COMP 5416 Assignment 2

Due: 5pm, Friday, 15/SEP/2017

Question 1 (Socket Programming for Blackbox Probing)**.** In this question, you are required to build a TCP client to test the *propagation delay* and *link capacity* of a black box shown in the Fig. 1. The black box emulates an uplink, a downlink, and a server. You need to build a TCP client using Python, to connect to the server using hostname

仿真；模仿

`ec2-52-65-60-139.ap-southeast-2.compute.amazonaws.com`

and port number `12010`. You client should be able to send a message to the server and receive a response message from the server. The length of your message is limited to 100 characters.

(1) Send you student number (as a string) to the server and find the response message from the server.

(2) Use Wireshark to capture the response packet. Find the IP address of the server. Screenshot the packet.

(3) In the blackbox, suppose that the uplink has a limited link capacity and the downlink has an infinite link capacity. The propagation delays of the uplink and the downlink are the same. Only transmission delays and propagation delays are considered. All other delays in the system can be ignored (e.g., queueing and processing delays). Revise your client to obtain the propagation delay and link capacity of the uplink. (Hint: You can test the delay from sending a packet to receiving the response.)

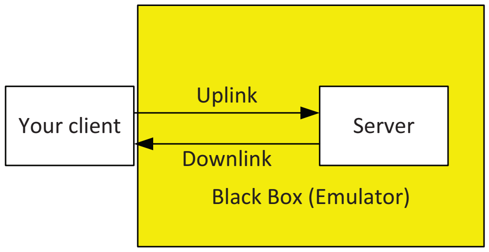
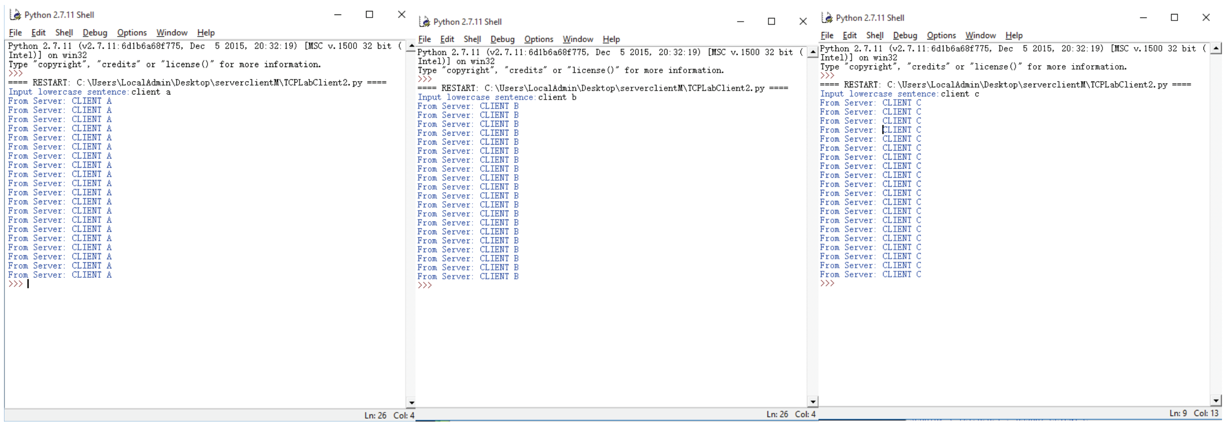


Fig. 1. Question 1.

Question 2 (Socket Programming)**.** You are given the complete code for the TCP Ping client. Your task is to write the TCP Ping server. The Ping client code is in TCPClient2.py. You do not need to modify this code. The client will send 20 ping requests to the server. The server receives the data and converts characters to uppercase, and then sends back 20 ping responses (capitalized data). You should submit your server code as LastnameFirstnameServer.py.

You need to code and implement the server program. The server must be able to serve multiple clients simultaneously. In order to serve multiple clients simultaneously. The server should run multiple threads.

To test the server, you should run multiple clients simultaneously. In the following example in Fig. 2, three clients send "client a", "client b", and "client c" 20 times simultaneously, and they receive "CLIENT A", "CLIENT B", and "CLIENT C" 20 times from the server.



```
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec  5 2015, 20:32:19) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\LocalAdmin\Desktop\serverclientM\TCPLabClient2.py ====
Input lowercase sentence:client a
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
From Server: CLIENT A
>>>
```

```
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec  5 2015, 20:32:19) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\LocalAdmin\Desktop\serverclientM\TCPLabClient2.py ====
Input lowercase sentence:client b
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
From Server: CLIENT B
>>>
```

```
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec  5 2015, 20:32:19) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\LocalAdmin\Desktop\serverclientM\TCPLabClient2.py ====
Input lowercase sentence:client c
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
From Server: CLIENT C
>>>
```

Fig. 2. Example in Question 2.

Question 3 (DHT)**.** Consider the following circular DHT network in Fig. 3. Transmission over each link takes 1 msec. The links are directional as noted in the figure. When a key is found, the key holder creates a direct connection to the query-originating node and transmits the key. The delay on that link is also 1 msec.

(1) What is the average time to search for a key in the above network if the keys are distributed uniformly among the nodes? Show all your work.

(2) Add one direct link in the figure, so that the average time to search for a key is minimized. Show all your work.

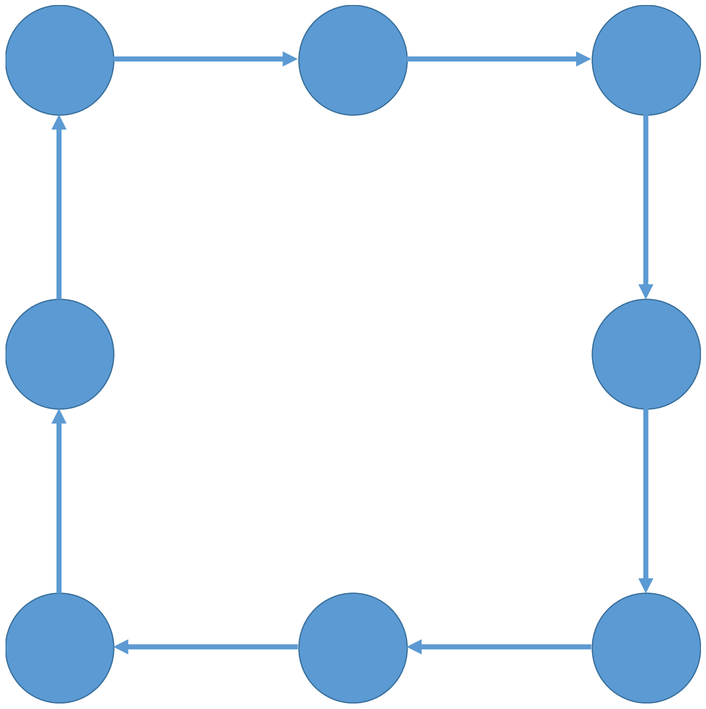


Fig. 3. Topology in Question 3.

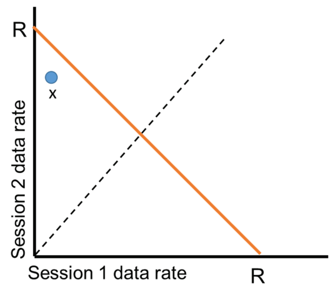
Question 4 (HTTP and TCP)**.** Suppose within your web browser you click on a link to obtain a web page. Assume that the webpage you are fetching has 3 images located on the same server as the main webpage. Each image is a large file that fits into 10 TCP segments. The main page also fits into 1 segment. Assume that the client and server are using TCP Tahoe. The one-way propagation delay in the network is 5 msec. Ignore the size of all packets. Ignore the TCP termination delay (i.e., FIN, FINACK). In parts (a) and (b), there is no loss in the network. ssthresh=8 at the beginning. (Hint: more than one rounds are needed to transmit one large file; you need to consider cwnd in each round.)

(1) How long does it take to obtain the whole web page including the objects if non-persistent HTTP is used? Show all your work.

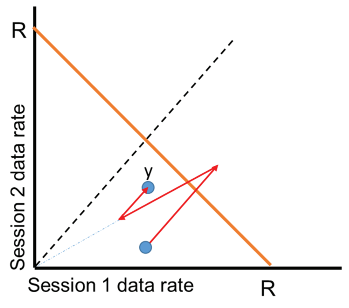
(2) How long does it take to obtain the whole web page including the objects if persistent HTTP is used? (Hint: one TCP connection is established to transmit the main page and three objects. You can start to request the first object if the main page has been received. You can start to request the second object if the first object has been fully received.)

Question 5 (TCP)**.** Review additive increase multiplicative decrease (AIMD) of TCP and complete the following figures. The bottleneck link capacity of the two sessions is R.

(1) In the following figure, the data rates of the two competing TCP sessions start at point x. Sketch the evolution of data rates of the two TCP sessions.



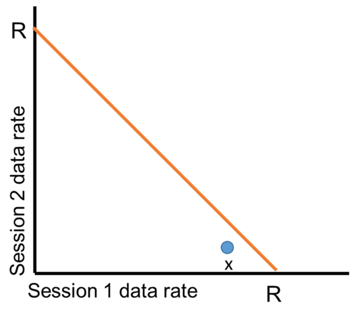
(2) In the following figure, TCP session 2 terminates at point y. Sketch the subsequent evolution of data rates of the two TCP sessions.



(3) In the above questions, both sessions employ the standard AIMD: In the additive increase phase, cwnd is increased by 1 maximum segment size (MSS) every RTT until loss detected; in the multiplicative decrease phase, cwnd is halved after a packet loss. Now, consider the scenario where TCP session 1 employs the standard AIMD, but TCP session 2 selfishly employs a more aggressive AIMD as follows:

- In the additive increase phase, cwnd is increased by K MSS every RTT until loss detected.
- In the multiplicative decrease phase, cwnd is halved after a packet loss (no change).

The data rates of the two competing TCP sessions start at point x. Sketch the evolution of data rates of the two TCP sessions when $K = 2$ and $K = 4$. Explain your new findings


R
x
Session 2 data rate
Session 1 data rate
R