EE 542 – Laboratory Assignment #2: Network Bandwidth Measurement (Team)

Instructor: Young H. Cho T.A.: Yue Shi

Report Due date: September 7, 2019 at 11:59pm

1. Exploring Bandwidth and Throughput

This exercise will let you explore measuring bandwidth and throughput on your 1Gbps network testbed. Your report should contain tables or graphs that show your results in a meaningful manner, and more importantly your thoughts and conclusions as to what the results mean. Remember, never perform an experiment just once, make sure to collect several runs and average the results. Please include any files you create as an addendum to your report. Have fun!

You'll need to familiarize yourself with the iperf tool. Use Linux OS do all of the tests. Be complete in tests and explanations.

Go through some/all of the following tutorials before you begin (depends on your level of familiarity):

http://www.ee.surrey.ac.uk/Teaching/Unix/
https://www.garron.me/en/linux/visudo-command-sudoers-file-sudo-default-editor.html
https://openmaniak.com/iperf.php
https://netbeez.net/blog/how-to-use-the-linux-traffic-control/
https://www.poftut.com/linux-ethtool-tutorial-usage-examples/

1.1. Bandwidth-Delay Product.

1) Use a network of Linux computers connected via 1Gbps links and a Gbps router.
   Use the ping tool to measure the round-trip delay between two machines.
   Use iperf (look at iPerf hint section 1.3) in UDP mode to measure the maximum bandwidth
   Now use iperf in TCP mode to measure the TCP throughput.

2) Now use 'netem' on one of the computer to artificially inject a delay to the link in one direction.
   For example, if your ethernet link is named 'eth0'

   ```
   # sudo tc qdisc add dev eth0 root netem delay 100ms
   ```

   should add an approximate delay of 100ms to the network link.
   Verify the added network delay using ping and iperf.

   Perform 4 additional emulated network latency experiments using netem. (i.e. 10ms, 50ms, 200ms, and 500ms)

   Test the correct functioning of the tool with ping and iperf

3) Now use 'ethtool' to change speed of your Ethernet link.
   For example, if your ethernet link is named 'eth0'

   ```
   # sudo ethtool -s eth0 speed 10
   ```

This should change the speed of your Ethernet link from 1000Mbps down to 10Mbps. In order for the entire network to have the same speed, this command needs to be entered in both of the machines.

Verify the changed speed using iperf.
Change the network speed to 10Mbps, 100Mbps, and 1000Mbps and do the step (2).
Comment on the results at this point. Were you surprised at all?

Find out the receive and send network buffer sizes

```
# sudo cat /proc/sys/net/core/rmem_max
# sudo cat /proc/sys/net/core/wmem_max
```

4) Change the maximum size for the send/receive windows to 2Mbyte:

```
# sudo sysctl -w net.core.rmem_max==2097152
# sudo sysctl -w net.core.wmem_max==2097152
```

Perform step 2 and 3 again and comment on the result. What do you think is going on?
Experiment with the window sizes and comment on the results.

5) Now use the -w option for iperf to change the send and receive windows to 64, 128 and 256 kilobytes while again measuring the TCP throughput.

6) Comment on your new results. You might want to put all of these results in a table.
At 100Mbit/s and 25ms of delay, what is the theoretically optimal send/receive window size?
If you run your experiment at this size, do you get 100Mbit/s of throughput?
If yes, why? If not, what could cause your link to run at less than full bandwidth?

1.2.Bandwidth, Delay, and Loss.

Set the network operate at 1Gbps with 25ms delay.

1) Use ping to determine the average Round-Trip-Time (RTT) between the computers.

2) What is the bandwidth-delay product for this link?
Use iperf in UDP mode to measure the bandwidth.
Use iperf in TCP mode to measure the TCP throughput.

3) If you left the send/receive windows at their default values, would this link perform well?

4) Can we scale the send/receive windows to see the effect of a bandwidth-delay product/TCP window size mismatch?
Does it make sense to do so?

5) Set the send/receive window values to something too small (i.e. about the value calculated in (2). Now measure the TCP throughput. What happened?

6) Comment on these results. Specifically, what happens if we have a gigabit speed link with 1ms of delay? 5ms?

   Would you be happy with a 1GB DSL connection knowing you can only control the receive window size (hint: the RTT to the east coast of the US from Los Angeles can be 50ms or greater)?

7) Use 'netem' on one of the computer to Emulate packet loss. For example, if your ethernet link is named 'eth0'

```
# sudo tc qdisc change dev enp0s8 root netem loss 0.1%
```

This causes 1/10th of a percent (i.e 1 out of 1000) packets to be randomly dropped.
Verify the packet loss using iperf.

1.3. iPerf hints

1) To test the available bandwidth using iperf in UDP mode, use the following on the server machine:

```
# iperf -s -u -p <port num>
```

Where <port num> is any port number you choose > 1024

2) Then on the client:

```
# iperf -c <server name> -u -p <port num> -b <BW>
```

   Where <server name> is the name of the server (defined in your experiment setup), and <port num> is the same as chosen above. <BW> is the attempted bandwidth. Keep increasing this number towards the link speed until you see packet loss. Report the highest bandwidth possible with no packet loss.

3) To test TCP throughput using iperf use the following server line:

```
# iperf -s -p <port num> -w <window size>
```

4) And the following client:

```
# iperf -c <server name> -p <port num> -w <window size>
```

   Where <server name> and <port num>are defined as above. <window size> is the window size you would like to use for send (on the client) and receive (on the server).

2. Secure Copy

The `secure copy program' scp is a standard tool on modern UNIX-like machines. It is used to copy files between machines, securely and reliably. However, as we will see, it does not always provide good throughput.

1) Set the network to operate at 100Mb/s.

2) Once the experiment is started, use the DETER website to set the delay on the link to zero. What is the RTT delay between the nodes now? What is the bandwidth delay product?

3) On each node create and mount a temporary local file system:

```
# sudo mkextrafs /mnt
```

4) On one node, create a 200MB file in /mnt:

```
# cd /mnt
# dd if=/dev/urandom of=data.bin bs=1m count=200
```

5) Explain briefly what the above commands do.

6) Copy the file to the other node:

```
# scp data.bin USER@othernode_IP:/mnt
```

7) What transfer throughput do you get?

8) Explore the problem: increase the delay on the link. What happens to the throughput? Can you improve the performance using the kernel parameters (i.e. default and maximum TCP window sizes)? Make sure to explore RTT delays at least up to 50ms.

9) Does scp seem to have some sort of built-in limitation? Can you guess what it is? Hint: scp uses the SSH protocol to transfer data.

10) Now set your delay back to zero. At the same time add a small amount of packet-loss to the link. Start at 0.1% (i.e. 1 packet dropped in 1000) and test at various loss levels up to 5%. Graph throughput vs. loss. Discuss these results. Does this seem extreme? Can you explain why this happens?

11) Estimate (or measure) the delay and loss from Los Angeles to Switzerland. If you had a physicist for a colleague, and he wanted to download some data from the new atom smasher at CERN, would you expect him to come to you for help? Can you help him? If so, how?

Create a report slide deck with results and answers to all of the above questions and demonstrate your work on the video recording.