ELEN 4017 – Network Fundamentals

LAB # 3 - Socket Programming with Multithreading

Objective:

1. Get introduced to multithreading and socket programming with multithreading.
2. Get familiarized with the methods in Python language which are used for basic multithreading.
3. Create a basic multithreaded TCP Client and a Server in Python language.

Instructions:

1. This lab must be completed individually by each student.
2. An individual lab report, consisting of responses to the tasks given in this handout, must be submitted to the demonstrators for marking.
3. In addition to the report, demonstrators may ask questions to test your understanding of the material and marks for the lab will be allocated based on your ability to answer the questions.
4. Consult the references in the end to perform your tasks.

Introduction:

Multithreading is an important concept, especially for today's multi-core architecture platforms, where different parts/threads of a process are executed in parallel [1]. This concept is also important for our context of client-server programming where a server is often required to deal with multiple clients in parallel. In such a case, instead of launching multiple server processes for each client, multiple threads of the same server process are launched in order to optimize on the used resources, delays etc.

This lab is about introductory concepts about multithreading i.e. how to launch parallel sub-processes/threads from a single main process. We will make use of Python language to implement basic features of multithreaded programming. There are two ways of implementing multithreading with Python; using the classical thread() function (I believe this has been scraped off in Python 3.x) or making use of the recent threading.thread module/class.

Tasks:

1. Refer to the references for multithreading with Python [2-3] to see how multithreading is implemented via Python.

2. Execute the source code (given along with this lab) of a basic multithreaded program, observe the output.

    a) What is the total execution time of the program? Comment on its value

    b) Identify the classes/modules and methods used for implementation of multithreading in the provided source code.

    c) Modify the source code so that

        i. The program runs with 5 threads

        ii. The delays associated with each of the threads are: for thread # *i,* delay = *i* seconds i.e. for thread # 1, delay = 1 sec, for thread # 2*,* delay= 2 secs, and so on. With such delays, determine the total execution time of the program? Comment on its value

        iii. Make use of other available methods of threading module e.g. get_name(), active_count(), current_thread() to determine their behaviour and the values returned by them.

3. Use the echo client and server programs from last lab and extend them so that the echo server becomes a multithreaded server such that a new thread is created to deal each client.

    a) Do this for both the TCP as well as the UDP echo application.

    b) Test the working of the multithreaded server by creating multiple clients, each sending a unique message to the server that would be echoed.

References:

1. https://en.wikipedia.org/wiki/Thread_(computing)
2. http://www.techbeamers.com/python-multithreading-concepts/
3. https://docs.python.org/3/library/threading.html