University of Crete, Dept. of Computer Science
CS-435: Network Technology & Programming Lab.
Stefanos Papadakis, Manolis Spanakis
Spring 2016

ASSIGNMENT #2
SOCKETS PROGRAMMING AND THE TCP PARAMETERS

15/3/'16

Report Due: 31/03/'16
Date of Exam: 1/04/'16

Assignment Target:

This assignment is designed so that the students become familiar with the programming of sockets and the TCP fundamentals.

Goals:

1. Learn socket programming.
2. Understand the need for simple networking measurements statistics.
3. Become familiar with TCP, UDP.

Tasks:

0. Write a program that utilizes sockets to transport messages of arbitrary given size between two networked nodes. The program must have command line options for:
    a. Client/Server instance
    b. The transport layer protocol to be used (TCP/UDP)
    c. The number of messages to send in a run
    d. The buffer size used in case of TCP or the message payload size in bytes in case of UDP.
    e. Number of parallel streams. Each stream should use its own socket and port number.

   The following must be outputs of the program:
    a. Average throughput for a run
    b. Delay per packet; interarrival jitter, mean and variance for a run
       Tip: To measure the RTT of a message, construct the following protocol in your program: when a message is received by the server, it is acknowledged by a duplicate transmission back to the client. RTT is then the time elapsed between the message transmission and the duplicate reception and therefore the message delay can be assumed to be RTT/2.[1]

1. Report the Maximum Transfer Unit (MTU) for the path initiated at a COMLAB workstation to the www.edet.gr server, using ping (use the "don't fragment" option). Report the ping Maximum Segment Size (MSS); Use netstat to report the fragmentation of datagrams with size larger than the

[1] Can you think of a way to calculate the one-way delay? Extra bonus if you implement it!

MTU. What is the MSS for TCP and what for UDP over the same e2e connection?

2. Connect one pair of the lab PCs on the switch. Set the network interfaces to 100Mbps (Full Duplex). Using your program, report the mean delay and throughput for TCP and UDP using in each case enough messages of payload equal to 0 and 2^i bytes for i=[4,6,8,10,12,14,16] for 30 seconds run-time. Present your mean delay results versus the payload size in two graphs (TCP, UDP); use error bars to indicate the minimum and maximum values observed per sample size, do likewise for mean throughput.

3. For the message size that had the largest mean value of throughput in (2), conduct for the same number of messages throughput & delay measurements for 2, 4, 8 and 16 parallel connections. Report what you observe. Is the bandwidth equally shared among the flows?

4. For the message size that had the largest mean value of throughput in (2), conduct for the same number of messages throughput & delay measurements for TCP window sizes of 0.01 up to 100 times, in steps of 10x, of the default used by the Linux distribution you have. Report which window is the optimum and explain why?

5. Repeat 2 & 4 setting the network interfaces to 10Mbps (full duplex). Comment the differences focusing on the optimum TCP window.

6. Repeat 2 & 4 setting the network interfaces to 1000Mbps (full duplex). Comment the differences focusing on the optimum TCP window.

7. Draw a plot of the capacity percentage usage ($\frac{average\,throughput}{capacity} * 100\%$) for each payload overlaying all the results of the 10,100,1000Mbps. Comment the results.

8. Draw the plot of the theoretical capacity percentage usage for payload 0 to MTU bytes (default 10/100 Ethernet size), overlaying the curves of the 10,100,1000Mbps Ethernet (for TCP & UDP). For 1000Mbps draw also a separate plot assuming jumbo frame MTU size (9000bytes). How do the theoretical with the measured values compare?

9. Draw the plot of the theoretical maximum packets per second (pps) capacity for payload 0 to MTU bytes (default 10/100 Ethernet size), overlaying the curves of the 10,100,1000Mbps Ethernet (for TCP & UDP).

10. Repeat 4 by disabling the Nagle Algorithm of TCP. Plot the results along with the results of the task 4. Which window size is the optimum? Please explain the reasons. (*see man tcp(7) for more information*)

11. Repeat 2 for the UDP case, using a socket send buffer of 64, 128, 512 and 8192. The value of the socket send buffer can be altered either through the */proc* interface, or by using *setsockopt()*. Do not get confused with the send buffer size of your application. What differences do you observe in the results compared to the results of Task 2?