# CSCI 367 - Computer Networks I

# Assignment 07 – UDP Transport Layer

# Client-Server –Load Balancing Server

# Program Specification

Create a UDP client-server application that uses the Sockets API.

1. The load balancing server listens for client connections.
2. The client prompts the user for a message or quit to exit the application.
   1. "Enter message or quit to exit: "
3. The client sends the message to the load balancing server.
4. The load balancing server receives the client message and forwards the message to one of two data store servers. The load balancing server should toggle between the two data store servers.
5. The data store server receives the message and saves the message to an ASCII text file.
   1. The message written to the ASCII text file should be prepended with the following two fields:
      1. Date-Time Stamp
      2. Relay Server’s IP address
6. After recording the message information, the data store server sends back a status message represented as an ASCII text string to the load balancing server:
   1. SUCCESS
   2. FAIL
7. The load balancing server forwards the status message to the client application.
8. The client should prompt the user to see if he/she wants to send another message.
9. NOTE: Two data store servers are running:
   1. The data store server’s port number must be a command-line parameter.
   2. The load balancing relay server needs two command-line parameters for each of the two data store server port numbers.

**System Overview**

The implementation of the client and server applications uses the sockets API and the UDP (User Datagram Protocol) transport layer protocol to transmit and receive messages over a TCP/IP network.

The following diagram illustrates a connection between the client and the server.

* + The client sends a message to the load balancing server.
  + The load balancing server receives the client message and forwards the message to one of two data store servers.
  + The data store server receives the message and saves the message to an ASCII text file.
  + After recording the message information, the data store server sends back a status message represented as an ASCII text string to the load balancing server application.
  + The load balancing server forwards the status message to the client application.

**Client Sends Message**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Load Balancing Server Forwards Message to Message Recorder 1**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Message Recorder 1 Sends Status Message to Load Balancing Server**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Load Balancing Server Forwards Status Message to Client**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Client Sends Message**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Load Balancing Server Forwards Message to Message Recorder 2**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Message Recorder 2 Sends Status Message to Load Balancing Server**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

**Load Balancing Server Forwards Status Message to Client**

127.0.0.1



127.0.0.1



4000

8221

6072

13

50444

23

Client Computer

Load Balancing Server

127.0.0.1



13

50445

23

Message Recorder 1

127.0.0.1



13

50446

23

Message Recorder 2

# Program Execution

Compile your source code files.

* If your source code has no syntax errors, an executable file is produced.
* Execute the program and check the accuracy of the program outputs.
* Below is a sample run showing the server and client output, along with the ASCII text file content.

Client

Graphical user interface, text, application

Description automatically generated

Load Balancing Relay Server

Graphical user interface, text, application, email

Description automatically generated

Data Store Server – Port 50445

Graphical user interface, text, application, email

Description automatically generated

Data Store Server – Port 50446

Graphical user interface, text

Description automatically generated

ASCII Text File

Graphical user interface, text, application, email

Description automatically generated

# File Header

At the top of the source code files, add the following file header:

 /\*

# Name:

# Description:

# Date:

# Specification:

\*/

# Submission

1. Upload your source code and header ﬁles to Canvas.
2. For this lab, Canvas has been conﬁgured to permit only files that end with the .c and .h file extensions.