CSci 4211: Introduction to Computer Networks

Programming Assignment

- ## Objectives

In this project, you will learn the concept of how the domain name system (DNS) works and implement a simplified version of both client and server by using sockets.

- ## Details

  - ### Domain Name System

The DNS is a decentralized naming system for computers and devices connected to the internet or other networks. The main purpose of DNS is to translate easily memorized domain names to numerical internet protocol addresses (IP addresses). Since managing and remembering numerical IP addresses are difficult, DNS is a crucial part in networks and has been in use since 1985. Typically DNS uses user datagram protocol (UDP) on port 53 to provide services. However, in this project you are asked to use transmission control protocol (TCP) on different ports to provide services to your clients.

There are two types of DNS queries, recursive queries and iterative queries. Recursive queries will ask the DNS server to do all the job of fetching the final IP address for your request and return it directly to you. In the process, the queried DNS server may also query other DNS servers to get the final result. On the other hand, iterative queries means the DNS server will only resolve part of your request and refer you to other DNS servers for your final result.

You can find more details about the concept of DNS in the video[1].

  - ### Client

In this project, a client is a user interface which sends requests to DNS servers with domain names.

- The client will connect to a default local DNS server at a certain port to query domain names.

- The query should be in the format <hostname, I/R>, where hostname is the host name that the client wants to query for the corresponding IP address and I/R indicates whether this is an iterative request or a recursive request.
- The hostname in the query is case insensitive.
- The client program will end when ‘q’ is entered. All the sockets should be closed properly. Note that the server should still be running and servicing other clients without any problem.
- The client should be initialized with a unique name (ID) in order to print out the log later on.
- When sending/receiving any message to/from the server, print out the whole received message in a log file named <ID.log>. For example, when receiving a request format is invalid message from a server, print out **0xEE, ID, “Invalid format”** in a new line in the log file.
- The client program should take parameters in the following sequence:
    1. Client ID
    2. Server IP
    3. Server Port

○ Server

In this project, the server is responsible for resolving the DNS requests and replies the corresponding IP addresses to clients.

- The server will be listening on a certain port to service incoming DNS requests. Note that the server should be able to handle multiple clients by using the same port.
- The DNS servers will read a file (default.dat) to build the local DNS mapping when initialized. We will test your program based on a different file but with the same format.
- The root DNS server will read a file (server.dat) to get the IP and port of the other DNS servers. The other DNS servers should be listening on the specific ports assigned in the file.

- The server should be initialized with a unique name (ID) in order to print out the log later on.
- When sending/receiving any message to/from clients or other servers, print out the whole received message in a log file named ID.log. For example, when receiving an iterative request from a client, print out **ID, hostname, I** in a new line in the log file.
- Resolved DNS mapping should be cached on the **default local DNS server** when returned by other DNS servers.
- The default local DNS server should dump the cached mapping into a log file named “mapping.log”.
- When shutting down the server, a broadcast message should be sent to all the connected clients and all the sockets should be closed.
- You can implement a single server program for all the servers or implement specific server programs for each of them. However, the com, gov, and org server should use the same program.
- The server program(s) should take the following parameters:
    1. Server ID
    2. Server Port
    3. Default mapping file (com.dat, gov.dat, or org.dat)
    4. Default DNS servers list (server.dat)

The servers will act differently when receiving iterative or recursive requests. This will be further explained in the following section.

- Message Format

Client	
<ID, hostname, I/R>	Send a DNS request to server, I indicates iterative request and R indicates recursive request.
Server	
<0x00, ID, IP>	Resolved IP for a specific DNS request. ID is the server ID.

<0x01, ID, IP, port>	Redirect to the next DNS server for resolving the request.
<0xEE, ID, “Invalid format”>	Request format is invalid.
<0xFF, ID, ”Host not found”>	Error message when fail to resolve the requested hostname.

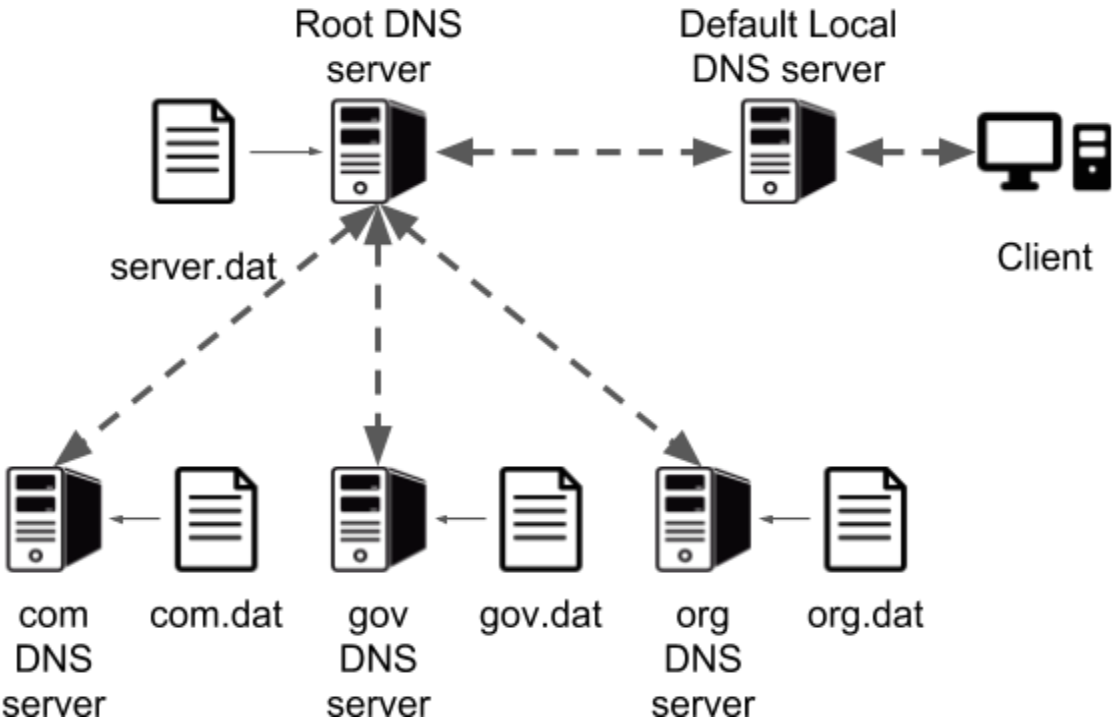
- What to Submit

You should put all your project files in a folder and tar/zip the folder in a single file then upload it to canvas. The project files should include:

- Server source code
- Client source code
- A readme which briefly explains how to compile and run your source code
- The default_local.log generated by using the sample example
- All the log files generated by using the sample example

Please do try to run your program on any CSE-Lab machines as we are going to test your program on them.

The project will have two phases with different targets. Please refer to the following figure for the system overview.



- Phase 1

In phase 1, you will implement a basic client-server program where they are able to communicate through sockets. The default local DNS server should be able to service more than one client with the same port. This corresponds to the default local DNS server and the client in the figure above.

- Phase 2

In phase 2, two different types of DNS requests will be implemented based on the client-server model of phase I.

There will be five servers in total. In this example, the default local DNS server will be listening on port 5352, the root DNS server will be listening on port 5353 with three other DNS servers handling org, gov, and com domains listening on port 5354, 5355, and 5356. These three servers will handle the DNS request for “xxx.org”, “xxx.gov”, and “xxx.com” respectively.

When handling recursive requests, default local DNS server will help the clients to resolve the hostname directly. For example, if a client issues a request for the hostname “google.com”, the default local DNS server will issue a request to the root DNS server. The root DNS server will then send a request to the com DNS server, finally the com

DNS server will resolve the hostname and return the corresponding IP address all the way back to the client.

On the other hand, the root server will return the IP and port of the com DNS server to the default local DNS server when receiving an iterative request with the hostname “google.com”. In this case, the default local DNS server will have to connect to the com DNS server with the information provided by the root DNS server. Finally the com DNS server will resolve the hostname for the default local DNS server where it will cache and return the corresponding IP address to the client.

- Tips
  - Start early before it’s too late
  - Save and compile your code frequently
  - Comment your code properly, it will also be part of your grade
  - Use any programming languages you prefer
- Deadline
  - Phase 1: **November 12, 11:55 p.m.**
  - Phase 2: **November 26, 11:55 p.m**.
- Reference

[1] https://www.youtube.com/watch?v=dE4rsNuG0aw

[2] https://en.wikipedia.org/wiki/Domain_Name_System