Homework 4 (Due: Mon 12/5)

1. Checking up on the DNS
We’ve looked at factors in the layers up through TCP that can impact home Internet performance. Now it’s time to look at an application layer factor: Web browsing often involves DNS queries and responses to translate site names into IP addresses. If the DNS performance is slow, then Web performance may suffer. This question asks you to learn about the DNS and write a short program to analyze some traces of DNS traffic to see for yourself how well it works.

*We recommend you do this part of the assignment with a partner (only two people per team).*

Try your hand at name resolution

First, let’s see all the packets that are exchanged when a nameserver resolves a name.

**Question 1:** Your task is to use the `dig` tool to explore the DNS. `dig` is installed on the CSE lab machines and is widely available with documentation online. Your friend in Perth, Australia wants to send mail to you at `top-layer@uw.edu`, which requires that they find the hostname and IP address of a machine that accepts mail for accounts at uw.edu. Assume that your friend knows only the root DNS machine `d.root-servers.net` to start, and that when multiple servers are returned in response to queries the one with the highest IPv4 address is selected. You may further assume that you start with no cache of prior DNS responses anywhere, and that no packets are lost. Draw a diagram similar to Figure 7-6 in your textbook that shows the series of queries and responses. Give domain names and IP addresses.

Now analyze real DNS traces

Now, let’s switch to our standard view of a trace taken in a home network. The home computer sends its DNS queries to a nameserver in the ISP, that nameserver follows the process you explored above, and the result is returned to the home computer. Thus the home computer sees only the first query and the final response when resolving a name. We are interested in how long this process takes, as that delay may hold up a subsequent TCP connection to a web site. Note that the home computer also keeps a cache of recent DNS name resolutions so that sometimes it does not need to send any DNS query to learn the answer.

**Question 2:** Suppose you have a trace that contains DNS name resolutions. Describe a procedure by which you would analyze the packets in the trace to compute the name resolution delays. To do this you will have to learn about DNS packets by looking in your textbook, with WireShark, etc. Your procedure should be simple, but it should be robust enough to handle lost request and response packets and interleaved name resolutions. Hint: think about how your computer knows how to match a response to a query.

Luckily, we have a trace of DNS traffic for you to work with. It is up on the course website. From a home computer, we browsed the front page of the top 20 web sites according to Alexa. Only name resolutions that caused DNS traffic to be sent are recorded, and the trace contains only DNS traffic. Open the file in WireShark and have a good look at it. (This will help you answer the
above question too.) There are hundreds of name resolutions even though we visited only 20 sites. You might be interested to see what happens when you visit a web page!

**Question 3:** Write a program that processes a trace to compute all the name resolution delays to the nearest millisecond, and prints a report of the delays in the order that the name resolutions complete. Run it on the trace we supplied.

**Program inputs and outputs:** Call your program “dnsdelay”. Have it run with only the “-t” command-line argument. Have it print the delays to stdout as an XML report that follows the example below (but will be much longer). Each delay tag contains one DNS name resolution delay as an integer. Use the description tag to tell us what was going on in the trace and the DNS setup, e.g., “supplied trace”, or when you make your own trace later perhaps “visiting the top 50 web sites with Google DNS” or “evening browsing on DSL with my usual DNS”. This form is for grading purposes.

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<report>
  <description>Web surfing at home with Comcast DNS</description>
  <delay>285</delay>
  <delay>775</delay>
  <delay>1234</delay>
  <delay>88</delay>
  <delay>33</delay>
  <delay>543</delay>
  <delay>120</delay>
  <delay>213</delay>
</report>
```

**Turn in:**

- Your program source (so we can run it)
- A graph of the delays (pdf) sorted from lowest to highest. Put delay in milliseconds on the y-axis, and sort order on the x-axis. (An easy way to get this data is to add another command-line switch to your program to make it print only the delays, which you can then copy and put into your favorite graphing tool or Excel.)

Now you can see what DNS delays are like!

**Question 4:** Characterize the DNS name resolution delays you computed in no more than two sentences. Then use your networking expertise to tell us why page load times may not be extended by these DNS delays in no more than two sentences.

Now let’s gather some data from your home. Run an experiment in which you use WireShark to capture DNS traffic in a home setup. You can do this by using WireShark on any platform, not in promiscuous mode and with a capture filter of “port 53” so that only DNS traffic is gathered. While you are capturing, use a workload of your choice, e.g., browse normally for a while, use
Alexa to visit the top sites, or something else. Gather at least 1000 DNS packets (which won’t take long).

**Question 5:** Run your dnsdelay program on your trace to compute the DNS name resolution delays and generate a report. Now upload your report as usual using your post tool from HW1 to http://amlia.cs.washington.edu/cse461/dnsdelay/. We want to gather data from different homes and will show you the overall results.

**Just for fun**

Read about Google Public DNS. (Just search on the Web.) Do you think it will improve your Web performance at home? Let’s see. You can try it by following their configuration options. You might use Wireshark to record activity with your ISP’s DNS on one day and with Google DNS on another day, and analyze both. We hope that some of you will try Google DNS for the experiment you run above; please note this in the description. Which did you find was better? Drop us a mail, as we’d like to know.

2. **Textbook**

6.12, 6.14, 6.21, 6.29, 6.32, 6.33