***Lab Title:*** Wireshark – DNS

***Objective of this lab:***

As described in Section 2.5 of the textbook, 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. 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.

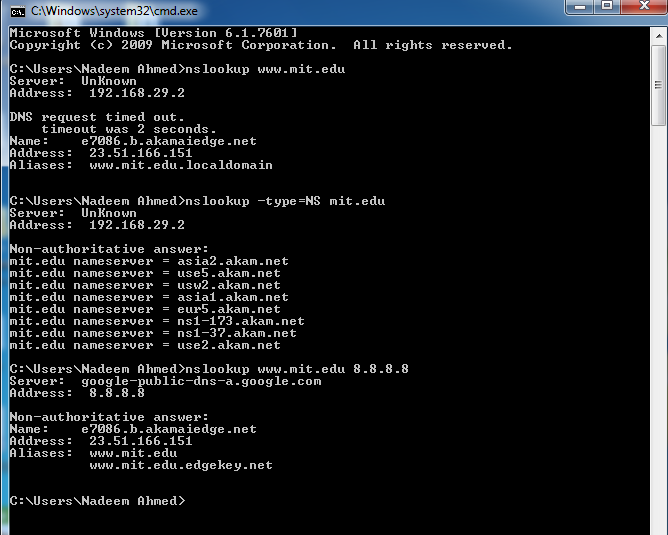
Before beginning this lab, you’ll probably want to review DNS by reading Section 2.5 of the textbook. In particular, you may want to review the material on **local DNS servers**, **DNS** **caching**, **DNS records and messages**, and the **TYPE field** in the DNS record.

1. **Introduction to DNS**

There are two ways to identify a host -- a hostname and an IP address.  People prefer the more mnemonic hostname identifier, while routers prefer fixed-length, hierarchically-structured IP addresses. In order to reconcile these different preferences, we need a directory service that translates hostnames to IP addresses. This is the main task of the Internet's **Domain Name System (DNS)**.  The DNS is (i) a distributed database implemented in a hierarchy of **name servers**and (ii) an application-layer protocol that allows hosts and name servers to communicate in order to provide the translation service.

1. **Introduction to 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 thatsame DNS server, and displays the result.

The above screenshot shows the results of three independent *nslookup* commands (displayed in the Windows Command Prompt). In this example, the client host is using Windows 7 and is connected to an ISP using ADSL modem. When running *nslookup*, if no DNS server is specified, then *nslookup* sends the query to the default DNS server, which in this case is 192.168.29.2. Consider the first command:

1. **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 local DNS server that provides the answer; and (2) the answer itself, which is the host name and IP address (of a CDN node hosting the site) www.mit.edu. Although the response came from the local DNS server, it is quite possible that this local DNS server iteratively contacted several other DNS servers to get the answer, as described in Section 2.5 of the textbook.

Now consider the second command:

1. **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 servers 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 MIT nameservers (again hosted on Akamai CDN). 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. Note that the answer does not include the IP addresses of the authoritative DNS servers for MIT.

Now finally consider the third command:

1. **nslookup www.mit.edu 8.8.8.8**

In this example, we indicate that we want the query sent to the Google public DNS server (well known IP 8.8.8.8) rather than to the default local DNS server (192.168.29.2). Thus, the query and reply transaction takes place directly between our querying host and Google public DNS server. In this example, the Google DNS server provided the IP address of the host [www.mit.edu](http://www.mit.edu) that is similar to the response in first nslookup.

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**

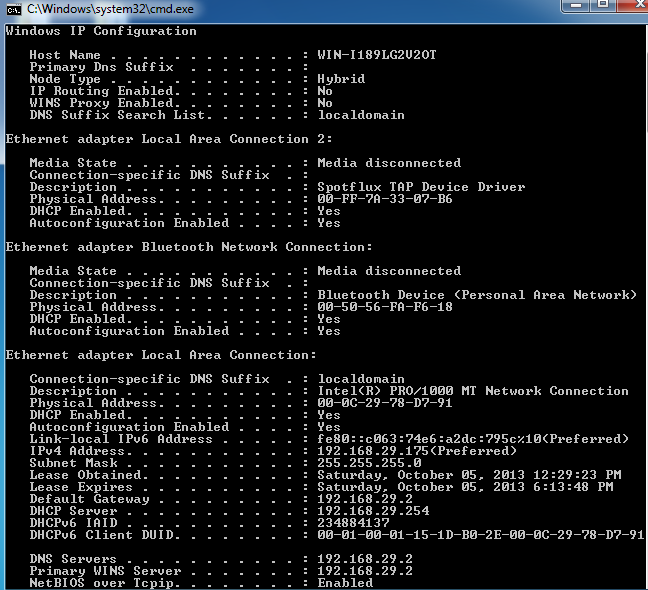
In general, *nslookup* can be run with zero, one, two or more options. And as we have seen in the above examples, the dns-server is optional as well; if it is not supplied, the query is sent to the default local DNS server.

You can also run the nslookup in interactive mode. Simply type nslookup and press enter. Now at the > prompt, type help to see the list of all options available.

1. **Introduction to IPCONFIG**

*ipconfig* (for Windows) and *ifconfig* (for Linux/Unix) are among the most useful littleutilities in your host, especially for debugging network issues. Here we’ll only describe *ipconfig*, although the Linux/Unix *ifconfig* is very similar. *ipconfig* can be used to showyour current TCP/IP information, including your address, DNS server addresses, adapter type and so on. For example, if you can get all this information about your host simply by entering:

1. **Ipconfig /all**



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

*ipconfig* is also very useful for managing the DNS information stored in your host. InSection 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:

1. **ipconfig /displaydns**

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

1. **ipconfig /flushdns**

Flushing the DNS cache clears all entries.

1. **dig**

Dig is a more detailed Name server resolver than NsLookup.

Note that dig is not available on Windows by default but can be installed.

The output from dig can be easily related to the DNS header structure that you learnt in the lecture slides. In this sample interaction, Dig is providing 3 answers for 1 query question against yahoo.com.

