PERFORMANCE EVALUATION

BENCHMARKING TOOL

PA 1

DESIGNED BY:

SAMI AHMAD KHAN
A20352677
COURSE: CS 553 CLOUD COMPUTING

1. INTRODUCTION

The purpose of this assignment was to benchmark different computer components like CPU, Memory, Disk and Network. Benchmarking is done on them by calculating their operations per second by causing load on them and hence evaluating the performance for the same.
This document has experimental values of all the benchmarks. First I have described the environment, then the specifications of the experiment for each benchmark is presented and at last all the results will be shown and analyzed.

2. EXPERIMENTAL ENVIRONMENT

For all the benchmarks, I have performed all the experiments on Amazon **AWS.**

Amazon AWS Information:

It is a collection of remote computing services, also called web services, that make up a cloud-computing platform offered by Amazon.com. These services operate from 11 geographical regions[2] across the world. The most central and well-known of these services arguably include Amazon Elastic Compute Cloud, also known as "EC2", and Amazon Simple Storage Service, also known as "S3".

Amazon Linux **AMI 2015.09.1 (HVM)**
SSD Volume Type **- ami-f0091d9**
Instance ID: **i-8851de4c**
Instance Type**: t2.micro**

3. OVERALL EXPERIMENTS

1. CPU Benchmarking:

Measuring the processor speed, in terms of floating point operations per second (Giga FLOPS, 109 FLOPS) and integer operations per second (Giga IOPS, 109 IOPS) at varying concurrency levels (1 and 2)

2. Memory Benchmarking :

- Measuring the memory speed of the host by using read and write operations (e.g. memcpy), sequential access, random access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread & 2 threads) .
- The metrics measured are throughput (Megabytes per second, MB/sec) and latency (milliseconds, ms)

3. Network Benchmarking:

Measuring the network speed between 2 instances. The parameter space includes the TCP protocol stack, UDP, varying packet/buffer size (1B, 1KB, 64KB), and varying the concurrency (1 thread & 2 threads).

4. EXPERIMENT RESULTS AND ANALYSIS

This section consists of experimental results for each benchmark and explanations for the same.

4.1 CPU Benchmark Results

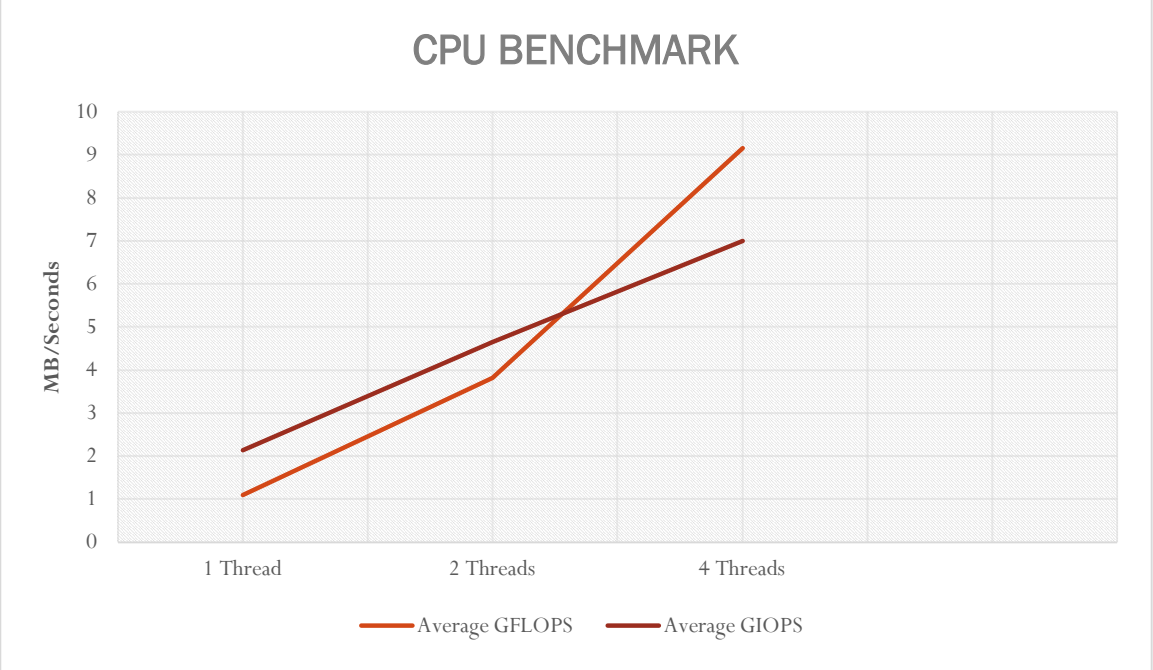
For finding CPU benchmark results, I found out GFLOPS and GIOPS value for different number of threads, i.e. 1, 2 and 4 threads.

a). GFLOPS

Thread #	Average GFLOPS	Latency (in Millisecond)
1	1.0876	2201.7254
2	3.8153	2987.3245
4	6.1541	3001.1233

b). GIOPS

Thread #	Average GIOPS	Latency (in Millisecond)
1	2.1342	2109.1134
2	4.6477	2678.7501
4	7.0012	3091.90



From this graph above, it is very well evident that with the increase in the number of threads, the computation power of the cores increases. And hence the overall performance.

THEORITICAL CALCULATIONS

Theoretical peak performance of CPU is given by,

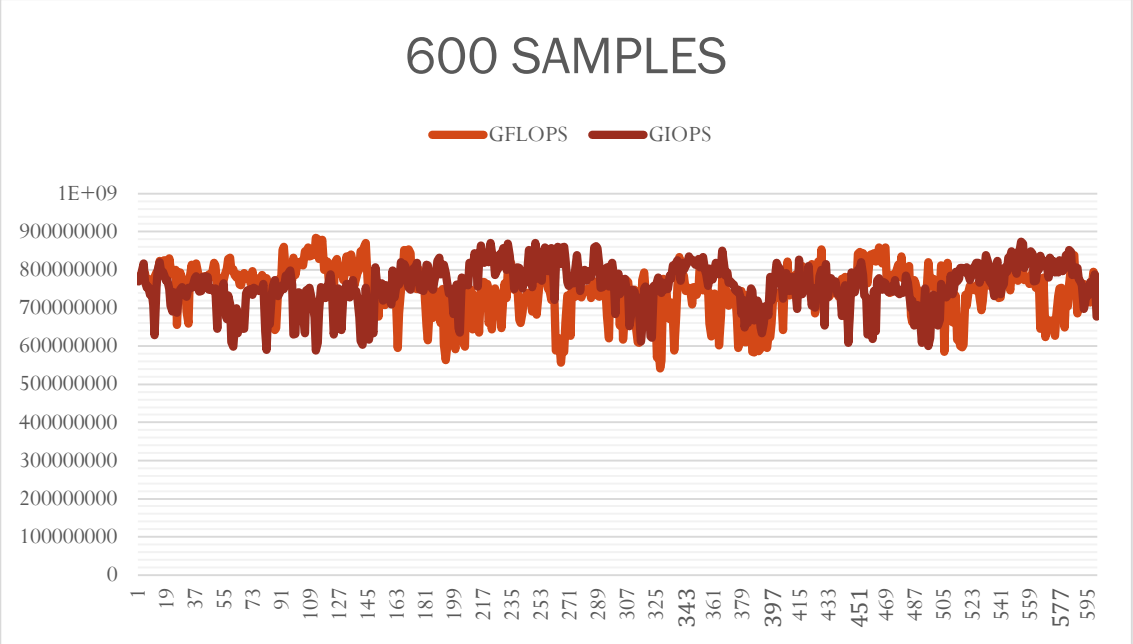
No. of Cores * CPU Speed(GHz) * Instructions per cycle = 2 * 2.7 * 420.8 GFLOPS

LINPACK Comparison:

A benchmark tool was provided to compare my CPU benchmark with the LINPACK's benchmark

```
Number of threads: 1

Parameters are set to:

Number of tests: 15
Number of equations to solve (problem size) : 1000  2000  5000  10000 15000 18000 20000 22000 25000 26000 27000 30000 35000 40000 45000
Leading dimension of array                  : 1000  2000  5008  10000 15000 18008 20016 22008 25000 26000 27000 30000 35000 40000 45000
Number of trials to run                     : 4     2     2     2     2     2     2     2     2     2     1     1     1     1     1
Data alignment value (in Kbytes)            : 4     4     4     4     4     4     4     4     4     4     4     1     1     1     1

Maximum memory requested that can be used=800204096, at the size=10000

================ Timing linear equation system solver ================

Size   LDA    Align. Time(s)    GFlops   Residual     Residual(norm) Check
1000   1000   4      0.026      25.9030  9.632295e-13 3.284860e-02   pass
1000   1000   4      0.026      25.9429  9.632295e-13 3.284860e-02   pass
1000   1000   4      0.026      25.8274  9.632295e-13 3.284860e-02   pass
1000   1000   4      0.026      26.0581  9.632295e-13 3.284860e-02   pass
2000   2000   4      0.206      25.9400  4.746648e-12 4.129002e-02   pass
2000   2000   4      0.204      26.2180  4.746648e-12 4.129002e-02   pass
5000   5008   4      2.585      32.2614  2.651185e-11 3.696863e-02   pass
5000   5008   4      2.575      32.3875  2.651185e-11 3.696863e-02   pass
10000  10000  4      18.987     35.1219  9.014595e-11 3.178637e-02   pass
10000  10000  4      18.975     35.1451  9.014595e-11 3.178637e-02   pass

Performance Summary (GFlops)

Size   LDA    Align.  Average  Maximal
1000   1000   4       25.9329  26.0581
2000   2000   4       26.0790  26.2180
5000   5008   4       32.3244  32.3875
10000  10000  4       35.1335  35.1451

Residual checks PASSED
```

600 Samples: I ran the benchmark on floating point and integer instructions and 4 threads for a 10-minute period for each one, and took samples every second on how many instructions per second were achieved during the experiment.

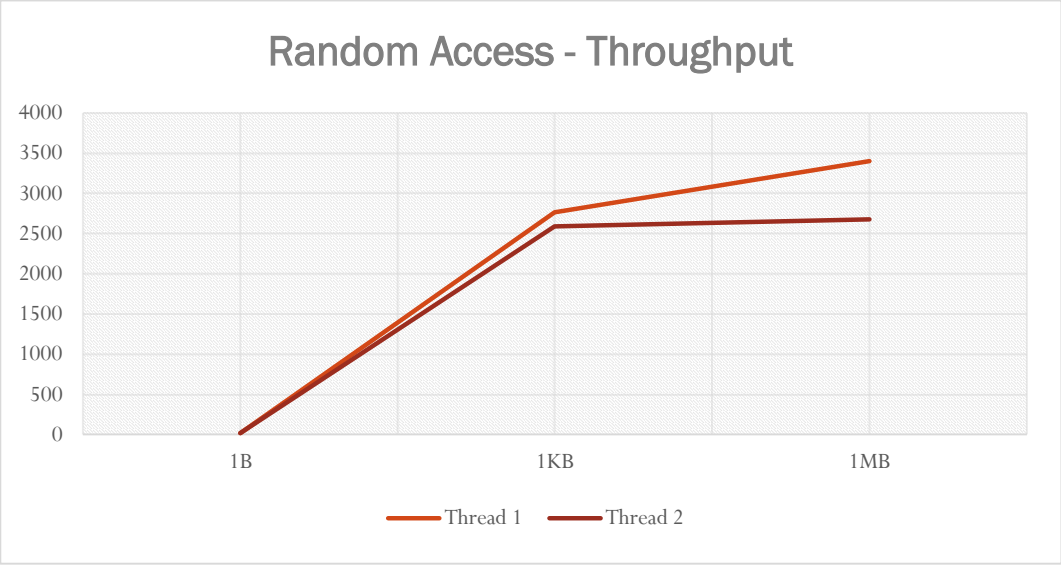


4.2 MEMORY BENCHMARK

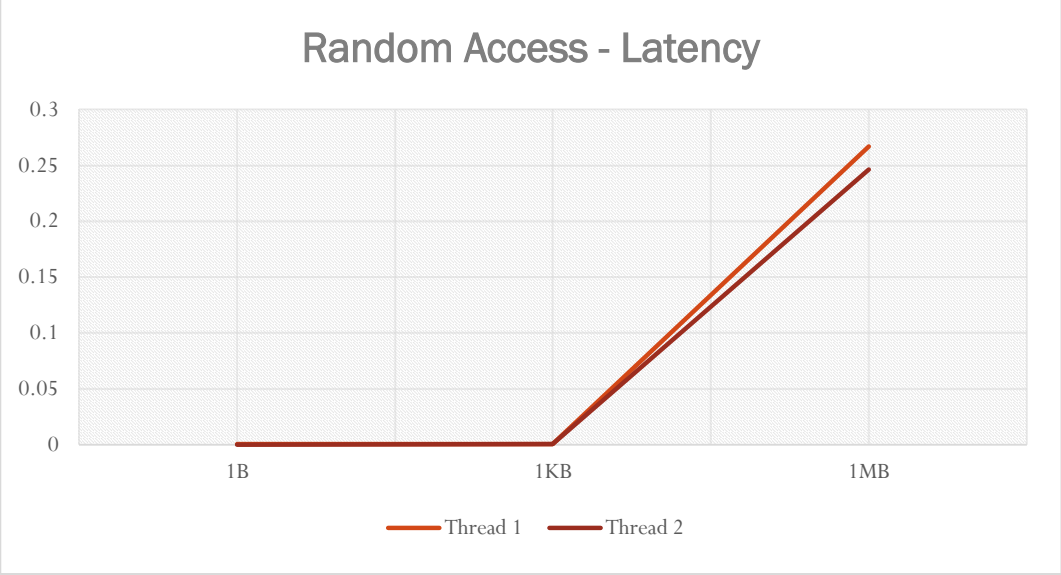
For Memory benchmarking, “memcpy” operation is done on the memory using the code written in C. The memcpy operation copies bytes from one location to another using Random memory access operation and Sequential Memory Access operation. The benchmark executes for different memory block sizes of 1Byte, 1KBye and 1Mbyte.

Below are the results of Random and Sequential Access:

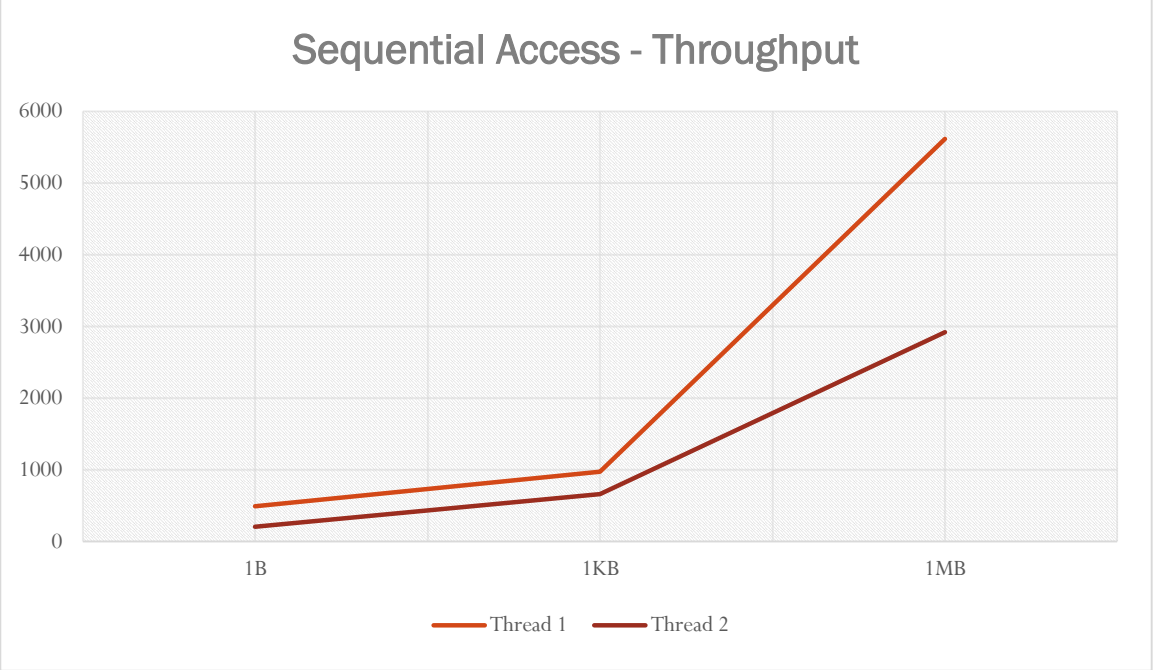
a) **RANDOM ACCESS - Throughput**



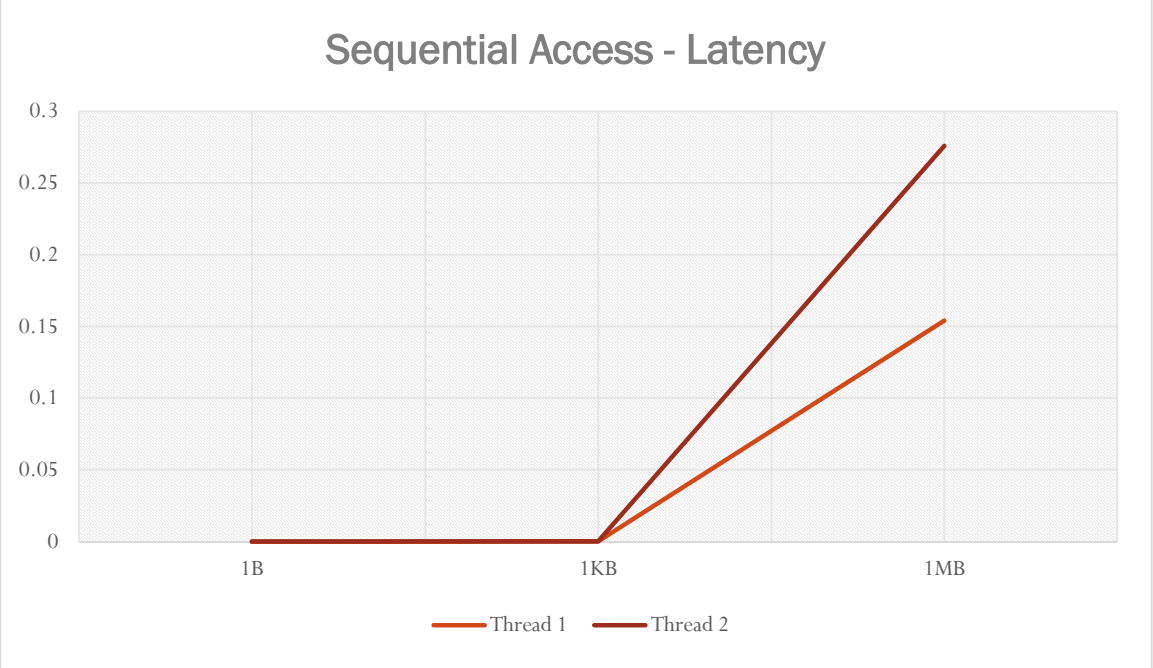
b) **RANDOM ACCESS – Latency**



c) SEQUENTIAL ACCESS - Throughput



d). SEQUENTIAL ACCESS - Latency



RANDOM ACCESS

# of Thread	Block Size	Average Throughput (Mbps)	Average Latency (in MilliSecond)
1	1B	18	0.000018
1	1KB	2766	0.000201
1	1MB	3400	0.246233
2	1B	19	0.000011
2	1KB	2592	0.000402
2	1MB	2680	0.275689

SEQUENTIAL ACCESS

# of Thread	Block Size	Average Throughput (Mbps)	Average Latency (in MilliSecond)
1	1B	193	0.000004
1	1KB	286	0.000132
1	1MB	3927	0.154232
2	1B	122	0.000011
2	1KB	294	0.000235
2	1MB	4371	0.275689

RUNNING STREAM BENCHMARK TOOL

The stream benchmark tool was run in order to compare my own benchmark tool

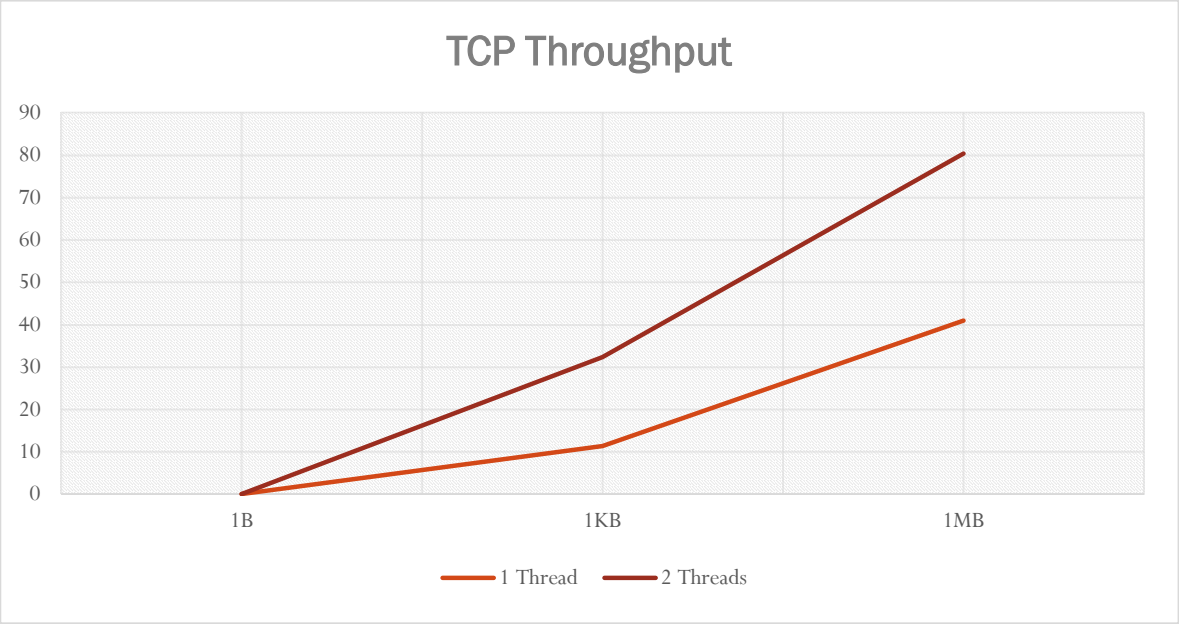
```
[ec2-user@ip-172-31-29-202 cloud]$ ./memory
------------------------------------------------------------------------------
# Thread        Operation       Data Size       Latency(ms)     Throughput
------------------------------------------------------------------------------
Segmentation fault
[ec2-user@ip-172-31-29-202 cloud]$ ./stream
-------------------------------------------------------------
STREAM version $Revision: 5.10 $
-------------------------------------------------------------
This system uses 8 bytes per array element.
-------------------------------------------------------------
Array size = 10000000 (elements), Offset = 0 (elements)
Memory per array = 76.3 MiB (= 0.1 GiB).
Total memory required = 228.9 MiB (= 0.2 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
-------------------------------------------------------------
Your clock granularity/precision appears to be 1 microseconds.
Each test below will take on the order of 14747 microseconds.
   (= 14747 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
-------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           7843.6     0.020648     0.020399     0.020980
Scale:          7712.3     0.021204     0.020746     0.021661
Add:            8878.6     0.027408     0.027031     0.028240
Triad:          8713.3     0.027885     0.027544     0.028187
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
```

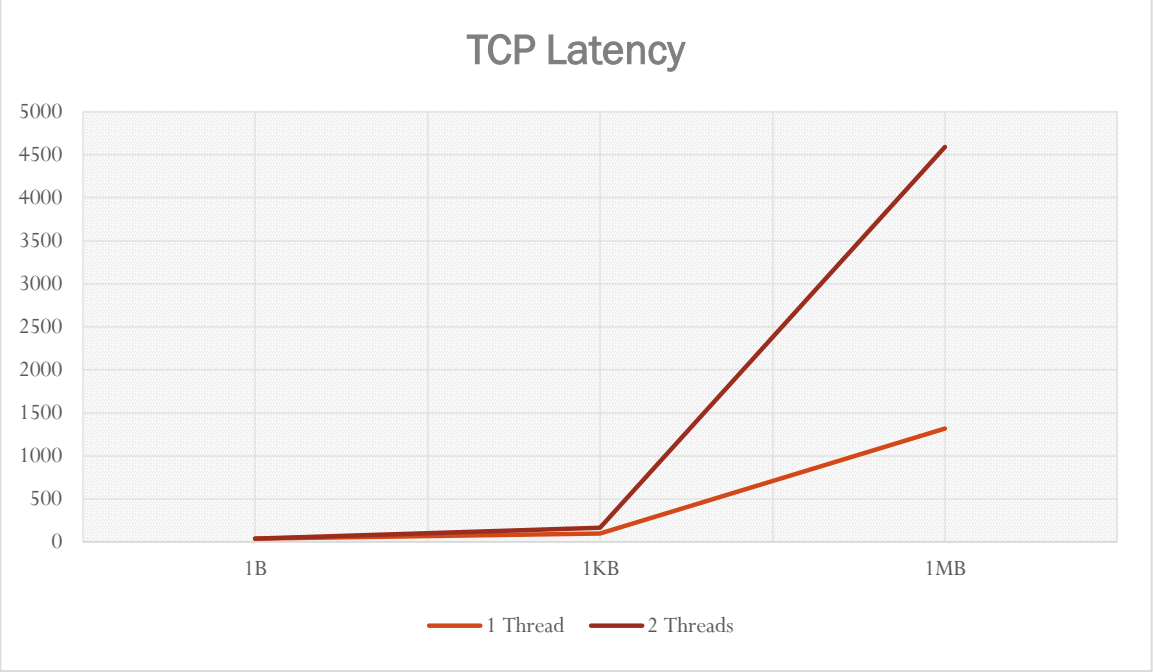
4.3 NETWORK BENCHMARK

This benchmark tool will measure the network speed by running two instances of a Server and a Client and sending data from Client to Server 1000 times and then downloading the same from the Server. And then calculating the total time and hence the total network speed. The tool runs on both UDP and TCPThe results are as follows:

Throughput --> in MB / second
Latency --> in MilliSecond

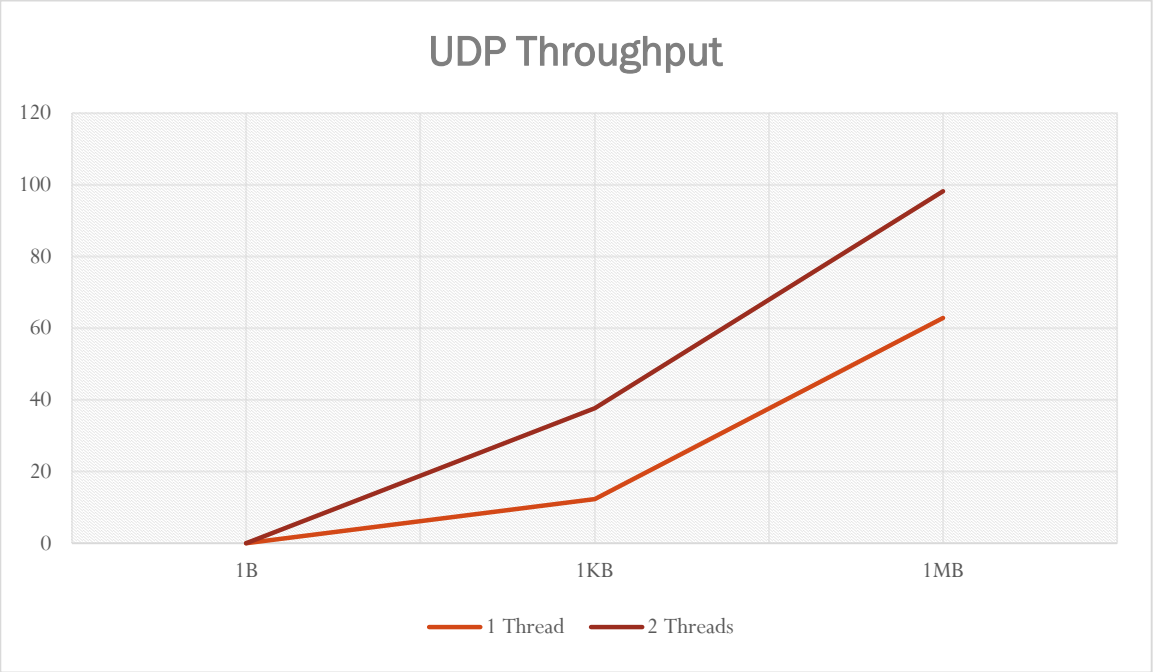
TCP Throughput



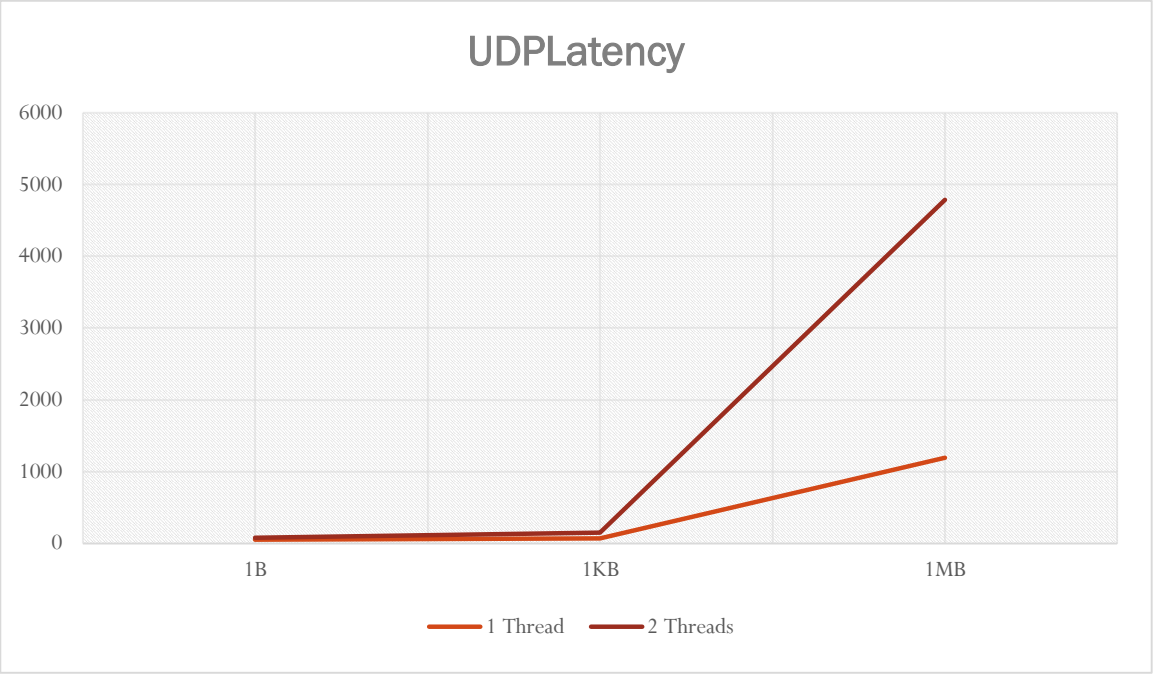


# of Threads	Packet Size	Avg Throughput (MBps)	Latency (ms)
1	1B	0.0015	51
1	1KB	8.547	117
1	1MB	41.955	2908
2	1B	0.0023	48
2	1KB	30.213	133
2	1MB	87.458	5418

UDP Throughput



UDP Latency



# of Threads	Packet Size	Avg. Throughput(Mbps)	Latency (in Millisecond)
1	1B	0.0019	53
1	1KB	12.313	69
1	1MB	62.876	1189
2	1B	0.0036	76
2	1KB	37.65	152
2	1MB	98.222	4789

RUNNING IPERF

```
[ec2-user@ip-172-31-20-140 cloud]$ sudo iperf3 -c 172.31.29.202 -i 1 -t 10 -V -p 80
iperf 3.0.11
Linux ip-172-31-20-140 4.1.10-17.31.amzn1.x86_64 #1 SMP Sat Oct 24 01:31:37 UTC 2015 x86_64 x86_64 x86_64 GNU/Linux
Time: Sun, 14 Feb 2016 03:58:15 GMT
Connecting to host 172.31.29.202, port 80
      Cookie: ip-172-31-20-140.1455422295.471047.1
      TCP MSS: 8949 (default)
[  4] local 172.31.20.140 port 48188 connected to 172.31.29.202 port 80
Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 10 second test
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec   114 MBytes   956 Mbits/sec    5   1.25 MBytes
[  4]   1.00-2.00   sec   110 MBytes   923 Mbits/sec    4    996 KBytes
[  4]   2.00-3.00   sec   110 MBytes   923 Mbits/sec    1   1.12 MBytes
[  4]   3.00-4.00   sec   108 MBytes   902 Mbits/sec    3    891 KBytes
[  4]   4.00-5.00   sec   110 MBytes   923 Mbits/sec    1   1.02 MBytes
[  4]   5.00-6.00   sec   110 MBytes   923 Mbits/sec    1   1.15 MBytes
[  4]   6.00-7.00   sec   109 MBytes   912 Mbits/sec    2    944 KBytes
[  4]   7.00-8.00   sec   109 MBytes   912 Mbits/sec    1   1.05 MBytes
[  4]   8.00-9.00   sec   110 MBytes   923 Mbits/sec    2   1.15 MBytes
[  4]   9.00-10.00  sec   108 MBytes   902 Mbits/sec    3    883 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
Test Complete. Summary Results:
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  1.07 GBytes   920 Mbits/sec   23             sender
[  4]   0.00-10.00  sec  1.07 GBytes   918 Mbits/sec                  receiver
CPU Utilization: local/sender 3.2% (0.1%u/3.3%s), remote/receiver 2.5% (0.3%u/2.2%s)

iperf Done.
[ec2-user@ip-172-31-20-140 cloud]$
```

```
-----------------------------------------------------------
Server listening on 80
-----------------------------------------------------------
Accepted connection from 172.31.20.140, port 48187
[  5] local 172.31.29.202 port 80 connected to 172.31.20.140 port 48188
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-1.00   sec   107 MBytes   896 Mbits/sec
[  5]   1.00-2.00   sec   110 MBytes   921 Mbits/sec
[  5]   2.00-3.00   sec   110 MBytes   925 Mbits/sec
[  5]   3.00-4.00   sec   108 MBytes   905 Mbits/sec
[  5]   4.00-5.00   sec   110 MBytes   921 Mbits/sec
[  5]   5.00-6.00   sec   110 MBytes   920 Mbits/sec
[  5]   6.00-7.00   sec   109 MBytes   917 Mbits/sec
[  5]   7.00-8.00   sec   109 MBytes   915 Mbits/sec
[  5]   8.00-9.00   sec   109 MBytes   914 Mbits/sec
[  5]   9.00-10.00  sec   109 MBytes   910 Mbits/sec
[  5]  10.00-10.04  sec  4.54 MBytes   908 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  5]   0.00-10.04  sec  1.07 GBytes   916 Mbits/sec   23             sender
[  5]   0.00-10.04  sec  1.07 GBytes   914 Mbits/sec                  receiver
-----------------------------------------------------------
```

```
[ec2-user@ip-172-31-20-140 cloud]$ sudo iperf3 -c 172.31.29.202 -p 80 -u -b 100m
Connecting to host 172.31.29.202, port 80
[  4] local 172.31.20.140 port 49212 connected to 172.31.29.202 port 80
[ ID] Interval           Transfer     Bandwidth       Total Datagrams
[  4]   0.00-1.00   sec  10.8 MBytes  90.5 Mbits/sec  1381
[  4]   1.00-2.00   sec  11.9 MBytes   100 Mbits/sec  1526
[  4]   2.00-3.00   sec  11.9 MBytes   100 Mbits/sec  1526
[  4]   3.00-4.00   sec  11.9 MBytes   100 Mbits/sec  1526
[  4]   4.00-5.00   sec  11.9 MBytes   100 Mbits/sec  1526
[  4]   5.00-6.00   sec  11.9 MBytes   100 Mbits/sec  1526
[  4]   6.00-7.00   sec  11.9 MBytes  99.9 Mbits/sec  1525
[  4]   7.00-8.00   sec  11.9 MBytes   100 Mbits/sec  1527
[  4]   8.00-9.00   sec  11.9 MBytes   100 Mbits/sec  1527
[  4]   9.00-10.00  sec  11.9 MBytes  99.9 Mbits/sec  1524
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  4]   0.00-10.00  sec   118 MBytes  99.0 Mbits/sec  0.045 ms  3331/15095 (22%)
[  4] Sent 15095 datagrams

iperf Done.
[ec2-user@ip-172-31-20-140 cloud]$
```

```
[ec2-user@ip-172-31-29-202 iperf-2.0.5]$ sudo iperf3 -s -p 80
-----------------------------------------------------------
Server listening on 80
-----------------------------------------------------------
Accepted connection from 172.31.20.140, port 48191
[  5] local 172.31.29.202 port 80 connected to 172.31.20.140 port 49212
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  5]   0.00-1.00   sec  7.17 MBytes  60.2 Mbits/sec  0.075 ms  463/1381 (34%)
[  5]   1.00-2.00   sec  8.45 MBytes  70.9 Mbits/sec  0.090 ms  444/1526 (29%)
[  5]   2.00-3.00   sec  9.02 MBytes  75.7 Mbits/sec  0.090 ms  371/1526 (24%)
[  5]   3.00-4.00   sec  10.1 MBytes  85.1 Mbits/sec  0.066 ms  227/1526 (15%)
[  5]   4.00-5.00   sec  10.2 MBytes  85.5 Mbits/sec  0.092 ms  221/1526 (14%)
[  5]   5.00-6.00   sec  9.30 MBytes  78.0 Mbits/sec  0.063 ms  336/1526 (22%)
[  5]   6.00-7.00   sec  9.07 MBytes  76.1 Mbits/sec  0.073 ms  364/1525 (24%)
[  5]   7.00-8.00   sec  10.1 MBytes  84.3 Mbits/sec  0.063 ms  240/1527 (16%)
[  5]   8.00-9.00   sec  10.3 MBytes  86.2 Mbits/sec  0.055 ms  212/1527 (14%)
[  5]   9.00-10.00  sec  8.22 MBytes  68.9 Mbits/sec  0.045 ms  453/1505 (30%)
[  5]  10.00-10.04  sec  0.00 Bytes  0.00 bits/sec  0.045 ms  0/0 (-nan%)
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  5]   0.00-10.04  sec   118 MBytes  98.7 Mbits/sec  0.045 ms  3331/15095 (22%)
-----------------------------------------------------------
Server listening on 80
-----------------------------------------------------------
```