Fundamentals of Computer Networking	Project 1: Simple DNS Client
CS4700/CS5700 Fall 2009	17 September 2009

This project is due at 11:59pm on October 7th, 2009.

1 Description

The Domain Name System (DNS) is a hierarchical system for converting domain names (e.g., `www.google.com`) to Internet Protocol (IP) addresses (e.g., `209.85.129.99`). DNS is often referred to as a "phone book" for the Internet, translating human-friendly domain names into machine-friendly IP addresses. In this project, you will implement a DNS client program, which handles DNS requests by querying other machines. Note that the graduate version of this project has additional requirements, which serve as an opportunity for extra credit for students enrolled in the undergraduate version of this course.

2 Requirements

Your will write a DNS client program which, given a name to query for and a DNS server to query will:

- Construct a DNS query packet for the specified name
- Send the query to the specified DNS server using UDP
- Wait for the response to be returned from the server
- Interpret the response and output the result to `STDOUT`

Your client must support the following features:

- Queries for `A` records (IP addresses)
- Responses that contain `A` records (IP addresses) and `CNAME`s (DNS aliases)
- Retransmissions of queries that are lost

You should be strict; if the returned message does not conform to the DNS specification, you should assert an error. You may receive other packets that are not responses to your query; you should ignore these and continue to wait for a response to your query. Remember that network-facing code should be written defensively. We will test your code by sending corrupted packets to your client; you should handle these errors gracefully and *not* crash.

3 Your client program

For this project, you must use C. In subsequent projects, you will be allowed to use the programming language of your choice. You may *not* use any DNS libraries in your project (such as `getaddrinfo` or `gethostbyname` in C). You must construct the DNS request packet yourself, and interpret the reply yourself.

The command line syntax for your client is given below. The client program takes command line argument of the domain name to interpret and the IP address of the domain server to query. The syntax for launching your program is therefore:

```
dnsclient [-t <timeout>] [-i <max-retries>] [-p <port>] @<server> <name>
```

`timeout` (Optional) Indicates in seconds, how long to wait before regenerating an unanswered query. Default value: 5.

`max-retries` (Optional) Indicates the number of times the resolver will re-generate the query, before it quits with an error, if no response is received from the server. Default value: 3.

`port` (Optional) The UDP port number of the DNS server. Default value: 53.

`server` (Required) The IP address of the DNS server, in `a.b.c.d` format.

`name` (Required) The name to query for.

You should develop your client program on the CCIS Linux machines, as these have the necessary compiler and library support. You are welcome to use your own Linux/OS X machines, but you are responsible for getting your code working, and your code *must* work when graded on the CCIS Linux machines. If you do not have a CCIS account, you should get one ASAP in order to complete the project.

Your code must be `-Wall` clean on `gcc`. Do not ask the TA for help on (or post to the forum) code that is not `-Wall` clean unless getting rid of the warning is what the problem is in the first place.

Your client should print results to standard using the following format:

```
IP <tab> <IP address> <tab> <seconds can be cached> <tab> <auth|nonauth>
CNAME <tab> <alias> <tab> <seconds can be cached> <tab> <auth|nonauth>
NOTFOUND
ERROR <tab> <description of the error>
```

If an response to a query contains multiple answers (such as multiple IP addresses or aliases), your client must print an `IP` or `CNAME` line for each one of these. If the requested name does not exist, your client must print a `NOTFOUND` line. Finally, if an error occurs, your client should print an `ERROR` line containing a description of the error.

4 Extra requirements for graduate version

The graduate version of this project should also support queries for `MX` (mail server) and `NS` (name server) records. Therefore, your program should accept the following input syntax:

`dnsclient [-t <timeout>] [-i <max-retries>] [-p <port>] [-ns|-mx] @<server> <name>`

where the optional `-ns` or `-mx` flags request their respective records (if no flag is given, you should query the `A` record). Your output for these records should look like

```
MX <tab> <alias> <tab> <preference> <tab> <seconds can be cached> <tab> <auth|nonauth>
NS <tab> <alias> <tab> <seconds can be cached> <tab> <auth|nonauth>
```

These requirements are optional for the undergraduate version of this project. If successfully completed, they will serve as 10% extra credit total.

5 Submitting your project

You should submit your project by emailing the instructor at `amislove@ccs.neu.edu`. You should submit a `.tar.gz` file as an attachment to the email, which contains your code. Please have the subject line for the email be `[CS4700] [Project 1 Submission]` followed by your last name and the last name of your other group member(s).

This `.tar.gz` file should unpack to have the following structure

```
project1-{lastname1}-{lastname2}/
                                  Makefile
                                  foo.c
                                  foo.h
                                  ....
```

Your `Makefile` should have a target `dnsclient` which compiles a binary named `dnsclient`. If you are unfamiliar with any of this process, you should attend office hours or the TA's lab hours.

6 Advice

A few pointers that you may find useful while working on this project:

- Remember to convert your integers, shorts, and longs to network ordering (using `hton()` and associated functions).
- You should check the output of your program versus other DNS utilities. On Linux/OS X/UNIX, you can use the `dig` program, which outputs a fair amount of debug information about the requests and responses.

- Check the Blackboard forum for question and clarifications. You should post project-specific questions there first, before emailing the course staff.
- Finally, get started early and come to the TA lab hours - these are held from 5:00pm - 7:00pm on Wednesdays in the lab at 212 West Village H. You are welcome to come to the lab and work, and ask the TA and instructor any questions you may have.