Cloud Computing
PA 1
Benchmarking
Design Document and Performance

Name: Sharanheer Choudhari (A20398615)

Design Document

This assignment aims at Benchmarking different parts of the computer system using the virtual cluster running on the chameleon testbed.

CPU Benchmarking

● For CPU Benchmarking, I used strong scaling experiment and design a source code in C language.
● This program will measure the processor speed, in terms of quarter precision, half precision, single precision and double precision floating point operations.(GOPS/Sec)
● This program has different function each consisting of 100 instruction which include addition, subtraction, multiplication operation on char, short, integer and floating point numbers which will run for 1 Trillion times to measure the speed.
● Main aim of this is to utilize the complete CPU cycle by performing diverse types of instructions and measure the speed.
● There is a performance evaluation which evaluates OPS executed per second using different levels of concurrency (1 thread, 2 threads, 4 threads).
● Ran the Linpack benchmark (http://en.wikipedia.org/wiki/LINPACK) and compare the performance achieved using double precision floating point.

Test Results: Performed experiment on Prometheus Cluster for CPU benchmark.

Workload	Concurrency	MyCPUBench Measured Ops/Sec (GigaOPS)	HPL Measured Ops/Sec (GigaOPS)	Theoretical Ops/Sec (GigaOPS)	MyCPUBench Efficiency (%)	HPL Efficiency (%)
QP	1	1.12	N/A	537.6	0.002	N/A
QP	2	2.15	N/A	537.6	0.0039	N/A
QP	4	2.17	N/A	537.6	0.00403	N/A
HP	1	1.16	N/A	537.6	0.0022	N/A
HP	2	2.24	N/A	537.6	0.0042	N/A
HP	4	2.13	N/A	537.6	0.0039	N/A
SP	1	2.11	N/A	537.6	0.0039	N/A
SP	2	3.97	N/A	537.6	0.0074	N/A
SP	4	4.08	N/A	537.6	0.0076	N/A
DP	1	2.06	421.79	537.6	0.0038	78.45
DP	2	4.12	421.79	537.6	0.0076	78.45
DP	4	3.66	421.79	537.6	0.0068	78.45

Theoretical Performance

Based on the hardware specification of the Prometheus Cluster.

FLOPS = Number of Cores * CLOCK SPEED * IPC * Number of CPU's

= 8* 2.1*10^6* 4 * 8 = 537.6 GFLOPS

Linpack Benchmark:

Performed experiment on Hyperion Cluster for CPU benchmark.
I have ran the LINPACK Benchmark on the system to compare the CPU performance and from the below it can seen that the maximal value is obtained at 429.27 GFLOPS which is 79.8% matching with the theoretical performance.

```
schoudhari@hyperionides:~/linpack_11.1.2/benchmarks/linpack$ ./xlinpack_xeon64
Input data or print help ? Type [data]/help :
1000000000
Number of equations to solve (problem size): 10000
Leading dimension of array: 10000
Number of trials to run: 10
Data alignment value (in Kbytes): 10000
Current date/time: Tue Mar 27 21:59:04 2018

CPU frequency:    3.090 GHz
Number of CPUs: 16
Number of cores: 16
Number of threads: 16

Parameters are set to:

Number of tests: 1
Number of equations to solve (problem size) : 10000
Leading dimension of array                  : 10000
Number of trials to run                     : 10
Data alignment value (in Kbytes)            : 10000

Maximum memory requested that can be used=810440000, at the size=10000

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

Size   LDA    Align. Time(s)    GFlops   Residual     Residual(norm) Check
10000  10000  10000    1.607      414.9071 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.570      424.6473 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.563      426.7138 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.589      419.5710 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.596      417.8517 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.615      412.9670 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.553      429.2731 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.557      428.1991 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.602      416.3675 7.750056e-11 2.732748e-02   pass
10000  10000  10000    1.560      427.4160 7.750056e-11 2.732748e-02   pass

Performance Summary (GFlops)

Size   LDA    Align.  Average  Maximal
10000  10000  10000     421.7914 429.2731

Residual checks PASSED

End of tests

schoudhari@hyperionides:~/linpack_11.1.2/benchmarks/linpack$
```

Memory Benchmark:

● For Memory Benchmarking, I used strong scaling experiment and design a source code in C language.
● This program will measure the memory speed of host which will include parameters like read + write operations, sequential read-write access, random read-write access.
● This experiment used different block sizes (1B, 1KB, 1MB, 10MB), and varying the concurrency (1thread, 2 threads and 4 threads).
● We allocate 1 GB of memory (x100) to perform read + write operations in both sequential and random access.
● The program also measures the throughput (Megabytes per second, MB/sec) and latency(microseconds, us). 1B block case used to measure latency, and the 1KB, 1MB, and 10MB cases used to measure throughput.
● Computed the theoretical memory bandwidth of the cluster memory, based on the type of memory and the speed.
● Ran the PMBW benchmark (preinstalled on the cluster) and compared both the performance to see weather benchmarking is running properly.

Test Output: Performed experiment on Prometheus Cluster for memory benchmark.

Throughput Reading

Work-load	Con-currency	Block Size	MyRAMBench Measured Throughput (GB/sec)	pmbw Measured Throughput (GB/sec)	Theoretical Throughput (GB/sec)	MyRAMBench Efficiency (%)	pmbw Efficiency (%)
RWS	1	1KB	3.93	9.8	33.23	11.82	26.72
RWS	1	1MB	3.56	16.86	29.45	12.08	57.25
RWS	1	10MB	4.33	16.88	31.12	13.91	54.24
RWS	2	1KB	6.04	30.7	52.50	11.504	58.47
RWS	2	1MB	4.98	32.17	40.12	12.41	80.18
RWS	2	10MB	5.224	33.09	41.23	12.67	80.25
RWS	4	1KB	6.26	35.6	54.43	11.5	65.40
RWS	4	1MB	5.77	36.46	43.23	13.34	84.33
RWS	4	10MB	6.08	31.89	45.34	13.40	70.33
RWR	1	1KB	2.49	6.7	22.19	11.22	30.19
RWR	1	1MB	3.61	14.34	25.89	13.94	55.38
RWR	1	10MB	3.68	15.23	24.56	14.98	62.01
RWR	2	1KB	1.33	23.45	19.34	5.44	95.98
RWR	2	1MB	5.47	24.43	39.74	13.76	65.47
RWR	2	10MB	5.69	25.43	38.56	14.75	65.94
RWR	4	1KB	1.91	27.89	16.23	6.7	97.96
RWR	4	1MB	5.90	28.19	48.33	12.20	58.32
RWR	4	10MB	6.23	28.79	46.97	13.27	61.29

Latency Reading:

Work-load	Con-currency	Block Size	MyRAMBench Measured Latency (us)	pmbw Measured Latency	Theoretical Latency (us)	MyRAMBench Efficiency (%)	pmbw Efficiency (%)
RWS	1	1B	0.004	0.027	0.031	12.90	87.09
RWS	2	1B	0.0026	0.009	0.014	18.57	64.28
RWS	4	1B	0.002251	0.008	0.012	18.75	66.66
RWR	1	1B	0.13	0.583	0.783	16.60	74.45
RWR	2	1B	0.35	0.562	0.89	39.32	63.15
RWR	4	1B	0.399	0.619	0.93	42.90	66.55

Theoretical Performance:

Throughput = (Data Transfer/Clock) * Clock Frequency * Memory bus width * Number of interfaces

PMBW BENCHMARK:

Below is the screenshot of PMBW benchmarking showing the results of Memory Speeds.
The result below shows the memory speed having a total of 23.54GB/s.

```
schoudhari@compute-3:~$ pmbw -p 2 -s 1024 -S 1024
Running benchmarks with at least 2 threads.
Running benchmarks with array size at least 1024.
Running benchmarks with array size up to 1024.
CPUID: mmx sse avx
Detected 3951 MiB physical RAM and 2 CPUs.

Allocating 2048 MiB for testing.
Running nthreads=2 factor=1073741824 areasize=1024 thrsize=640 testsize=1280 repeats=1677722 testvol=2147484160 testacces
s=268435520
run time = 0.101292 -> rerunning test with repeat factor=15900620294
Running nthreads=2 factor=15900620294 areasize=1024 thrsize=640 testsize=1280 repeats=24844720 testvol=31801241600 testac
cess=3975155200
run time = 1.35052 -> next test with repeat factor=17660564596
RESULT  datetime=2018-03-28 03:23:24    host=compute-3  version=0.6.2   funcname=ScanWrite64PtrSimpleLoop       nthreads=
2       areasize=1024   threadsize=640  testsize=1280   repeats=24844720        testvol=31801241600     testaccess=397515
5200    time=1.3505191789881791919      bandwidth=23547419462.658626556 rate=3.3973998775901356956e-10
Skipping ScanWrite64PtrSimpleLoop test with 2048 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 3072 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 4096 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 6144 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 8192 maximum array size due to -S 1024.
```

```
schoudhari@compute-3:~$ pmbw -p 1 -s 1024 -S 1024
Running benchmarks with at least 1 threads.
Running benchmarks with array size at least 1024.
Running benchmarks with array size up to 1024.
CPUID: mmx sse avx
Detected 3951 MiB physical RAM and 2 CPUs.

Allocating 2048 MiB for testing.
Running nthreads=1 factor=1073741824 areasize=1024 thrsize=1152 testsize=1152 repeats=932068 testvol=1073742336 testacces
s=134217792
run time = 0.065818 -> rerunning test with repeat factor=24470723674
Running nthreads=1 factor=24470723674 areasize=1024 thrsize=1152 testsize=1152 repeats=21241948 testvol=24470724096 testa
ccess=3058840512
run time = 1.57442 -> next test with repeat factor=23314089693
RESULT  datetime=2018-03-28 03:22:19    host=compute-3  version=0.6.2   funcname=ScanWrite64PtrSimpleLoop       nthreads=
1       areasize=1024   threadsize=1152 testsize=1152   repeats=21241948        testvol=24470724096     testaccess=305884
0512    time=1.5744164420175366104      bandwidth=15542726462.283370972 rate=5.1471020991157075161e-10
Skipping ScanWrite64PtrSimpleLoop test with 2048 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 3072 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 4096 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 6144 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 8192 maximum array size due to -S 1024.
Skipping ScanWrite64PtrSimpleLoop test with 12288 maximum array size due to -S 1024.
```

Disk Benchmarking

● For Disk Benchmarking, I used strong scaling experiment and design a source code in C language.
● This program will measure the disk speed of host which will include parameters like read + write operations, sequential read-write access, random read-write access.
● This experiment used different block sizes (1KB, 1MB, 10MB, 100MB), and varying the concurrency (1thread, 2 threads, 4 threads) for throughout and (thread 1, thread 2, thread 4, thread 8, thread 16, thread 32, thread 64, thread 128) for Latency.
● It implements 4 methods, sequential read and write, random read and write.
● Allocated a large piece file of 10GB, and perform read and write operations on either sequential or random access within this 10GB file.
● Ran the IOZone benchmark (http://www.iozone.org/) and compared the performance achieved with theoretical performance.

Test Output: Performed experiment on Hyperion Cluster for disk benchmark.

Throughput Reading

Work-load	Con-currency	Block Size	MyDiskBench Measured Throughput (MB/sec)	IOZone Measured Throughput (MB/	Theoretical Throughput (MB/sec)	MyDiskBench Efficiency (%)	IOZone Efficiency (%)
RS	1	1MB	137.18	341.9	593.37	23.19	57.62
RS	1	10MB	129.115	492.5	685.32	18.84	71.86
RS	1	100M	217.318	783.7	876.31	24.79	89.43
RS	2	1MB	150.07	492.179	657.04	22.84	74.90
RS	2	10MB	132.018	605.7	789.60	16.71	76.71
RS	2	100M	226.553	754.9	891.01	25.42	84.72
RS	4	1MB	162.86	847.23	1039.54	15.66	81.50
RS	4	10MB	197.18	646.6	998.42	19.74	64.76
RS	4	100M	461.906	817.03	1612.3	28.64	50.67
WS	1	1MB	43.91	196.6	302.49	14.51	64.99
WS	1	10MB	60.22	213.4	450.65	23.36	47.35
WS	1	100M	116.74	281.5	527.15	22.15	53.40
WS	2	1MB	44.04	289.86	350.3	12.57	82.74
WS	2	10MB	59.365	223.3	409.71	14.48	54.50
WS	2	100M	118.36	580.65	711.13	16.64	81.65
WS	4	1MB	43.67	355.9	432.21	10.10	82.34
WS	4	10MB	59.39	269.6	509.91	11.65	52.87
WS	4	100M	122.84	511.16	761.38	16.14	67.14
RR	1	1MB	558.83	247.9	357.54	30.57	69.33
RR	1	10MB	292.43	379.05	504.67	57.94	75.11
RR	1	100M	333.73	587.04	891.23	37.45	65.86
RR	2	1MB	181.363	422.7	534.32	33.94	79.10
RR	2	10MB	170.04	608.6	823.41	20.65	73.91
RR	2	100M	220.73	727.15	998.03	22.11	72.85
RR	4	1MB	172.79	778.9	937.40	18.43	83.09
RR	4	10MB	177.84	854.43	1120.32	15.87	76.26

RR	4	100M	172.85	920.63	1398.30	12.36	65.84
WR	1	1MB	51.15	267.4	398.09	12.84	67.17
WR	1	10MB	168.71	273.6	546.26	30.88	50.08
WR	1	100M	192.311	790.08	901.25	21.34	87.66
WR	2	1MB	48.140	292.3	402	11.97	72.71
WR	2	10MB	157.61	444.5	679.30	23.20	65.43
WR	2	100M	153.392	833.8	1023.43	14.98	81.47
WR	4	1MB	46.598	294.5	309.21	15.07	95.24
WR	4	10MB	141.42	417.23	661.23	21.38	63.09
WR	4	100M	132.754	510.21	729.41	18.20	69.94

Latency Reading

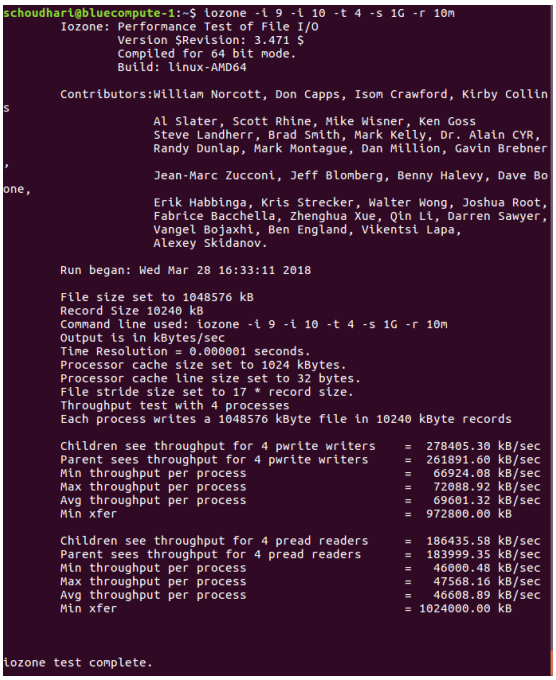
Work-load	Con-currency	Block Size	MyDiskBench Measured Latency (ms)	IOZone Measured Latency	Theoretical Latency (ms)	MyDiskBench Efficiency (%)	IOZone Efficiency (%)
RR	1	1KB	0.456	0.956	1.9	24	50.31
RR	2	1KB	0.165	0.665	0.83	19.87	80.12
RR	4	1KB	0.0667	0.367	0.55	12.12	66.72
RR	8	1KB	0.026	0.096	0.174	14.97	55.17
RR	16	1KB	0.001251	0.0091	0.0121	10.33	75.21
RR	32	1KB	0.00104	0.00804	0.0083	12.53	96.86
RR	64	1KB	0.0142	0.042	0.0787	18.04	53.36
RR	128	1KB	0.01176	0.04176	0.054	21.7	77.33
WR	1	1KB	0.5913	1.913	2.4	24.63	79.70
WR	2	1KB	0.630	1.63	2.3	27.39	70.86
WR	4	1KB	0.6705	1.4605	1.65	40.63	88.51
WR	8	1KB	0.6634	0.934	1.15	57.68	81.21
WR	16	1KB	0.662	1.462	2.23	29.68	65.56
WR	32	1KB	0.602	1.602	2.71	22.21	59.11
WR	64	1KB	0.577	1.577	1.69	34.14	93.31
WR	128	1KB	0.64	1.64	2.12	30.18	77.35

IOPS Reading

Work-load	Con-currency	Block Size	MyDiskBench Measured IOPS	IOZone Measured IOPS	Theoretical IOPS	MyDiskBench Efficiency (%)	IOZone Efficiency (%)
RR	1	1KB	2192.89	7192.89	8112	27.03	88.66
RR	2	1KB	6031.45	9031.45	16215	37.19	55.69
RR	4	1KB	14975.083	24975.083	35016.32	42.76	71.32
RR	8	1KB	38112.04	48112.04	73084	52.14	65.83
RR	16	1KB	799469.37	899469.37	921554.93	86.75	97.61
RR	32	1KB	958307.215	1258307.215	1794210.10	53.41	70.13
RR	64	1KB	69971.151	149971.151	201923.2	34.65	74.27
RR	128	1KB	84987.869	284987.869	326751.2	26.01	87.21
WR	1	1KB	1690.97	6690.97	8262.1	20.46	80.98
WR	2	1KB	1586.23	7586.23	9012.5	17.60	84.17
WR	4	1KB	1491.38	8491.38	9243.1	16.13	91.86
WR	8	1KB	1507.219	7507.219	8200.1	18.38	91.55
WR	16	1KB	1510.23	7510.23	7832	19.28	95.89
WR	32	1KB	1660.2	7660.2	9271	17.907	82.62
WR	64	1KB	1732.4	6732.4	9769	17.73	68.91
WR	128	1KB	1539.35	5539.35	7865	19.57	70.43

IoZone Benchmarking:

The following screenshot shows the performance metrics taken from IoZone benchmarking ran on the system.
The speeds are measured in terms of Kilobytes. The image shows the speeds for pread & pwrite which runs for block size of 10MB and file size 1GB.



Network Benchmarking

● For Network Benchmarking, I used strong scaling experiment and design a source code in C language.
● It will measure the network speed over the between 2 processes on the different nodes.
● It includes the TCP protocol stack, UDP, varying packet/buffer size (1KB, 32KB, 1Byte), and varying the concurrency (1thread, 2 threads, 4 threads, 8 threads) on both server and client side.
● It will measure the throughput (Megabits per second, Mb/sec) and latency (ms).
● We used fixed amount of data (1GB x100 times) to transfer using socket programming and multithreading.
● Ran the IPerf benchmark (http://en.wikipedia.org/wiki/Iperf) and compared achieved throughput performance over the cluster network with theoretical values.

Test Reading Performed experiment on Hyperion Cluster for network benchmark.

Throughput Reading

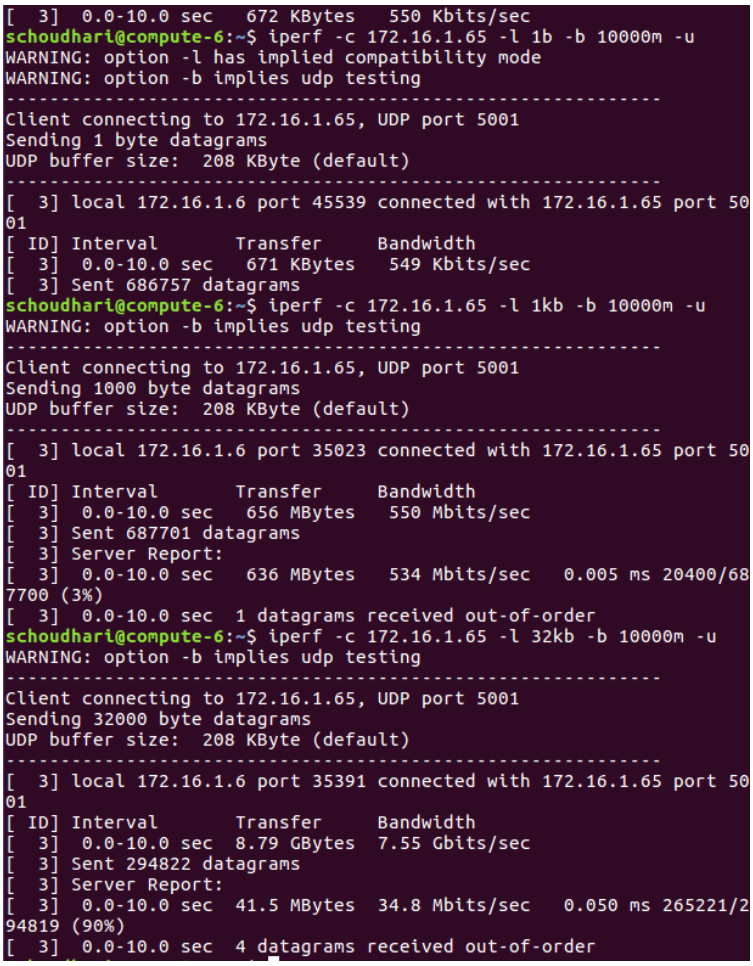
Protocol	Concurrency	Block Size	MyNETBench Measured Throughput (Mb/sec)	iperf Measured Throughput (Mb/sec)	Theoretical Throughput (Mb/sec)	MyNETBench Efficiency (%)	iperf Efficiency (%)
TCP	1	1KB	172.041	551	593.37	28.99	92.85
TCP	1	32KB	171.956	623.4	685.32	25.09	90.96
TCP	2	1KB	413.15	651	976.31	42.31	66.67
TCP	2	32KB	207.835	721.3	657.04	27.45	95.27
TCP	4	1KB	386.49	739.3	989.6	39.05	74.70
TCP	4	32KB	389.328	821.67	891.01	43.69	92.21
TCP	8	1KB	521.817	904.3	1239.54	42.09	72.95
TCP	8	32KB	531.48	839	998.42	53.23	84.03
UDP	1	1KB	157.824	720.1	912.3	17.29	78.93
UDP	1	32KB	603.232	1387	1602.49	37.64	86.55
UDP	2	1KB	184.512	834	950.65	19.4	87.73
UDP	2	32KB	744.18	1550	1927.15	38.61	80.42
UDP	4	1KB	177.77	893	950.3	18.7	93.97
UDP	4	32KB	712.812	1250	1309.71	54.42	95.44
UDP	8	1KB	156.612	851	911.13	17.18	93.4
UDP	8	32KB	987	1943	2332.21	42.32	83.21

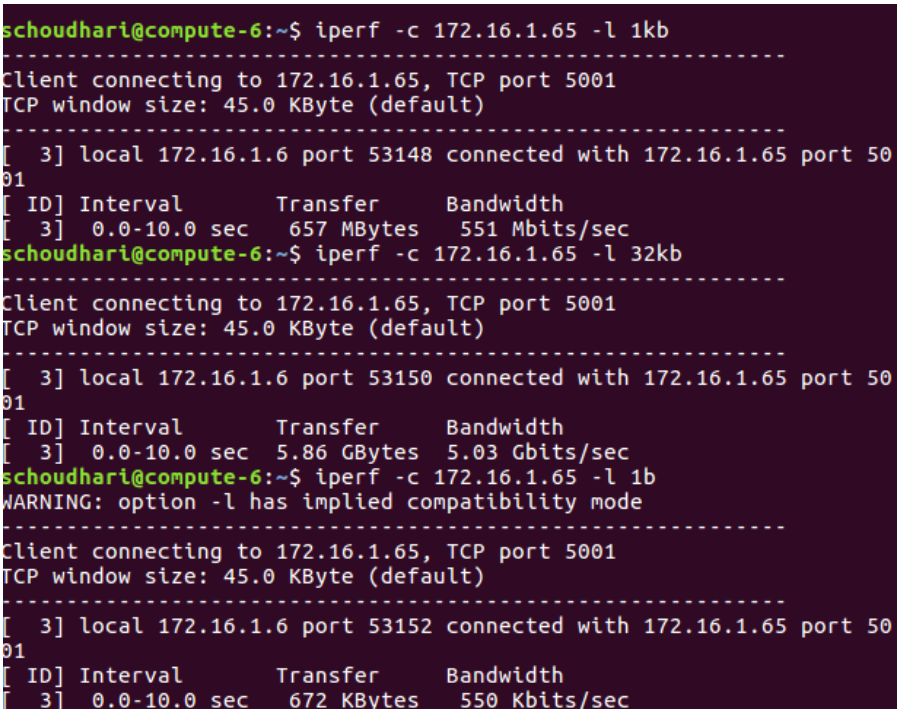
Latency Reading

Protocol	Concurrency	Message Size	MyNETBench Measured Latency (ms)	ping Measured Latency (ms)	Theoretical Latency (ms)	MyNETBench Efficiency (%)	iperf Efficiency (%)
TCP	1	1B	0.110705	0.55	0.63	17.57	87.3
TCP	2	1B	0.058607	0.35	0.36	16.27	97.22
TCP	4	1B	0.031419	0.12	0.21	14.96	57.14
TCP	8	1B	0.019337	0.155	0.174	11.11	89.08
UDP	1	1B	0.1092	0.51	0.821	13.30	62.11
UDP	2	1B	0.0554	0.32	0.43	12.88	74.41
UDP	4	1B	0.0308	0.21	0.287	10.73	73.17
UDP	8	1B	0.02152	0.11	0.154	13.97	71.42

IPERF Benchmarking for TCP and UDP:

The following screenshot shows the performance metrics taken from IPerf benchmarking ran on the system.
The speeds are measured in terms of Kilobytes/sec. The first image shows the bandwidth/ throughput for UDP and the second image shows the throughput for TCP protocol.





Conclusion:

Performed benchmarking on different computer parts including CPU, RAM(memory), Disk and Network. Implemented all the benchmark using strong scaling experiment written in C language and evaluated the efficiency, by comparing the output with the values obtained from the benchmark tool and calculated theoretical values.

References:

http://man7.org/linux/man-pages/man2/connect.2.html
http://www2.cs.uidaho.edu/~krings/CS270/Notes.S10/270-F10-28.pdf
https://www.programiz.com/c-programming/c-dynamic-memory-allocation
https://stackoverflow.com/questions/15731924/udp-multi-client-server-basics