Operating System
Lab02

Palak Jain 130050031
Rawal Khirodkar 130050014

Q1.

Setup:
Two linux laptops connected over LAN.

Server side information:
Sector size = 512 bytes (command: lsblk -o NAME,PHY-SeC)
CPU cores = 2 (command: less /proc/cpuinfo)
cpu MHz : 1547.753
cache size : 3072 KB

Client side Information:
CPU cores: 2
cpu MHz : 808.613
cache size : 2048 KB

Caches were dropped from server side using command :

```
sudo echo 3|sudo tee /proc/sys/vm/drop_caches
```

a. Disk Read Bandwidth:

Command : `iostat -d 1 -m`

Observations (disk program over 10000 files)

Device:	tps	MB_read/s	MB_wrtn/s	MB_read	MB_wrtn
sda	441.00	46.39	0.00	46	0
sda	460.00	48.34	0.00	48	0
sda	376.00	39.64	0.51	39	0
sda	456.00	48.00	0.00	48	0
sda	366.00	36.57	0.00	36	0
sda	399.00	41.30	0.00	41	0
sda	465.00	48.75	0.00	48	0

Avg MB_read/s : 43.7 MB/s
Request/s : 43.7 MB/s /2(MB) = **21.8 reqs/sec**

b. Network Bandwidth

Command : `iperf`

Observations: iperf output (server side):

```
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
local 10.4.128.102 port 5001 connected with 10.4.128.156 port 45600
[ ID] Interval       Transfer     Bandwidth
 0.0-10.0 sec   111 MBytes  92.5 Mbits/sec
```

Requests/sec : 92.5 /(2*8)= **5.78 reqs/sec**

Definitions with respect to this experiment:

- **Response Time (Latency) measures the end-to-end time processing time. It is calculated in this experiment by measuring the time between sending a request and receiving the response for server. Latency is measured from the client machine and includes the network overhead as well. (Note: think time excluded)**
- **Throughput measures the amount of file requests that a server processes during a specific time interval (e.g. per second here). Throughput is calculated by measuring the time taken to processes a set of requests and then using the following equation.**

Throughput = number of completed requests / time to complete the requests

Please refer to graphs below

Q2. **Case 1**: Sleep time : 0s
Total time: 120s
Mode: Random file access

No memory leak (top command's output client side after running for 3 mins):

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
10373	rawal	20	0	22916	1608	1496	S	12.0	0.0	0:08.28	client

a. **Optimal value of Number of Threads (N) : 2**
Explanation:
This is inferred, since, the throughput increases sharply from 1 to 2. However, after that it remains nearly constant i.e. at saturation level for N upto 20 and more.

b. Throughput, for lower values of N (in this case N=1) than the optimal values, the throughput is expected to increase, since the server is not saturated.
For higher values of N, the throughput has already reached saturation, so it stays steady at the same value.

Response time (latency) increases with number of client requests (concurrency) the server has to respond to. In this case it is majorly due to effects of network traffic as disk is not the bottleneck. More concurrency results in bigger network packet queues at both server side and client side (Note: all client requests are deployed by a single system). Also the server side creates more number of processes to cater to client requests, this increases load on server side system. This explains the increasing trend for response time.

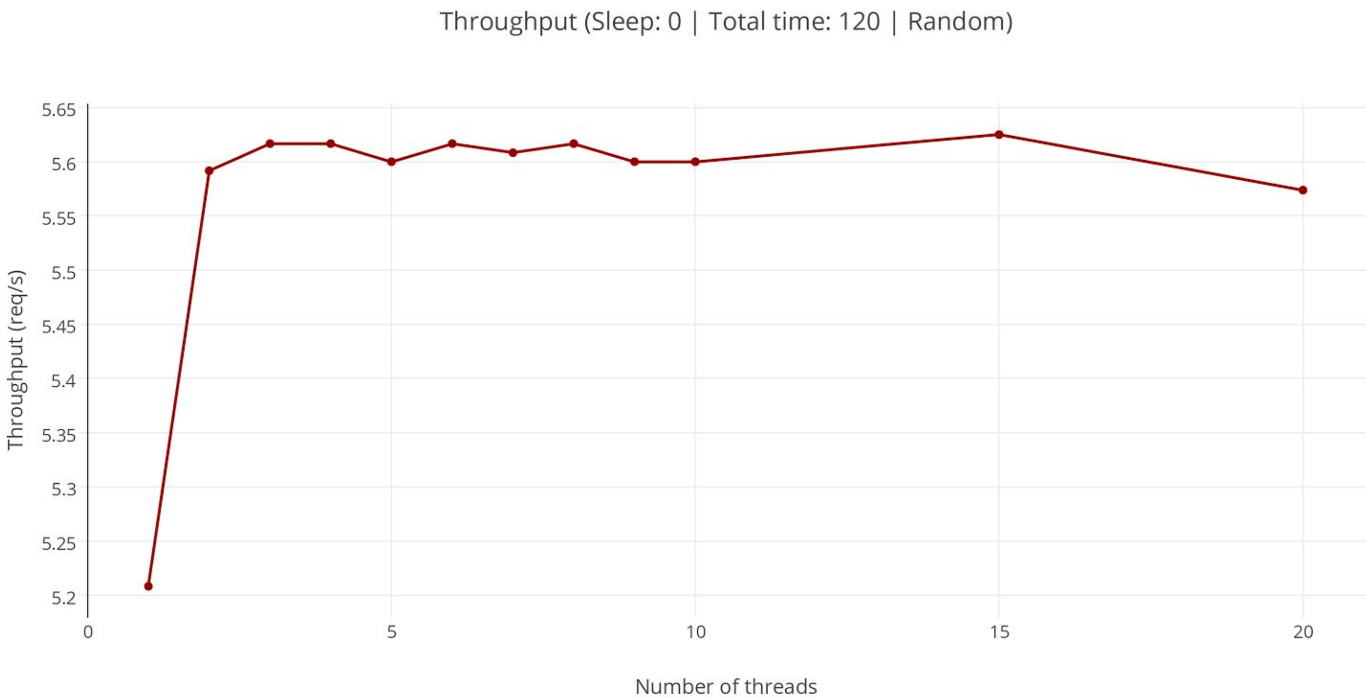
c. Bottleneck resource: **Network Bandwidth**
To identify the bottleneck , we monitored both disk usage and network bandwidth. The network bandwidth was saturated to its peak value (identified in part 1) while disk usage seemed far from saturation point.

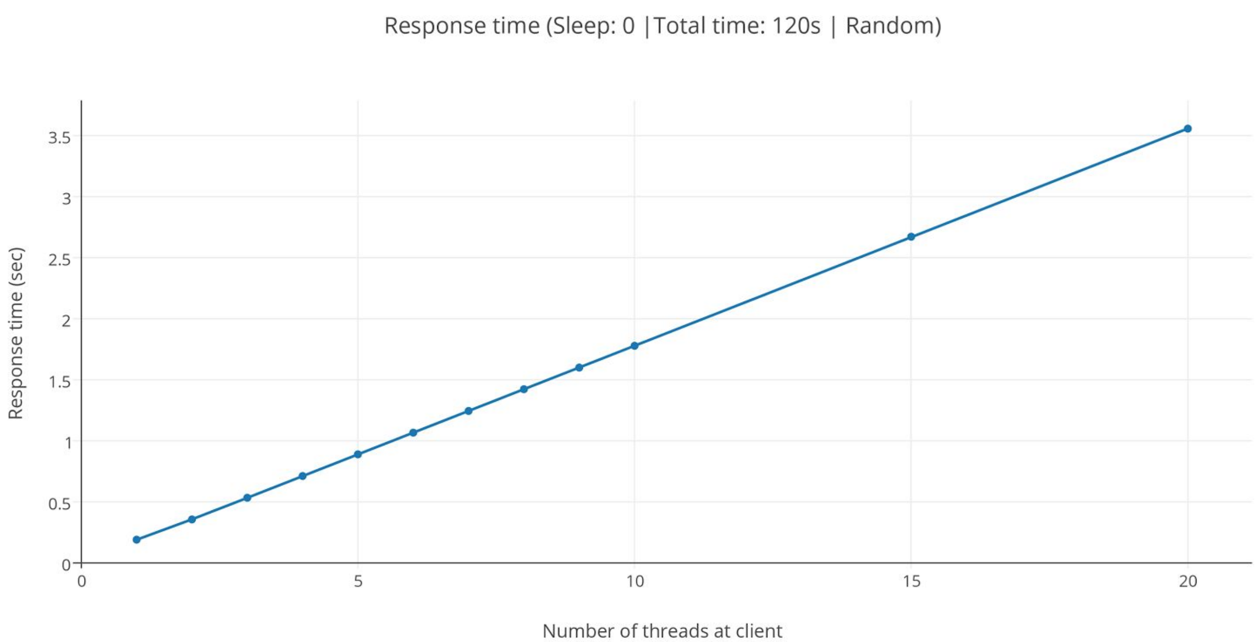
Command: iostat -d 1 -m (output from server side at optimal N = 2)

Device:	tps	MB_read/s	MB_wrtn/s	MB_read	MB_wrtn
sda	268.00	25.46	0.00	25	0
sda	226.00	23.21	0.04	23	0
sda	365.00	32.20	0.27	32	0

The average disk usage noticed here is way below the benchmark disk bandwidth noticed that is 43.7 MB_read/s
Clearly the disk usage is not the bottleneck.

d. Server Saturation Throughput ~ **5.6 req/s**
The network bandwidth is ~ 5.78 req/s which upper bounds the throughput. At saturation level, the throughput tends to approach the bandwidth value.





Q3) **Case 2** : Sleep: 1s
Total Time: 120s
Mode: Random

a. **Optimal N :** 10

Since, sleep time is increased to 1s, the client program does not stress the server as much as before for any given value of N. Thus, completely loading the server to achieve the saturation throughput takes more number of clients in this case.

b. Throughput and Response time shows similar trends as in q2. However, the saturation is attained for higher values of N as explained above.

c. **Bottleneck resource:** Network Bandwidth

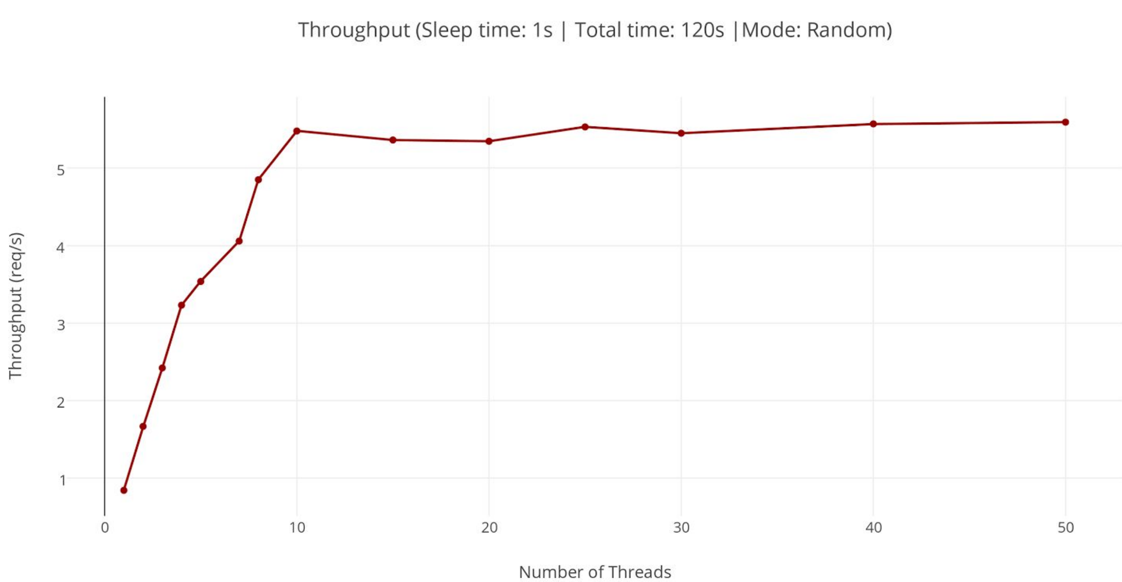
iostat output server side (at optimal N=10)

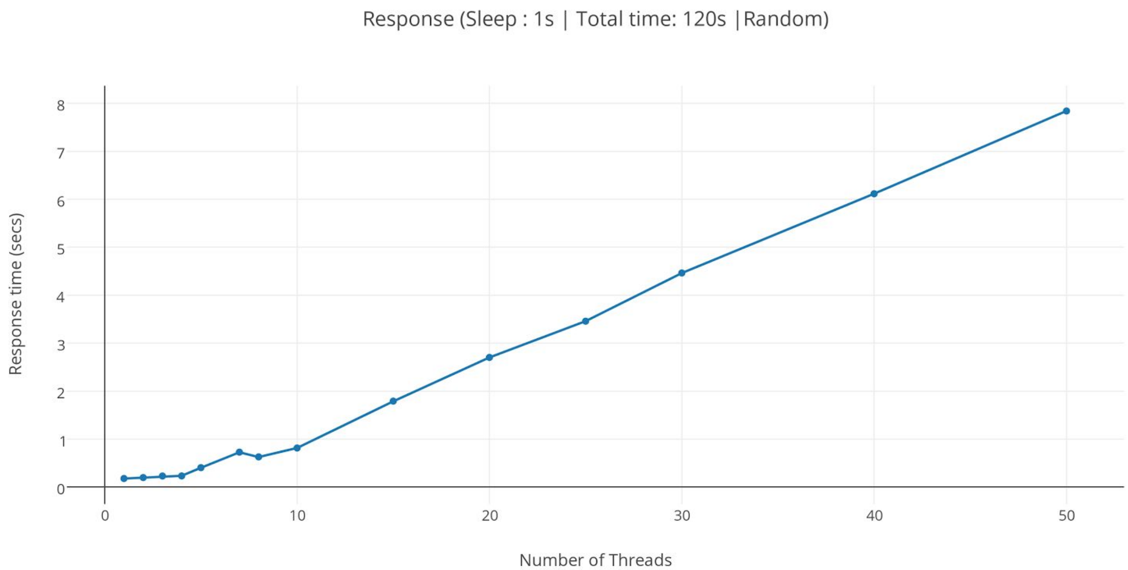
Device:	tps	MB_read/s	MB_wrtn/s	MB_read	MB_wrtn
sda	218.00	21.26	0.00	21	0
sda	202.00	17.94	3.64	17	3
sda	179.00	16.54	0.00	16	0

The network throughput noticed at optimal N (N=10) is far less than avg disk bandwidth noticed.

d. Saturation throughput ~5.6 req/s

The number is justified and is close to the upper bound 5.78 req/s





Q4. Case 3:

Sleep time: 0s
Total time: 120s
Fixed

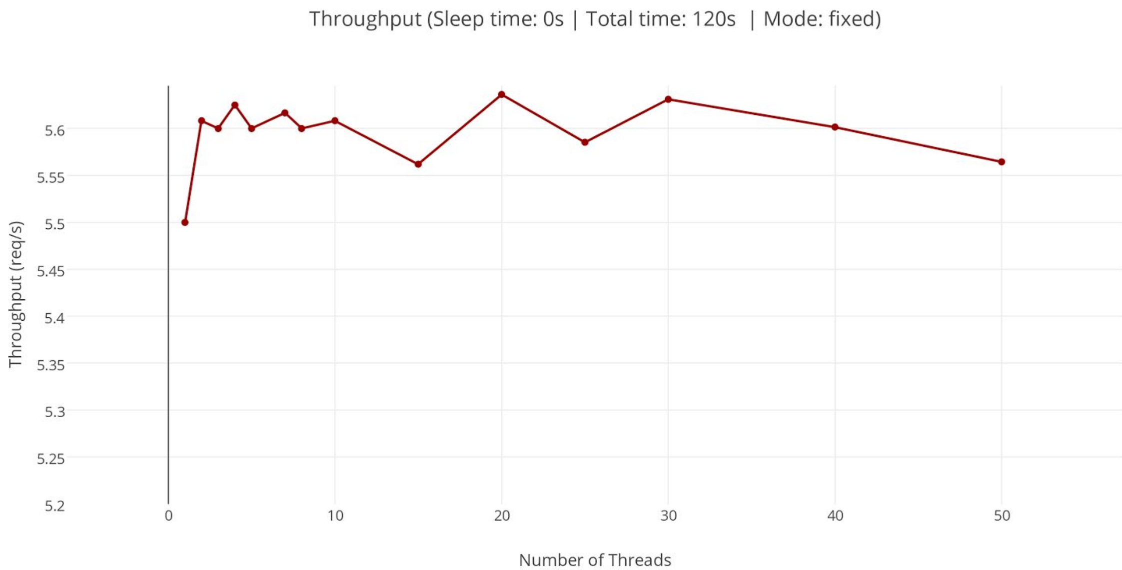
Fixed mode only affects the disk bandwidth. Since, fixed files are read, they are already present in the cache after the first access vis-a-vis random access wherein almost each file has to be loaded from the disk. Thus fixed mode reduces disk load. Thus, in case where disk bandwidth is a bottleneck, throughput could increase substantially.

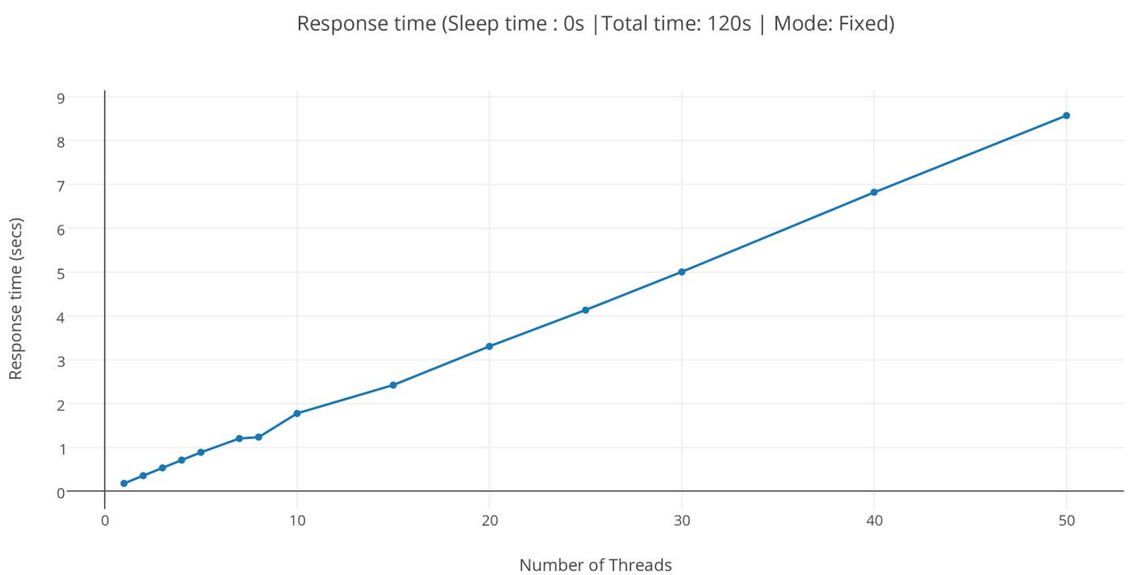
However, in our case, network get saturated much before disk bandwidth even in random mode, which has higher disk load. Thus, in fixed mode, reduction in disk accesses doesn't influence our results.
Results obtained for part a), b), c) are similar to case 1 (Q2)
Optimal N = 1, the trends and bottleneck resources are similar to Q2.

Again iostat output agrees with network being the bottleneck at optimal N = 1.

Device:	tps	MB_read/s	MB_wrtn/s	MB_read	MB_wrtn
sda	143.00	1.60	0.05	1	0
sda	99.00	1.91	0.03	1	0
sda	3.00	0.01	0.00	0	0

Notice the very less amount of disk access due to fixed mode. The network throughput is closer to the benchmark network bandwidth, hence it being the bottleneck is justified.


Throughput (Sleep time: 0s | Total time: 120s | Mode: fixed)
Throughput (req/s)
5.6
5.55
5.5
5.45
5.4
5.35
5.3
5.25
5.2
0
10
20
30
40
50
Number of Threads


Response time (Sleep time : 0s |Total time: 120s | Mode: Fixed)
Response time (secs)
9
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50
Number of Threads