operating systems. For example, if you all this information about your host simply by entering

```
ipconfig \all
```

into the Command Prompt, as shown in the following screenshot.



```
C:\>ipconfig /all

Windows IP Configuration

        Host Name . . . . . . . . . . . . : USG11631-ZMWQA6
        Primary Dns Suffix  . . . . . . . :
        Node Type . . . . . . . . . . . . : Hybrid
        IP Routing Enabled. . . . . . . . : No
        WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : poly.edu
        Description . . . . . . . . . . . : Intel(R) PRO/100 VE Network Connecti
on
        Physical Address. . . . . . . . . : 00-09-6B-10-60-99
        Dhcp Enabled. . . . . . . . . . . : Yes
        Autoconfiguration Enabled . . . . : Yes
        IP Address. . . . . . . . . . . . : 128.238.38.160
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 128.238.38.1
        DHCP Server . . . . . . . . . . . : 128.238.29.25
        DNS Servers . . . . . . . . . . . : 128.238.29.22
                                            128.238.29.23
                                            128.238.2.38
                                            128.238.32.22
        Primary WINS Server . . . . . . . : 128.238.29.23
        Secondary WINS Server . . . . . . : 128.238.29.22
        Lease Obtained. . . . . . . . . . : Monday, August 30, 2004 1:30:50 PM
        Lease Expires . . . . . . . . . . : Monday, August 30, 2004 7:30:50 PM

C:\>_
```

ipconfig is also very useful for managing the DNS information stored in your host. In Section 2.5 we learned that a host can cache DNS records it recently obtained. To see these cached records, after the prompt C:\> provide the following command:

```
ipconfig /displaydns
```

Each entry shows the remaining Time to Live (TTL) in seconds. To clear the cache, enter

```
ipconfig /flushdns
```

Flushing the DNS cache clears all entries and reloads the entries from the hosts file.

Mac instructions

On MacOS 10.10.4 and higher, the following command clears the DNS cache:

```
sudo killall -HUP mDNSResponder
```

Mac users can hold down the “option” key and click the wifi signal strength icon at top-right to see lots of network information, including your current IP address: http://osxdaily.com/2011/06/15/get-detailed-wifi-info-from-the-menu-bar/

3. Tracing DNS with Wireshark

Now that we are familiar with *nslookup* and *ipconfig/ifconfig*, we're ready to get down to some serious business. Let's first capture the DNS packets that are generated by ordinary Web-surfing activity.

- Use *ipconfig* to empty the DNS cache in your host.
- Open your browser and open an "Incognito" or "Private" browser tab. This will prevent your browser from the local cache.
- Open Wireshark and enter "ip.addr == your_IP_address" into the filter, where you obtain your_IP_address with ipconfig. This filter removes all packets that neither originate nor are destined to your host.
    - If your IP address has colons (:) instead of periods (.) in it, then you are using IPv6, and you'll have to use a filter like "ipv6.addr == your_IPv6_address"
    - If you see no packets, try removing the filter. It's not essential for this lab.
- Start packet capture in Wireshark.
- With your browser, visit the Web page: http://www.ietf.org
- Stop packet capture.

If you are unable to run Wireshark on a live network connection, you can download a packet trace file that was captured while following the steps above on one of the author's computers[2]. Answer the following questions.

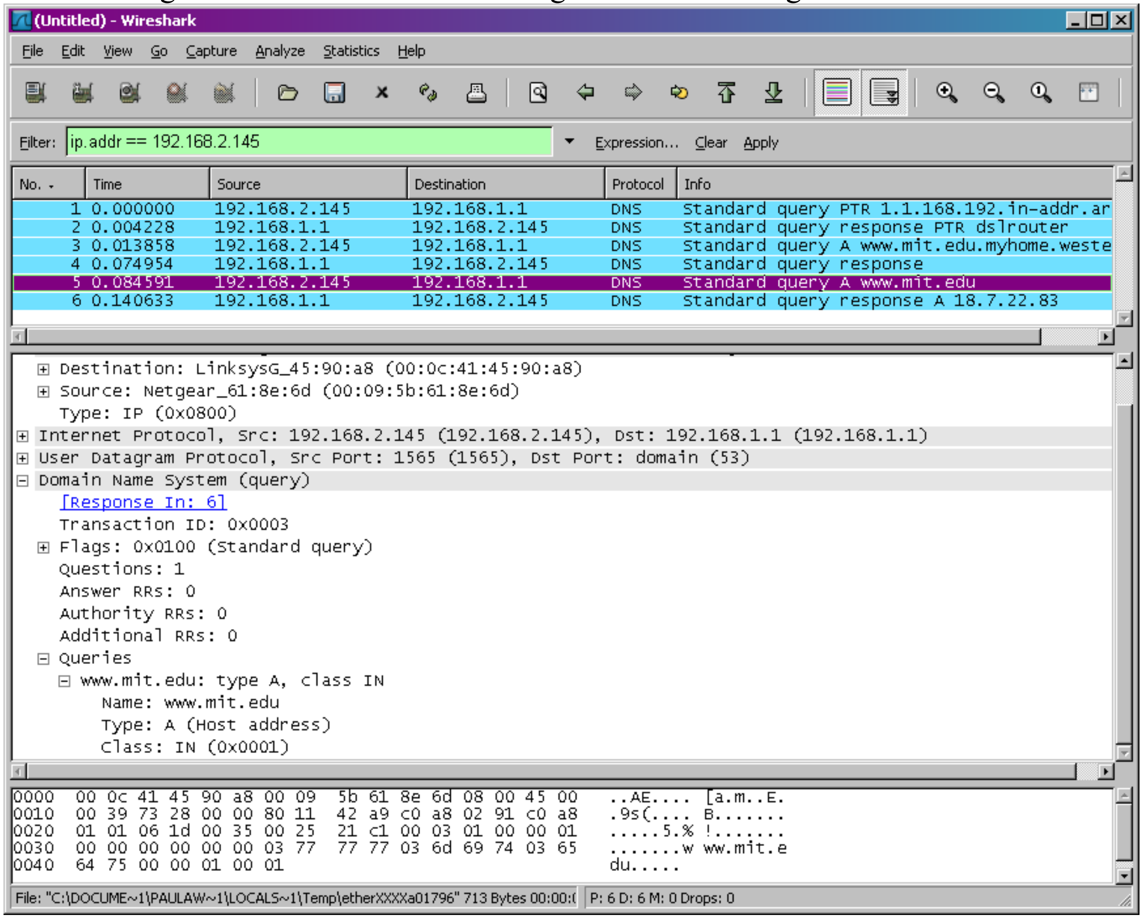
5. Locate the DNS query and response messages. Are then sent over UDP or TCP?
6. What is the destination port for the DNS query message? What is the source port of DNS response message?
7. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?
8. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
9. Examine the DNS response message. How many "answers" are provided? What do each of these answers contain?
10. 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?
11. This web page contains images. Before retrieving each image, does your host issue new DNS queries?

[2] Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zipand extract the file dns-ethereal-trace-1. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the dns-ethereal-trace-1 trace file.

Now let's play with *nslookup*[3].

- Start packet capture.
- Do an *nslookup* on www.mit.edu
- Stop packet capture.

You should get a trace that looks something like the following:


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Expression... Clear Apply

Filter: ip.addr == 192.168.2.145

```
No. -  Time      Source          Destination     Protocol  Info
     1 0.000000  192.168.2.145   192.168.1.1     DNS       Standard query PTR 1.1.168.192.in-addr.ar
     2 0.004228  192.168.1.1     192.168.2.145   DNS       Standard query response PTR dslrouter
     3 0.013858  192.168.2.145   192.168.1.1     DNS       Standard query A www.mit.edu.myhome.weste
     4 0.074954  192.168.1.1     192.168.2.145   DNS       Standard query response
     5 0.084591  192.168.2.145   192.168.1.1     DNS       Standard query A www.mit.edu
     6 0.140633  192.168.1.1     192.168.2.145   DNS       Standard query response A 18.7.22.83
```

```
  ⊞ Destination: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
  ⊞ Source: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
    Type: IP (0x0800)
⊞ Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 192.168.1.1 (192.168.1.1)
⊞ User Datagram Protocol, Src Port: 1565 (1565), Dst Port: domain (53)
⊟ Domain Name System (query)
    [Response In: 6]
    Transaction ID: 0x0003
  ⊞ Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊟ www.mit.edu: type A, class IN
        Name: www.mit.edu
        Type: A (Host address)
        Class: IN (0x0001)
```

```
0000  00 0c 41 45 90 a8 00 09  5b 61 8e 6d 08 00 45 00   ..AE.... [a.m..E.
0010  00 39 73 28 00 00 80 11  42 a9 c0 a8 02 91 c0 a8   .9s(.... B.......
0020  01 01 06 1d 00 35 00 25  21 c1 00 03 01 00 00 01   .....5.% !.......
0030  00 00 00 00 00 00 03 77  77 77 03 6d 69 74 03 65   .......w ww.mit.e
0040  64 75 00 00 01 00 01                               du.....
```

File: "C:\DOCUME~1\PAULAW~1\LOCALS~1\Temp\etherXXXXa01796" 713 Bytes 00:00:(P: 6 D: 6 M: 0 Drops: 0

We see from the above screenshot that *nslookup* actually sent three DNS queries and received three DNS responses. For the purpose of this assignment, in answering the following questions, ignore the first two sets of queries/responses, as they are specific to *nslookup* and are not normally generated by standard Internet applications. You should instead focus on the last query and response messages.

[3] If you are unable to run Wireshark and capture a trace file, use the trace file dns-ethereal-trace-2 in the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip

12. What is the destination port for the DNS query message? What is the source port of DNS response message?
13. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?
14. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
15. Examine the DNS response message. How many "answers" are provided? What do each of these answers contain?

Now repeat the previous experiment, but instead issue the command:

```
nslookup –type=NS mit.edu
```

Answer the following questions[4] :

16. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?
17. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
18. Examine the DNS response message. What MIT nameservers does the response message provide? Does this response message also provide the IP addresses of the MIT nameservers?

Now repeat the previous experiment, but instead issue the command:

```
nslookup www.aiit.or.kr google-public-dns-a.google.com
```

Answer the following questions[5]:

19. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server? If not, what does the IP address correspond to?
20. Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?
21. Examine the DNS response message. How many "answers" are provided? What does each of these answers contain?

[4] If you are unable to run Wireshark and capture a trace file, use the trace file dns-ethereal-trace-3 in the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip

[5] If you are unable to run Wireshark and capture a trace file, use the trace file dns-ethereal-trace-4 in the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip. However, please notice that the trace actually corresponds to the following (slightly different) command:

```
nslookup www.aiit.or.kr bitsy.mit.edu
```