Measuring Bandwidth using the Command Line

Goal:

To test and provide an empirical evaluation of the performance, specifically the effective bandwidth, of a network link. To learn how to find effective bandwidth and baseline transfer rate.

Procedures:

We want to get the empirical evaluation of the performance of SCU Network using Ubuntu system computer in our lab.

First, we log in to our school sever using ssh command and transferred the file. For that, we used this script to run to test with different size of file from 0MB to 100MB with step 10MB after we established SSH connection.

```
time cat ~/lab2.txt|head -c 0MB|ssh taung@linux.scudc.scu.edu "(cat -> ~/COEN146/)"
```

Later, I used the below python script given to automate the process and collected data in data.log file.

```
# -*- coding: utf-8 -*-
# Thida Aung
# COEN 146 lab2 a program to generate the data.log of runtime per each different file size on #
network using this command
# time cat ~/lab2.txt|head -c 0MB|ssh taung@linux.scudc.scu.edu "(cat -> ~/COEN146/)"

import subprocess

megs = 10**6
with open("data.log", 'w') as outfile:
        for file_size in range (0,1000*megs,100*megs):
        for run_num in range(10):
        subprocess.call('>&2 echo for file size: {}'.format(file_size),stderr = outfile, shell= True)
        subprocess.call('time cat ~/COEN146/lab2.txt|head -c {}|ssh -q taung@linux.scudc.scu.edu
"(cat -> ~/COEN146/lab2copy.txt)"\n '. format(file_size),stderr = outfile, shell= True )
```

There was a system delay during file transfer process due to overwhelm amount of users at the same time. So I recollected them almost at the end of the lab once system is not too slow which gave me the results as shown below.

Then, I calculate the time it takes to transfer each volume in excel along with average over 10 run on same volume. Then I plot the graph to find trans The graph that corresponds different data volumes in X axis and to indicate transfer time or effective bandwidth in the Y axis.

Result:

This is my **data.log** below. (transferred file size starting from 0 to 900MB)

```
for file size: 0

real    0m0.557s
user    0m0.016s
sys     0m0.009s
for file size: 0

real    0m0.185s
user    0m0.023s
sys     0m0.009s
for file size: 0

real    0m0.178s
```

user 0m0.015s
sys 0m0.010s

for file size: 0

real 0m0.189s
user 0m0.018s
sys 0m0.007s

for file size: 0

real 0m0.188s
user 0m0.025s
sys 0m0.011s

for file size: 0

real 0m0.171s
user 0m0.019s
sys 0m0.016s

for file size: 0

real 0m0.193s
user 0m0.016s
sys 0m0.011s

for file size: 0

real 0m1.023s
user 0m0.016s
sys 0m0.017s

for file size: 0

real 0m0.544s
user 0m0.023s
sys 0m0.012s

for file size: 0

real 0m0.279s
user 0m0.023s
sys 0m0.006s

for file size: 100000000

real 0m2.471s
user 0m0.518s
sys 0m0.431s

for file size: 100000000

real 0m2.509s
user 0m0.516s
sys 0m0.421s

for file size: 100000000

real 0m3.689s
user 0m0.532s
sys 0m0.421s

for file size: 100000000

real 0m2.542s
user 0m0.537s
sys 0m0.409s

for file size: 100000000

real 0m2.461s
user 0m0.495s
sys 0m0.445s

for file size: 100000000

real 0m2.410s
user 0m0.535s
sys 0m0.398s

for file size: 100000000

real 0m3.568s
user 0m0.531s
sys 0m0.401s

for file size: 100000000

real 0m2.466s
user 0m0.502s
sys 0m0.423s

for file size: 100000000

real 0m3.633s
user 0m0.509s
sys 0m0.367s

for file size: 100000000

real 0m4.730s
user 0m0.540s
sys 0m0.404s

for file size: 200000000

real 0m5.268s
user 0m1.068s
sys 0m0.850s

for file size: 200000000

real 0m5.887s
user 0m1.066s
sys 0m0.708s

for file size: 200000000

real 0m5.113s
user 0m1.099s
sys 0m0.896s

for file size: 200000000

real 0m6.327s
user 0m1.030s
sys 0m0.783s

for file size: 200000000

real 0m5.607s
user 0m1.058s
sys 0m0.824s

for file size: 200000000

real 0m5.898s
user 0m0.996s
sys 0m0.817s

for file size: 200000000

real 0m5.640s
user 0m1.016s
sys 0m0.822s

for file size: 200000000

real 0m5.085s
user 0m0.996s
sys 0m0.725s

for file size: 200000000

real 0m7.435s
user 0m1.660s
sys 0m1.105s

for file size: 200000000

real 0m6.880s
user 0m1.023s
sys 0m0.786s

for file size: 300000000

real 0m7.833s
user 0m1.566s
sys 0m1.166s

for file size: 300000000

real 0m7.696s
user 0m1.514s
sys 0m1.122s

for file size: 300000000

real 0m10.297s
user 0m1.587s
sys 0m1.079s

for file size: 300000000

real 0m7.930s
user 0m1.483s
sys 0m1.233s

for file size: 300000000

real 0m8.285s
user 0m1.543s
sys 0m1.222s
for file size: 300000000

real 0m7.633s
user 0m1.519s
sys 0m1.247s

for file size: 300000000

real 0m7.861s
user 0m1.559s
sys 0m1.209s

for file size: 300000000

real 0m10.029s
user 0m1.559s

sys 0m1.189s

for file size: 300000000

real 0m7.843s
user 0m1.314s
sys 0m1.085s

for file size: 300000000

real 0m10.026s
user 0m1.467s
sys 0m1.104s

for file size: 400000000

real 0m9.898s
user 0m1.919s
sys 0m1.429s

for file size: 400000000

real 0m12.293s
user 0m2.012s
sys 0m1.412s
for file size: 400000000

real 0m11.183s
user 0m2.031s
sys 0m1.556s

for file size: 400000000

real 0m9.901s
user 0m1.988s
sys 0m1.585s

for file size: 400000000

real 0m12.381s
user 0m2.044s
sys 0m1.576s

for file size: 400000000

real 0m10.055s
user 0m2.099s
sys 0m1.587s
for file size: 400000000

real 0m12.938s
user 0m2.331s
sys 0m1.634s

for file size: 400000000

real 0m10.716s
user 0m1.975s
sys 0m1.453s

for file size: 400000000

real 0m13.013s
user 0m2.641s
sys 0m2.000s

for file size: 400000000

real 0m11.216s
user 0m2.046s
sys 0m1.569s

for file size: 500000000

real 0m13.818s
user 0m2.632s
sys 0m1.884s

for file size: 500000000

real 0m13.846s
user 0m2.639s
sys 0m1.903s

for file size: 500000000

real 0m13.451s
user 0m2.644s
sys 0m1.858s

for file size: 500000000

real 0m12.937s
user 0m2.535s
sys 0m2.034s

for file size: 500000000

real 0m12.864s
user 0m2.574s
sys 0m1.845s

for file size: 500000000

real 0m12.799s
user 0m2.637s
sys 0m1.905s

for file size: 500000000

real 0m13.917s
user 0m2.563s
sys 0m1.869s

for file size: 500000000

real 0m12.951s
user 0m2.542s
sys 0m1.919s

for file size: 500000000

real 0m12.937s
user 0m2.632s
sys 0m1.889s

for file size: 500000000

real 0m13.786s
user 0m2.586s
sys 0m2.045s

for file size: 600000000

real 0m15.408s
user 0m3.059s
sys 0m2.368s

for file size: 600000000

real 0m15.663s
user 0m3.156s
sys 0m2.317s

for file size: 600000000

real 0m15.041s
user 0m3.123s
sys 0m2.226s

for file size: 600000000

real 0m15.576s
user 0m3.145s
sys 0m2.235s

for file size: 600000000

real 0m16.225s
user 0m3.215s
sys 0m2.230s

for file size: 600000000

real 0m14.675s
user 0m3.118s
sys 0m2.216s

for file size: 600000000

real 0m15.092s
user 0m3.079s
sys 0m2.299s

for file size: 600000000

real 0m15.491s
user 0m3.066s
sys 0m2.422s

for file size: 600000000

real 0m15.373s
user 0m3.247s
sys 0m2.187s

for file size: 600000000

real 0m14.286s
user 0m3.061s
sys 0m2.207s

for file size: 700000000

real 0m17.541s
user 0m3.715s
sys 0m2.535s

for file size: 700000000

real 0m19.900s
user 0m3.659s
sys 0m2.477s

for file size: 700000000

real 0m17.582s
user 0m3.701s
sys 0m2.571s

for file size: 700000000

real 0m18.235s
user 0m3.634s
sys 0m2.695s

for file size: 700000000

real 0m17.720s
user 0m3.592s
sys 0m2.467s

for file size: 700000000

real 0m19.052s
user 0m3.695s
sys 0m2.541s

for file size: 700000000

real 0m17.490s
user 0m3.536s
sys 0m2.623s

for file size: 700000000

real 0m20.144s
user 0m3.728s
sys 0m2.612s

for file size: 700000000

real 0m18.160s
user 0m3.731s
sys 0m2.501s

for file size: 700000000

real 0m18.269s
user 0m3.681s
sys 0m2.577s

for file size: 800000000

real 0m21.811s
user 0m4.200s
sys 0m2.945s

for file size: 800000000

real 0m22.310s
user 0m4.088s
sys 0m2.923s

for file size: 800000000

real 0m20.971s
user 0m4.407s
sys 0m3.248s

for file size: 800000000

real 0m23.584s
user 0m4.528s
sys 0m3.115s

for file size: 800000000

real 0m20.477s
user 0m3.549s
sys 0m2.790s

for file size: 800000000

real 0m19.299s
user 0m3.912s
sys 0m2.729s

for file size: 800000000

real 0m20.028s
user 0m4.053s
sys 0m2.909s

for file size: 800000000

real 0m19.839s
user 0m4.085s
sys 0m3.198s

for file size: 800000000

real 0m19.585s
user 0m3.976s
sys 0m2.761s

for file size: 800000000

real 0m20.333s
user 0m4.142s
sys 0m2.870s

for file size: 900000000

real 0m23.369s
user 0m4.539s
sys 0m3.194s

for file size: 900000000

real 0m23.237s
user 0m4.198s
sys 0m2.896s

for file size: 900000000

real 0m22.612s
user 0m4.126s
sys 0m2.824s

for file size: 900000000

real 0m22.211s
user 0m4.429s
sys 0m3.582s

for file size: 900000000

real 0m22.706s
user 0m4.313s
sys 0m3.278s

for file size: 900000000

real 0m22.759s
user 0m3.681s
sys 0m2.552s

for file size: 900000000

real 0m22.385s
user 0m4.255s
sys 0m2.825s

for file size: 900000000

real 0m22.480s
user 0m4.056s
sys 0m2.900s

for file size: 900000000

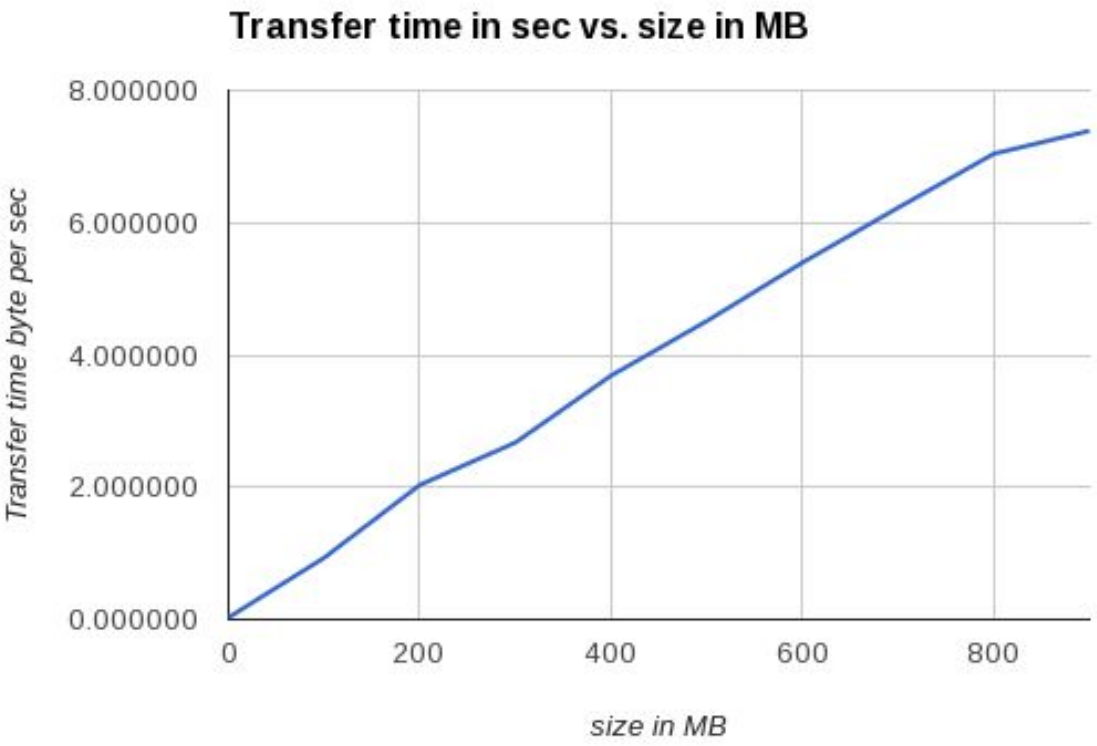
real 0m23.945s
user 0m4.701s
sys 0m3.438s

for file size: 900000000

real 0m21.248s
user 0m4.592s
sys 0m3.514s

Graph & Data:

Below is the graph showing an X axis that corresponds to different data volumes from 0MB to 900MB and the Y axis indicate either transfer time(bytes per seconds) over the average of 10 runs. Note here the graph shown here is size in X axis where as the slope is bytes per seconds which was not y/x exactly instead x/y.



Tables of Data collected over 10 runs

Size in MB	Run# 1	2	3	4	5	6	7	8	9	10	Average Transfer time in sec for 10 runs
0	0.025	0.032	0.025	0.025	0.036	0.035	0.027	0.033	0.035	0.029	0.030200
100	0.949	0.937	0.953	0.946	0.94	0.933	0.932	0.925	0.876	0.944	0.933500
200	1.918	1.774	1.995	1.813	1.882	1.813	2.854	1.721	2.765	1.809	2.034400
300	2.732	2.636	2.666	2.716	2.765	2.766	2.768	2.748	2.399	2.571	2.676700
400	3.348	3.424	3.587	3.573	3.62	3.686	3.965	3.428	4.641	3.615	3.688700
500	4.516	4.542	4.502	4.569	4.419	4.542	4.432	4.461	4.521	4.631	4.513500
600	5.427	5.473	5.349	5.38	5.445	5.334	5.378	5.488	5.434	5.268	5.397600
700	6.25	6.136	6.272	6.329	6.059	6.236	6.159	6.34	6.232	6.258	6.227100
800	7.145	7.011	7.655	7.643	6.339	6.641	6.962	7.283	6.737	7.012	7.042800
900	7.733	7.094	6.95	8.011	7.591	6.233	7.08	6.956	8.139	8.106	7.389300

Analysis:

During this lab, there are 3 different types of times collected; real, user and sys in data.log file. We need users+sys to tell us how much actual CPU time is used for this process, since

- **Real** is wall clock time - time from start to finish of the call. This is all elapsed time including time slices used by other processes and time the process spends blocked (for example if it is waiting for I/O to complete).
- **User** is the amount of CPU time spent in user-mode code (outside the kernel) *within* the process. This is only actual CPU time used in executing the process. Other processes and time the process spends blocked so we do not count towards this figure.
- **Sys** is the amount of CPU time spent in the kernel within the process. This means executing CPU time spent in system calls *within the kernel,* as opposed to library code, which is still running in user-space. Like 'user', this is only CPU time used by the process. (quoted from stackoverflow)

Bandwidth is the amount of data that can be transferred from one point to another normally measured in seconds. By increasing the bandwidth, we don't increase the speed but capacity which is what we did with testing from 0MB to 900MB.

Then, we observe the latency to be 0.0302 byte per second which is caused by the distance and the quality of the medium that the Internet packets travel through in real time. Capacity increased and speed stay the same. Basically, that is our **baseline transfer rate**. Finding the **baseline transfer rate/data transfer rate** is to determine the consistency of the connections to the application server during its normal operation. We can attempt to measure latency overhead needed to setup the connection using baseline transfer rate whether the issues being experienced are a result of high utilization or a high user load.

Deduct that from every rows (see below) we get actual speed at at which data can be transmitted on a connection which is our **Effective bandwidth**.This is as opposed to the theoretical maximum that the connection can carry. I calculated the effective bandwidth which is the slope of our graph (bits per seconds in our case), which I took to average over 10 runs to be 6.10*10^09. We see that in table below too.

By taking size in KB divide by transfer rate in seconds, we get our transfer rate in KB per sec as last column. Transferring Average = 99569.049 KB data per sec is pretty fast considering that we had delay in our network while everyone was trying to access it.

Efffective bandwidth = slope bits/seconds	Average Transfer time in sec for 10 runs	actual transfer time = average time - latency at 0 MB	transfer rate in KB per sec
0	0.030200	0.000	0
8.86E+08	0.933500	0.903	107123.728
1.45E+09	2.034400	1.101	98309.084
3.74E+09	2.676700	0.642	112078.305
3.16E+09	3.688700	1.012	108439.288
4.85E+09	4.513500	0.825	110778.775
5.43E+09	5.397600	0.884	111160.516
6.75E+09	6.227100	0.830	112411.877
7.85E+09	7.042800	0.816	113591.185
2.08E+10	7.389300	0.347	121797.735
Average = 6.10E+09			Average = 99569.049 KB per sec