

HW1: System vs OS Virtualization

Configurations for the experimental setup:

CPU : 2.6 GHz 6 -core intel core i7

Memory: 32 GB 2667 MHz DDR4

OS: mac OS Monterey version 12.3.1

Disk space: 120 GB



Note: Borrowed a mac for this assignment as windows is not recommended. Terminal screenshots will show a different userid.

Installation of QEMU and enabling QEMU VM:

Installed QEMU using homebrew and the following command:

```
$ brew install qemu
```

Followed the homework document to create QEMU image and to install the VM:

```
$ sudo qemu-img create ubuntu.img 10G -f qcow2
```

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -cdrom  
/Users/sbajoria/downloads/ubuntu-20.04.4-live-server-amd64.iso -m 2046  
-boot strict=on
```

Installation of Docker and enabling Docker container

Installed the docker desktop using the link:

<https://docs.docker.com/desktop/mac/install/>

Installed the docker image zyclonite/sysbench using the command:

```
docker pull zyclonite/sysbench
```

```
LUSC02CX8V0MD6V:~ sbajoria$ docker pull zyclonite/sysbench  
Using default tag: latest  
latest: Pulling from zyclonite/sysbench  
Digest: sha256:016020c3b53c7e65cdb58e7d4a98afd14f8a3e2f5781cf4c368596b2e448602b  
Status: Image is up to date for zyclonite/sysbench:latest  
docker.io/zyclonite/sysbench:latest  
LUSC02CX8V0MD6V:~ sbajoria$
```

Note: Terminal id shows sbajoria, since it's a borrowed laptop

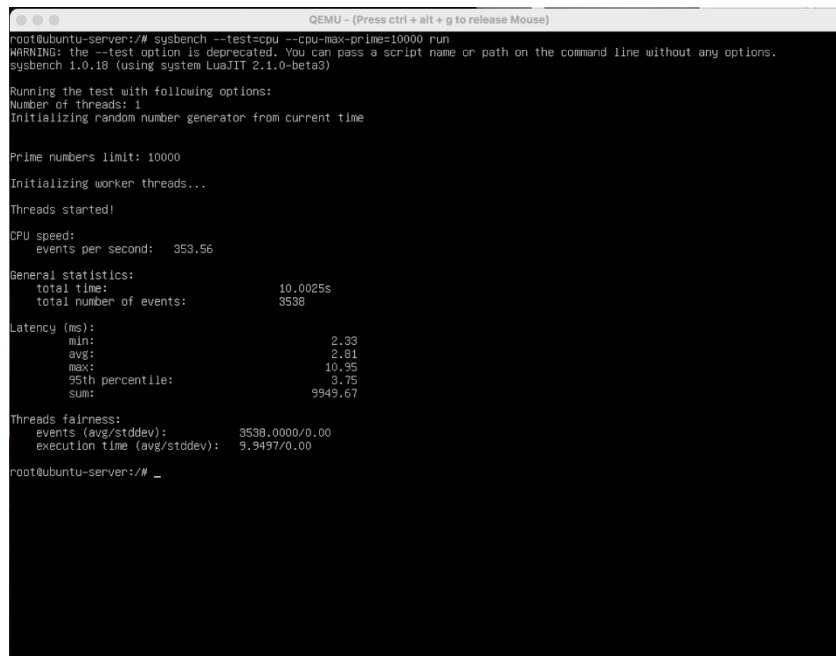
Screenshots of the experiments

CPU test:

Ran different prime numbers on both QEMU and Docker. Results of all three tests are added as screenshots.

QEMU:

Test 1: `sysbench --test=cpu --cpu-max-prime=10000 run`



```
QEMU - (Press ctrl + alt + g to release Mouse)
root@ubuntu-server:/# sysbench --test=cpu --cpu-max-prime=10000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 10000
Initializing worker threads...
Threads started!

CPU speed:
  events per second:   353.56

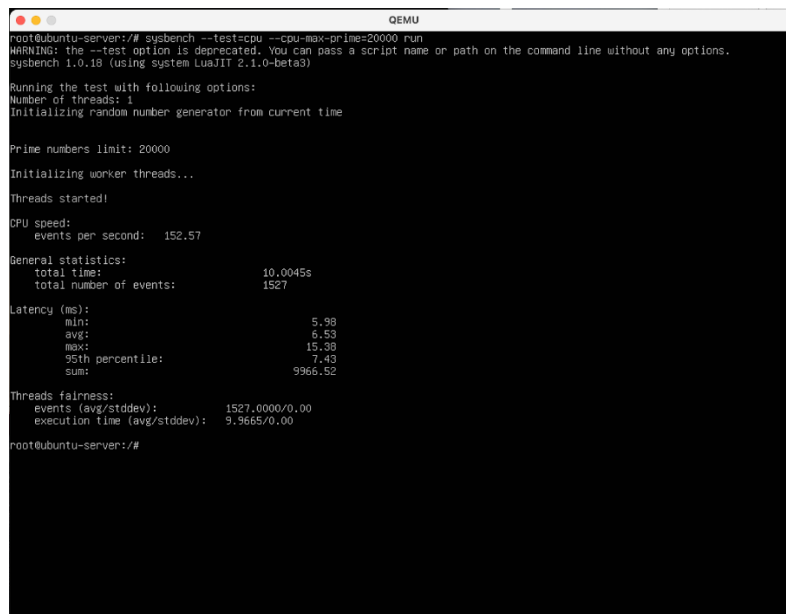
General statistics:
  total time:          10.0025s
  total number of events: 3538

Latency (ms):
  min:                 2.33
  avg:                 2.81
  max:                 10.95
  95th percentile:    3.75
  sum:                 9949.67

Threads fairness:
  events (avg/stddev):  3538.0000/0.00
  execution time (avg/stddev): 9.9497/0.00

root@ubuntu-server:/# _
```

Test 2: `sysbench --test=cpu --cpu-max-prime=20000 run`



```
QEMU
root@ubuntu-server:/# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 20000
Initializing worker threads...
Threads started!

CPU speed:
  events per second:   152.57

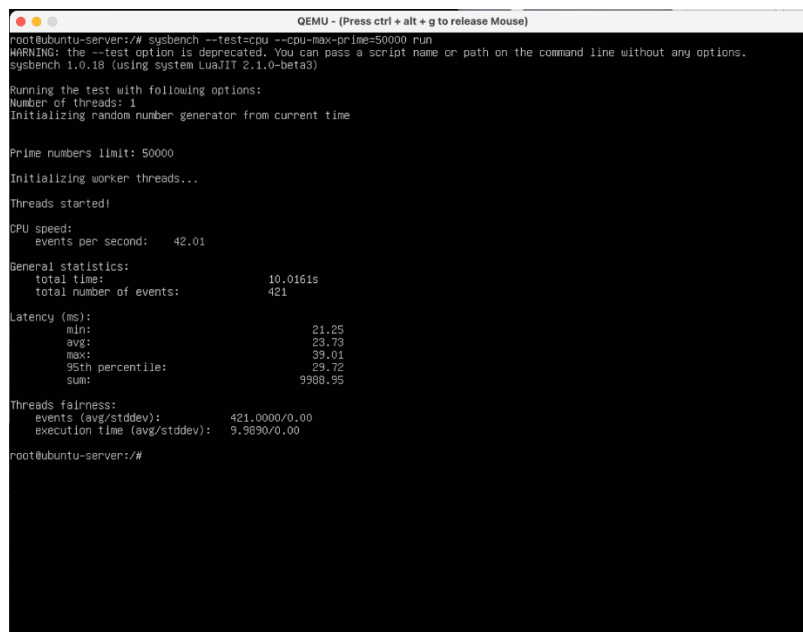
General statistics:
  total time:          10.0045s
  total number of events: 1527

Latency (ms):
  min:                 5.98
  avg:                 6.53
  max:                 15.38
  95th percentile:    7.43
  sum:                 9966.52

Threads fairness:
  events (avg/stddev):  1527.0000/0.00
  execution time (avg/stddev): 9.9665/0.00

root@ubuntu-server:/#
```

Test 3: `sysbench --test=cpu --cpu-max-prime=50000 run`



```
QEMU - (Press ctrl + alt + g to release Mouse)
root@ubuntu-server:~# sysbench --test=cpu --cpu-max-prime=50000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.19 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 50000
Initializing worker threads...
Threads started!

CPU speed:
  events per second:   42.01

General statistics:
  total time:          10.0161s
  total number of events: 421

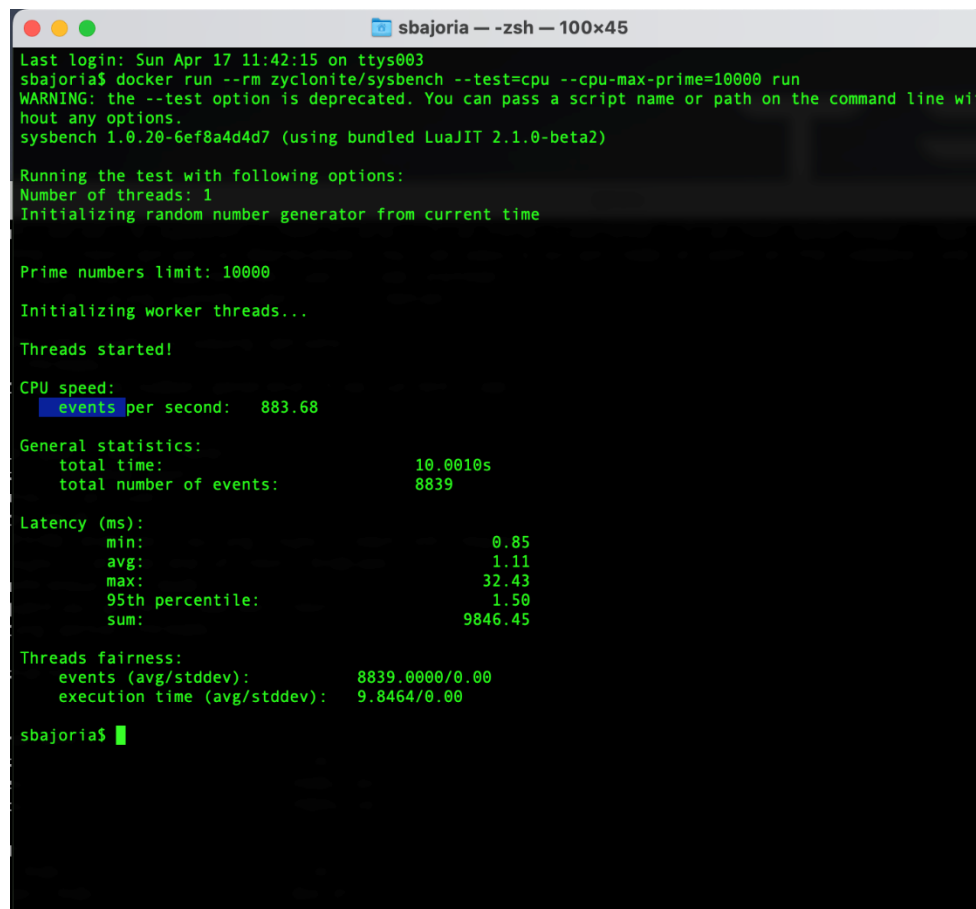
Latency (ms):
  min:                 21.25
  avg:                 23.72
  max:                 39.01
  95th percentile:    29.72
  sum:                 9988.95

Threads fairness:
  events (avg/stddev): 421.0000/0.00
  execution time (avg/stddev): 9.9890/0.00

root@ubuntu-server:~#
```

Docker:

Test 1: `docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=10000 run`



```
sbajoria -- -zsh -- 100x45
Last login: Sun Apr 17 11:42:15 on ttys003
sbajoria$ docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=10000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 10000
Initializing worker threads...
Threads started!

CPU speed:
  events per second:   883.68

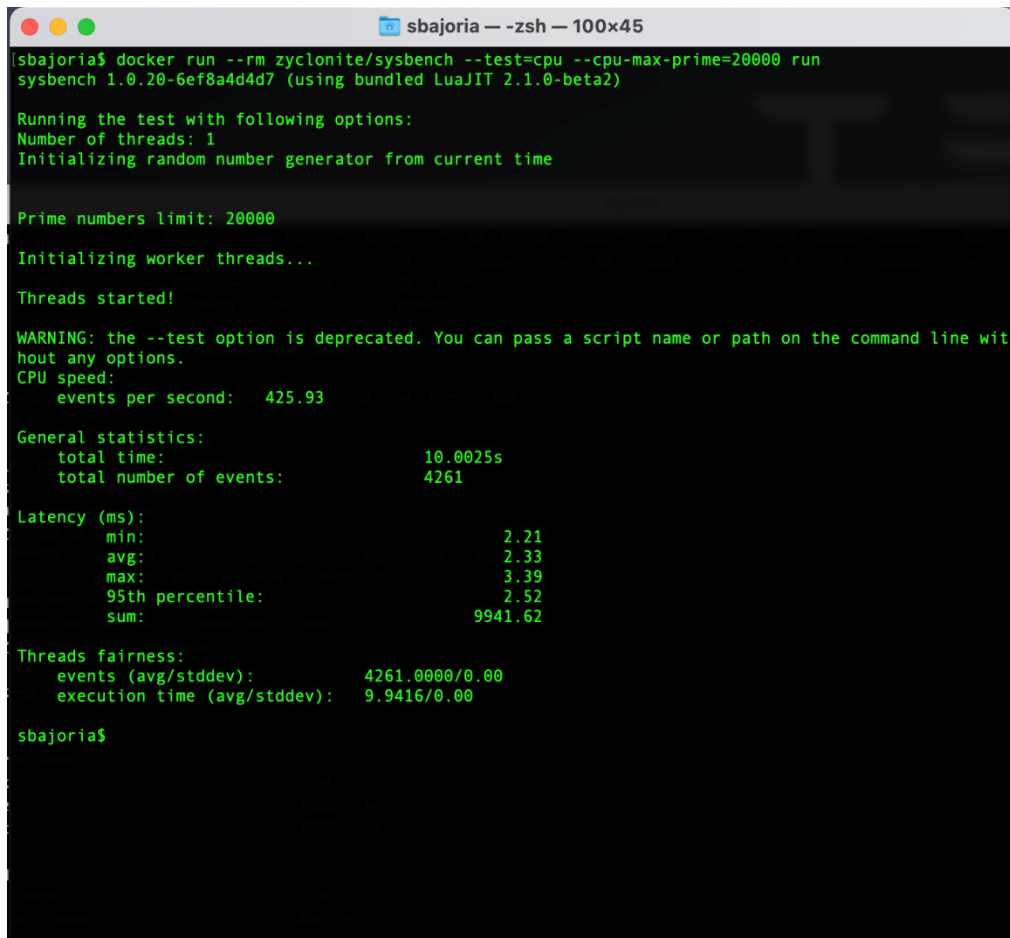
General statistics:
  total time:          10.0010s
  total number of events: 8839

Latency (ms):
  min:                 0.85
  avg:                 1.11
  max:                 32.43
  95th percentile:    1.50
  sum:                 9846.45

Threads fairness:
  events (avg/stddev): 8839.0000/0.00
  execution time (avg/stddev): 9.8464/0.00

sbajoria$
```

Test 2: `docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=20000 run`



```
sbajoria$ docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=20000 run
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 20000
Initializing worker threads...

Threads started!

WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
CPU speed:
  events per second: 425.93

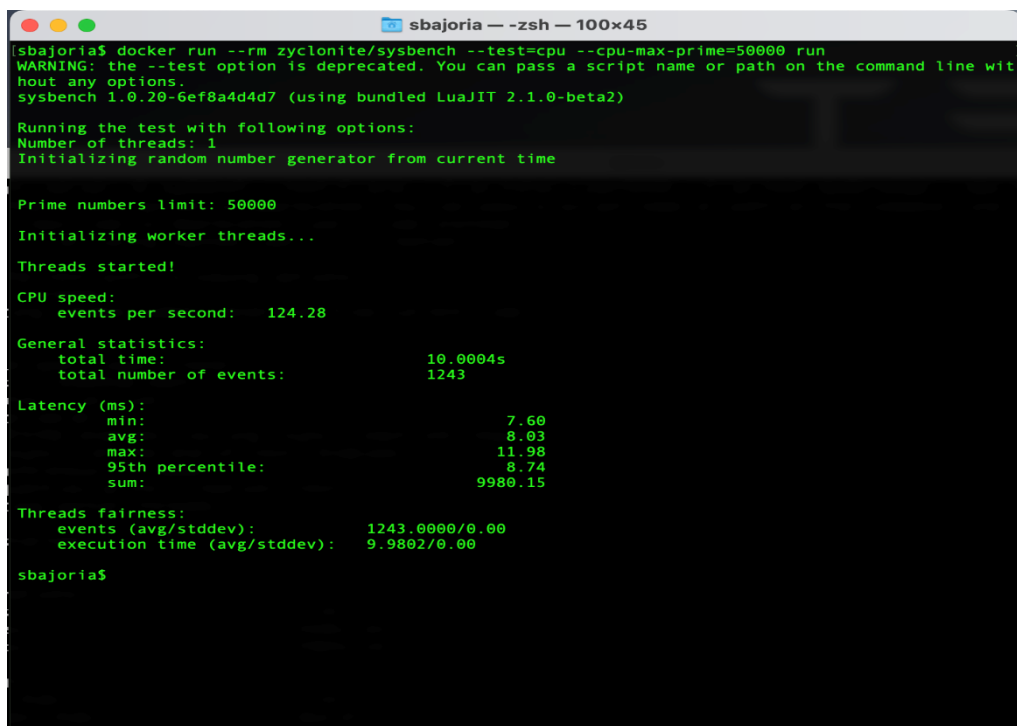
General statistics:
  total time:          10.0025s
  total number of events: 4261

Latency (ms):
  min:                 2.21
  avg:                 2.33
  max:                 3.39
  95th percentile:    2.52
  sum:                9941.62

Threads fairness:
  events (avg/stddev): 4261.0000/0.00
  execution time (avg/stddev): 9.9416/0.00

sbajoria$
```

Test 3: `docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=50000 run`



```
sbajoria$ docker run --rm zyclonite/sysbench --test=cpu --cpu-max-prime=50000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Prime numbers limit: 50000
Initializing worker threads...

Threads started!

CPU speed:
  events per second: 124.28

General statistics:
  total time:          10.0004s
  total number of events: 1243

Latency (ms):
  min:                 7.60
  avg:                 8.03
  max:                 11.98
  95th percentile:    8.74
  sum:                9980.15

Threads fairness:
  events (avg/stddev): 1243.0000/0.00
  execution time (avg/stddev): 9.9802/0.00

sbajoria$
```

CPU utilization for user-level and kernel -level while running the CPU test:

Screenshot of activity monitor when QEMU is idle:

```
Processes: 744 total, 3 running, 741 sleeping, 3784 threads      12:40:19
Load Avg: 2.54, 2.37, 2.51  CPU usage: 4.38% user, 3.2% sys, 92.59% idle
SharedLibs: 574M resident, 89M data, 35M linkedit.
MemRegions: 367296 total, 9327M resident, 230M private, 6100M shared.
PhysMem: 32G used (4092M wired), 310M unused.
VM: 27T vsize, 3136M framework vsize, 13429134(0) swapins, 14601490(0) swapouts.
Networks: packets: 20060869/15G in, 10193660/3410M out.
Disks: 11629692/161G read, 6812768/171G written.

PID    COMMAND      %CPU  TIME    #TH    #WQ    #PORT  MEM    PURG    CMPRS  PGRP
2958   Mattermost H 13.7   03:39:18 16     1      165    622M   0B      65M   2753
889    WindowServer 12.0   04:10:18 14/1    5      3975-  2413M  181M   119M   889
76     wapptunneId 11.7   67:31.64 26     1      96     78M    0B      18M    76
14012  top          9.0    00:07.02 1/1     0      29     8232K  0B      0B     14012
2854   Mattermost H 8.4    01:54:17 8       1      181    217M   0B      44M   2753
3663   com.docker.h 7.7    97:22.08 15      0      40     11G+   0B     6058M  3170
50081  qemu-system- 4.7    02:12:27 8       2      222    3328M  1216K  1142M  50067
0      kernel_task  3.5    71:10.78 294/12  0      0      507M   0B      0B     0
95640  Microsoft Te 3.2    09:11.33 18      1      277    355M   0B     142M  95573
2790   Finder       2.9    05:54.76 7       2      1529   350M-  1044K+ 90M    2790
394    com.crowdstr 1.3    85:58.93 27      6      338    321M   0B     246M  394
95635  Microsoft Te 1.0    02:32.81 9       1      228    188M   4096K  45M    95573
2753   Mattermost   0.9    19:35.88 29      1      454    102M   36K     43M   2753
10811  Terminal     0.5    00:39.96 6       1      310-   57M    388K-  0B     10811
```

Using activity monitor when running cpu test on sysbench on QEMU:

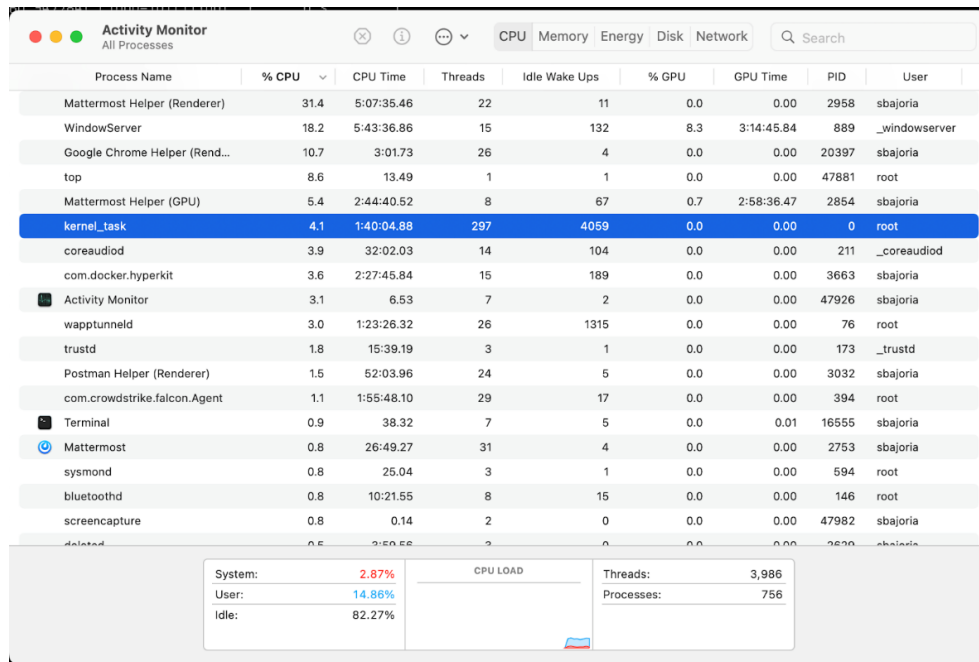
```
Processes: 746 total, 4 running, 742 sleeping, 3898 threads      12:40:46
Load Avg: 2.62, 2.39, 2.51  CPU usage: 10.97% user, 2.5% sys, 86.96% idle
SharedLibs: 574M resident, 89M data, 35M linkedit.
MemRegions: 367504 total, 9335M resident, 228M private, 6100M shared.
PhysMem: 32G used (4296M wired), 291M unused.
VM: 27T vsize, 3136M framework vsize, 13429198(0) swapins, 14601490(0) swapouts.
Networks: packets: 20062387/15G in, 10194335/3410M out.
Disks: 11629877/161G read, 6813734/171G written.

PID    COMMAND      %CPU  TIME    #TH    #WQ    #PORT  MEM    PURG    CMPRS  PGRP
50081  qemu-system- 101.2  02:12:43 8/1     2      223    3328M  1236K  1142M  50067
889    WindowServer 9.4    04:10:23 16/1    6      3979-  2454M- 210M   119M   889
2958   Mattermost H 8.5    03:39:21 15      1      162    622M   0B      65M   2753
14012  top          8.3    00:09.89 1/1     0      32     8236K  0B      0B     14012
2854   Mattermost H 5.6    01:54:19 8       1      181    217M   0B      44M   2753
3663   com.docker.h 4.9    97:23.61 15      0      40     11G    0B     6058M  3170
2790   Finder       3.5    05:56.16 8       3      1532   351M-  1380K+ 90M    2790
0      kernel_task  2.7    71:11.90 294/12  0      0      547M-  0B      0B     0
95640  Microsoft Te 1.9    09:12.09 19      1      282    355M   0B     142M  95573
487    com.apple.Ap 1.1    03:20.56 3       2      254    3536K  0B     1392K  487
2753   Mattermost   0.6    19:36.16 29      1      454    102M   36K     43M   2753
95635  Microsoft Te 0.6    02:33.05 9       1      228    188M   4096K  45M    95573
394    com.crowdstr 0.5    85:59.34 29      8      342    321M   0B     246M  394
86     JamfDaemon   0.4    05:21.48 3       2      80     59M    0B      43M   86
```

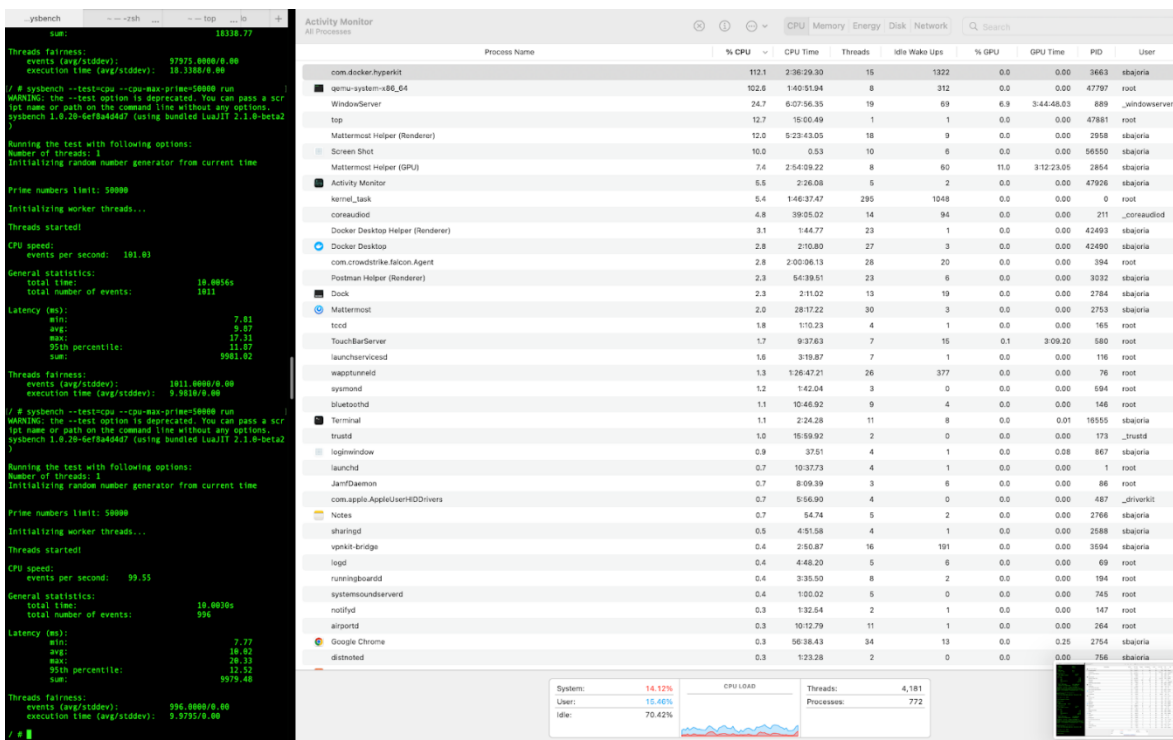
From the above screenshots we can see that the CPU usage has increased from 13.7 to 101.2 but kernel_task remains more or less the same.

Now, Running the same test for docker.

Screenshot of the activity monitor when docker is idle:



Screenshot while running the test:



	Sysbench running	Sysbench not running
User level CPU	15.46%	14.86%
Kernel level CPU	14.12%	2.87%

As we can see OS virtualization uses kernel level cpu.

FileIO test:

Since my laptop was running out of memory, tested the fileIO for a small memory, both on QEMU and Docker. Setting the max-requests to 0, meaning unlimited number of requests, and max-time to 20.

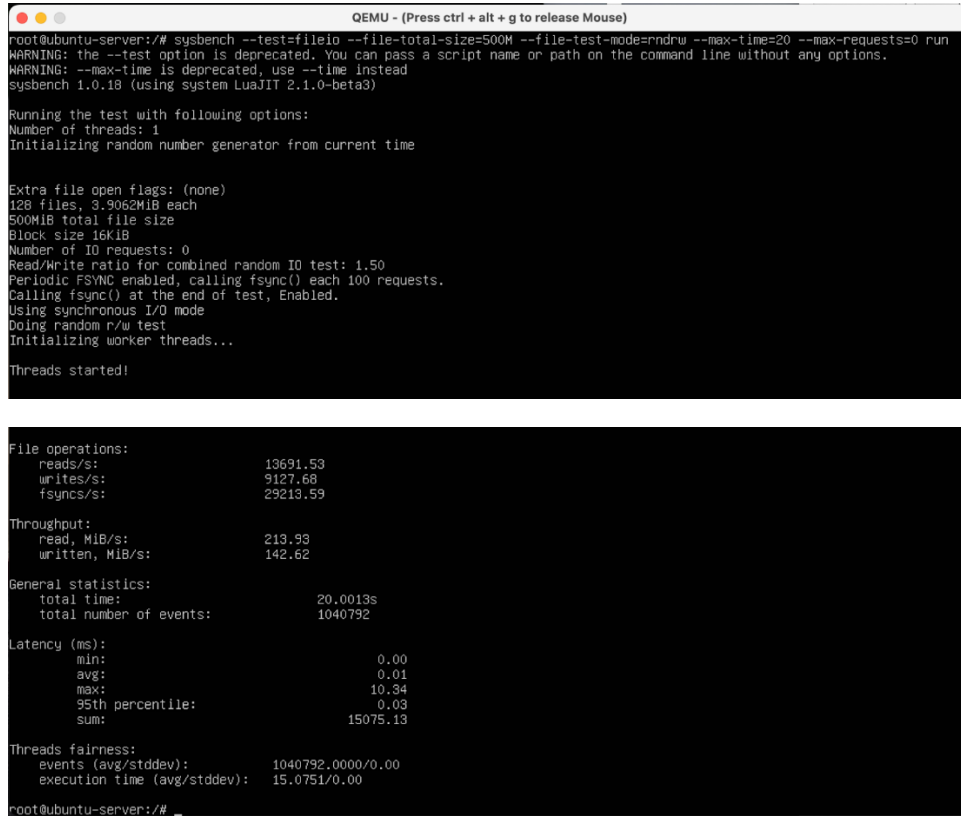
QEMU:

Test 1: Random read/write

```
sysbench --test=fileio --file-total-size=500MB prepare
```

```
sysbench --test=fileio --file-total-size=500MB --file-test-mode=rndrw --max-time=20 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500MB cleanup
```



```
QEMU - (Press ctrl + alt + g to release Mouse)
root@ubuntu-server:/# sysbench --test=fileio --file-total-size=500M --file-test-mode=rndrw --max-time=20 --max-requests=0 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --max-time is deprecated, use --time instead
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Number of IO requests: 0
Read/Write ratio for combined random IO test: 1.50
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random r/w test
Initializing worker threads...

Threads started!

File operations:
  reads/s:          13691.53
  writes/s:         9127.68
  fsyncs/s:         29213.59

Throughput:
  read, MiB/s:      213.93
  written, MiB/s:   142.62

General statistics:
  total time:       20.0013s
  total number of events: 1040792

Latency (ms):
  min:              0.00
  avg:              0.01
  max:              10.34
  95th percentile: 0.03
  sum:              15075.13

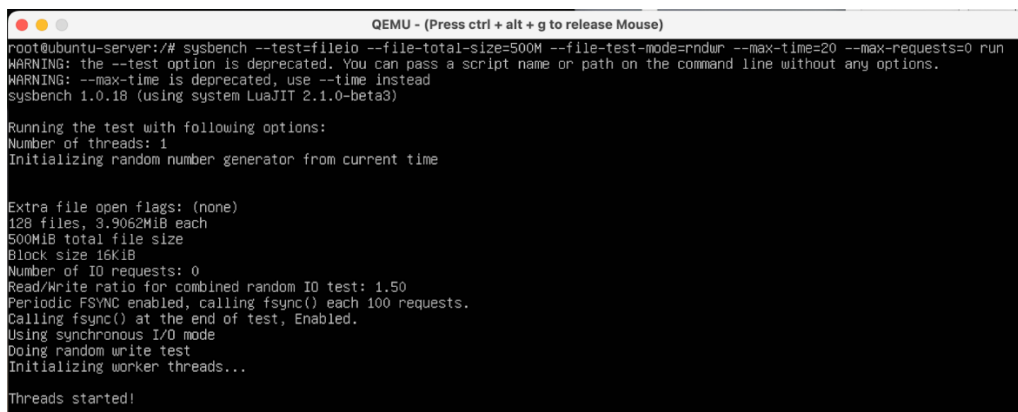
Threads fairness:
  events (avg/stddev): 1040792.0000/0.00
  execution time (avg/stddev): 15.0751/0.00
root@ubuntu-server:/#
```

Test 2: Random write

```
sysbench --test=fileio --file-total-size=500MB prepare
```

```
sysbench --test=fileio --file-total-size=500MB --file-test-mode=rndwr --max-time=20 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500MB cleanup
```



```
QEMU - (Press ctrl + alt + g to release Mouse)
root@ubuntu-server:/# sysbench --test=fileio --file-total-size=500M --file-test-mode=rndwr --max-time=20 --max-requests=0 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --max-time is deprecated, use --time instead
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Number of IO requests: 0
Read/Write ratio for combined random IO test: 1.50
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random write test
Initializing worker threads...

Threads started!
```

```
File operations:
  reads/s:          0.00
  writes/s:         24202.65
  fsyncs/s:         30981.05

Throughput:
  read, MiB/s:      0.00
  written, MiB/s:    378.17

General statistics:
  total time:        20.0013s
  total number of events: 1103822

Latency (ms):
  min:               0.00
  avg:               0.01
  max:               4.03
  95th percentile:  0.03
  sum:               15291.60

Threads fairness:
  events (avg/stddev): 1103822.0000/0.00
  execution time (avg/stddev): 15.2916/0.00

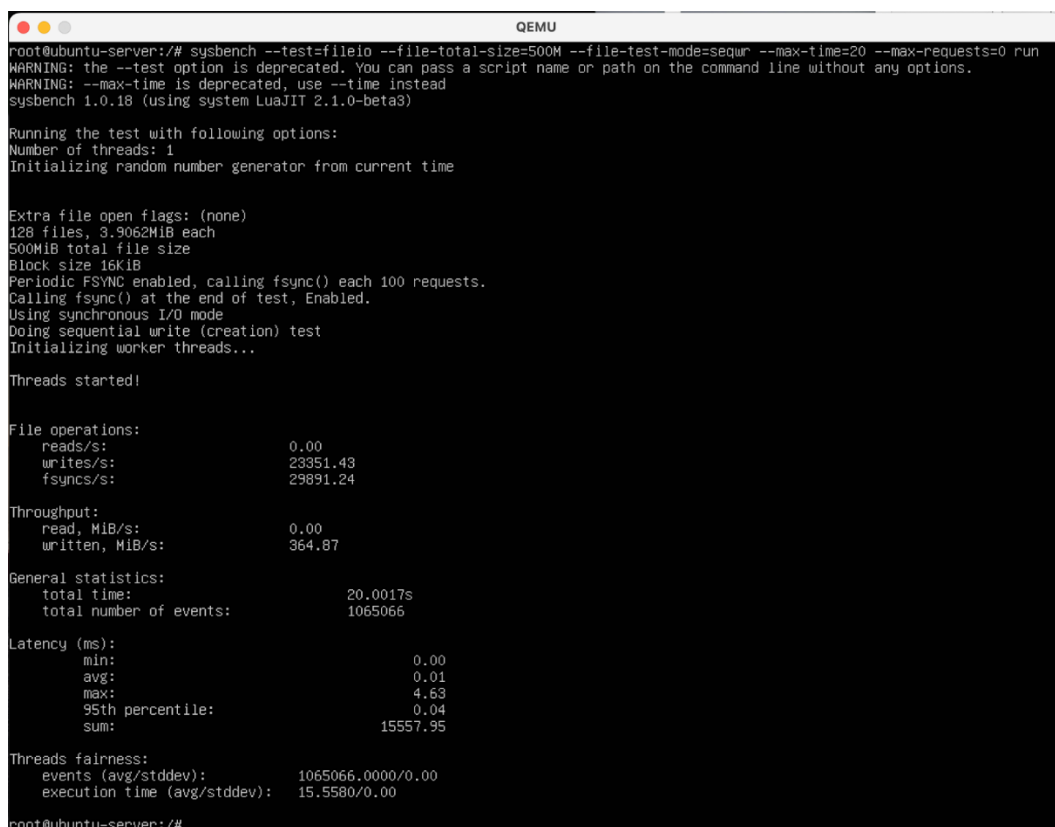
root@ubuntu-server:/#
```

Test 3: Sequential write

```
sysbench --test=fileio --file-total-size=500MB prepare
```

```
sysbench --test=fileio --file-total-size=500MB --file-test-mode=seqwr --max-time=20 -num-threads=4 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500MB cleanup
```



The screenshot shows a terminal window titled 'QEMU' with the following output:

```
root@ubuntu-server:/# sysbench --test=fileio --file-total-size=500M --file-test-mode=seqwr --max-time=20 --max-requests=0 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --max-time is deprecated, use --time instead
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing sequential write (creation) test
Initializing worker threads...

Threads started!

File operations:
  reads/s:          0.00
  writes/s:         23351.43
  fsyncs/s:         29891.24

Throughput:
  read, MiB/s:      0.00
  written, MiB/s:    364.87

General statistics:
  total time:        20.0017s
  total number of events: 1065066

Latency (ms):
  min:               0.00
  avg:               0.01
  max:               4.63
  95th percentile:  0.04
  sum:               15557.95

Threads fairness:
  events (avg/stddev): 1065066.0000/0.00
  execution time (avg/stddev): 15.5580/0.00

root@ubuntu-server:/#
```

Docker:

Run the command: `docker run --rm -it --entrypoint /bin/sh zyclonite/sysbench` before running the tests.

Test 1: Random read/write

```
sysbench --test=fileio --file-total-size=500M prepare
```

```
sysbench --test=fileio --file-total-size=500M --file-test-mode=rndrw --max-time=20 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500M cleanup
```



```
sbajoria — com.docker.cli • docker run --rm -it --entrypoint /bin/sh zyclonite/sysbench — 100x45
/ # sysbench --test=fileio --file-total-size=500M prepare
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

128 files, 4000Kb each, 500Mb total
Creating files for the test...
Extra file open flags: (none)
Creating file test_file.0
Creating file test_file.1
Creating file test_file.2
Creating file test_file.3
Creating file test_file.4
Creating file test_file.5
Creating file test_file.6
Creating file test_file.7
Creating file test_file.8
Creating file test_file.9
Creating file test_file.10
Creating file test_file.11
Creating file test_file.12
Creating file test_file.13
Creating file test_file.14
Creating file test_file.15
Creating file test_file.16
Creating file test_file.17
Creating file test_file.18
Creating file test_file.19
Creating file test_file.20
Creating file test_file.21
Creating file test_file.22
Creating file test_file.23
Creating file test_file.24
Creating file test_file.25
Creating file test_file.26
Creating file test_file.27
Creating file test_file.28
Creating file test_file.29
Creating file test_file.30
Creating file test_file.31
Creating file test_file.32
Creating file test_file.33
Creating file test_file.34
Creating file test_file.35
Creating file test_file.36

sbajoria — com.docker.cli • docker run --rm -it --entrypoint /bin/sh zyclonite/sysbench — 104x50
/ # sysbench --test=fileio --file-total-size=500M --file-test-mode=rndrw --max-requests=0 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Number of IO requests: 0
Read/Write ratio for combined random IO test: 1.50
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random r/w test
Initializing worker threads...

Threads started!

File operations:
  reads/s:          1668.12
  writes/s:         1112.08
  fsyncs/s:         3559.24

Throughput:
  read, MiB/s:      26.06
  written, MiB/s:   17.38

General statistics:
  total time:       10.0339s
  total number of events: 63490

Latency (ms):
  min:              0.01
  avg:              0.14
  max:              24.86
  95th percentile: 0.47
  sum:              9140.42

Threads fairness:
  events (avg/stddev): 63490.0000/0.00
  execution time (avg/stddev): 9.1404/0.00

/ #
/ # sysbench --test=fileio --file-total-size=500M cleanup
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Removing test files...
/ #
```

Test 2: Random write

```
sysbench --test=fileio --file-total-size=500MB prepare
```

```
sysbench --test=fileio --file-total-size=500MB --file-test-mode=rndwr --max-time=20 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500MB cleanup
```

```
sbajoria — com.docker.cli • docker run --rm -it --entrypoint /bin/sh zyclonite/sysbench — 104x50
/ # sysbench --test=fileio --file-total-size=500M --file-test-mode=rndwr --max-requests=0 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without
any options.
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Number of IO requests: 0
Read/Write ratio for combined random IO test: 1.50
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random write test
Initializing worker threads...

Threads started!

File operations:
  reads/s:           0.00
  writes/s:          2153.15
  fsyncs/s:          2762.12

Throughput:
  read, MiB/s:        0.00
  written, MiB/s:     33.64

General statistics:
  total time:         10.0303s
  total number of events: 49181

Latency (ms):
  min:                0.01
  avg:                0.19
  max:                25.70
  95th percentile:   0.51
  sum:                9320.30

Threads fairness:
  events (avg/stddev): 49181.0000/0.00
  execution time (avg/stddev): 9.3203/0.00

/ #
```

Test 3: Sequential write

```
sysbench --test=fileio --file-total-size=500MB prepare
```

```
sysbench --test=fileio --file-total-size=500MB --file-test-mode=seqwr --num-threads=4 --max-time=20 --max-requests=0 run
```

```
sysbench --test=fileio --file-total-size=500MB cleanup
```

```
/ # sysbench fileio --file-total-size=500M --file-test-mode=seqwr --num-threads=4 --max-time=20 --max-requests=0 run
WARNING: --num-threads is deprecated, use --threads instead
WARNING: --max-time is deprecated, use --time instead
sysbench 1.0.20-6ef8a4d4d7 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 4
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 3.9062MiB each
500MiB total file size
Block size 16KiB
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing sequential write (creation) test
Initializing worker threads...

Threads started!

File operations:
  reads/s:           0.00
  writes/s:          8078.05
  fsyncs/s:          10359.56

Throughput:
  read, MiB/s:        0.00
  written, MiB/s:     126.22

General statistics:
  total time:         20.0406s
  total number of events: 369014

Latency (ms):
  min:                0.01
  avg:                0.20
  max:                20.11
  95th percentile:   0.41
  sum:                74888.60

Threads fairness:
  events (avg/stddev): 92253.5000/239.79
  execution time (avg/stddev): 18.7222/0.01
```

I/O Utilization while running File I/O test:

Used `iostat` command to find the disk utilization while running a file i/o test on docker:

```
$ iostat -x
```

Screenshot of the same:

```
[sbajoria$ iostat -C
              disk0      cpu      load average
      KB/t  tps  MB/s  us sy id   1m   5m   15m
      22.13   67   1.46   10  4 86   3.67 3.99 3.73
[sbajoria$ iostat -d
              disk0
      KB/t  tps  MB/s
      22.13   67   1.46
sbajoria$ sc
```

Shell scripts:

1. FileIOQEMU.sh
2. FileIODocker.sh
3. cpuQEMU.sh
4. cpuDocker.sh

Analysis of the performance data:

Each test has been repeated 5 times, by clearing the cache after running each test using the command:

```
sync && sudo purge
```

CPU test for QEMU

Test 1:

	Total Time	Total number of events	Total Number of events per second
1	10.002	4005	400.4199
2	10.0042	3995	399.3323
3	10.0027	4018	401.6915
4	10.0038	3964	396.2494
5	10.0034	4022	402.0633

Test 2:

	Total Time	Total number of events	Total Number of events per second
1	10.0044	1551	155.0318
2	10.0021	1578	157.7669
3	10.0037	1579	157.8416
4	10.0055	1585	158.4129
5	10.0048	1581	158.0241

Test 3:

	Total Time	Total number of events	Total Number of events per second
1	10.0196	444	44.31315
2	10.0119	453	45.24616

3	10.0164	450	44.92632	10.0164
4	10.01	454	45.35465	10.01
5	10.0049	455	45.47772	10.0049

CPU test for Docker

Test 1:

	Total Time	Total number of events	Execution Time	Total Number of events per second
1	10.0007	10237	9.856	1023.48
2	10.0007	10592	9.8586	1058.96
3	10.0005	10515	9.8577	1051.29
4	10.0003	10572	9.8573	1057.01
5	10.0002	10622	9.8589	1062.03

Test 2:

	Total Time	Total number of events	Execution Time	Total Number of events per second
1	10.0011	4296	9.9428	429.5
2	10.0011	4305	9.9423	430.39
3	10.0024	4257	9.9442	425.54
4	10.0005	4272	9.941	427.1
5	10.0005	4288	9.9414	428.72

Test 3:

	Total Time	Total number of events	Execution time	Total Number of events per second
1	10.0074	1274	9.9887	127.29
2	10.007	1272	9.982	127.17
3	10.007	1273	9.9879	127.19
4	10.0011	1270	9.9824	126.97
5	10.0051	1268	9.9866	126.72

Performance analysis:

Used events per second to calculate the average, min and max values.

CPU max time = 10000

	Average	Min	Max	Standard Deviation
QEMU	399.9513	396.2494	402.0633	2.087
Docker	1050.554	1023.48	1062.03	13.98

CPU max time = 20000

	Average	Min	Max	Standard Deviation
QEMU	157.4155	155.0318	158.4129	1.212
Docker	428.25	425.54	430.39	1.733

CPU max time = 50000

	Average	Min	Max	Standard Deviation
QEMU	45.0636	44.31315	45.47772	0.41
Docker	127.068	126.72	127.29	0.2026

FileIO test for QEMU

Test 1:

	Total Time	Total number of events	Total Number of events per second
1	20.0014	129391	6469.02
2	20.0015	129550	6477.29
3	20.0015	129020	6450.31
4	20.0022	119250	5961.89
5	20.0012	129575	6478.96

Test 2:

	Total Time	Total number of events	Total Number of events per second
1	20.0014	122712	6135.86
2	20.0014	122299	6114.42
3	20.0016	122341	6116.16
4	20.002	123114	6155.94
5	20.0012	122122	6105.44

Test 3:

	Total Time	Total number of events	Total Number of events per second
1	20.0015	123815	6190.06
2	20.0014	122317	6115.67
3	20.0014	123721	6185.22
4	20.0008	124782	6238.55
5	20.0012	123841	6191.83

FileIO test for Docker

Test 1:

	Total Time	Total number of events	Total time taken by event execution	Total Number of events per second
1	20.0192	125165	18.2931	6842.197
2	20.026	123941	18.2955	6774.398
3	20.012	125671	18.2928	6869.971
4	20.023	124212	18.2966	6788.802
5	20.0188	125887	18.2921	6882.042

Test 2:

	Total Time	Total number of events	Total time taken by event execution	Total Number of events per second
1	20.0308	95934	18.6764	5136.643
2	20.035	91561	18.6714	4903.81
3	20.0295	94561	18.6591	5067.822
4	20.031	94981	18.6722	5086.76
5	20.0288	94812	18.6531	5082.908

Test 3:

	Total Time	Total number of events	Total time taken by event execution	Total Number of events per second
1	20.0437	364598	18.7355	19460.28
2	20.041	361766	18.7282	19316.65
3	20.0399	366732	18.7082	19602.74
4	20.0425	374374	18.6755	20046.26
5	20.0406	369014	18.7222	19709.97

Performance analysis:

Used events per second to calculate the average, min and max values.

File test mode = rndrw

	Average	Min	Max	Standard Deviation
QEMU	6367.367	5961.844	6478.78	203.0043
Docker	6831.482	6774.398	6882.042	42.97

File test mode = rndwr

	Average	Min	Max	Standard Deviation
QEMU	6125.444	6105.734	6155.084	17.6478
Docker	5055.589	4903.81	5136.643	79.34

File test mode = seqwr with 4 threads

	Average	Min	Max	Standard Deviation
QEMU	6184.371	6115.422	6191.678	39.52
Docker	19627.18	20046.26	19316.65	247.94

In some cases, my QEMU events are greater than docker, which I think is because I could not install native docker on my laptop so had to use docker desktop instead. Docker desktop is slower than native docker.

Observation and Analysis:

I ran all the tests for both QEMU and Docker on a single thread. Both in CPU tests and for File I/O tests, Docker has a better performance. Number of events per second for Docker (OS Virtualization) is way higher than QEMU (System Virtualization). This remains true to all the test-modes in File I/O tests. By looking at the results for a single thread, OS virtualization performs better than System Virtualization.

Also ran a test for 8 threads, for file i/o and the results are more or less the same, OS virtualization provides better performance than system virtualization.

Git Repository:

Url : <https://github.com/AshitaShetty/CloudComputing.git>

Shell scripts: <https://github.com/AshitaShetty/CloudComputing/tree/main/Homework1/shell%20scripts>

Pdf: <https://github.com/AshitaShetty/CloudComputing/tree/main/Homework1>