

Assignment 1– COEN 241 (Cloud Computing)
Submitted by: Arpita Verma
W1632653

System vs. OS Virtualization
Report

Table of Contents

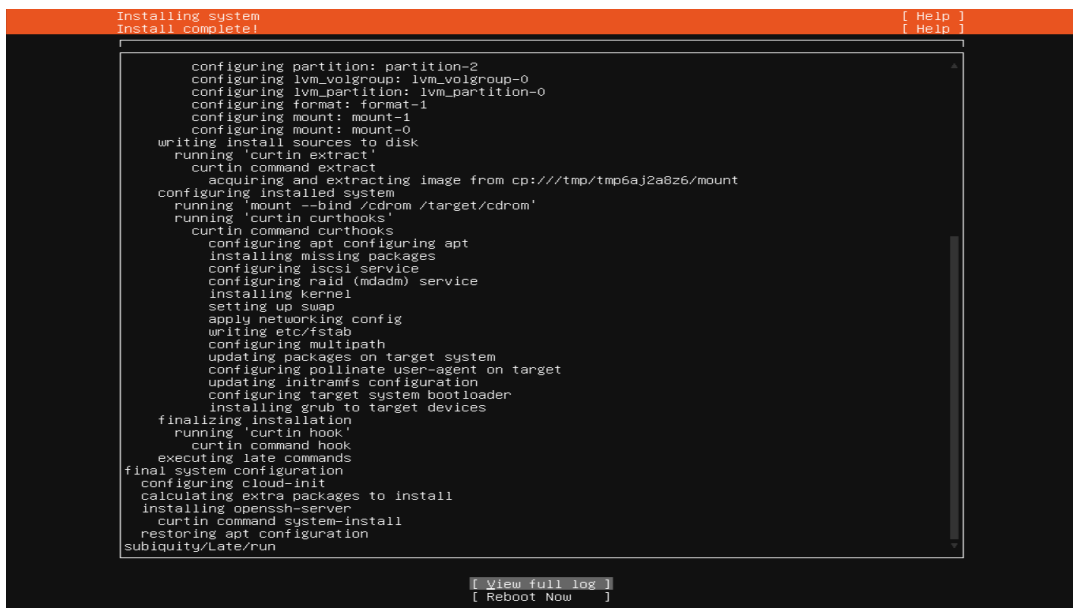
Environment Setup	3
Host System Configuration Details -	3
	3
QEMU Ubuntu Virtual Machine Configuration Details -	3
Docker Ubuntu Container Configuration Details -	3
System Virtualization Setup	4
QEMU Setup -	4
Sysbench Setup on QEMU VM -	6
Going In-Depth of QEMU	7
Operating System (OS) Virtualization Setup	8
Docker Setup -	8
Sysbench Setup in Docker Container -	9
Proof of Experiment	9
Sysbench Experiment	11
CPU Test -	12
FILEIO Test	17
Experiment Result Analysis	26
CPU Test - QEMU VM vs. Docker Container -	26
Test 1:	26
Test 2:	26
Test 3:	26
FILEIO Test - QEMU VM vs. Docker Container	27
Test 1:	27
Test 2:	27
Test 3:	27
Findings and Conclusion of CPU and FILEIO Tests-	27
Experiment Shell Scripts	28
Shell Scripts for CPU and FILE-IO Test -	28
Shell Script for FileIO Tests -	28
System Performance Tools and Analysis	30
CPU Utilization -	30
Table of comparison between above four cases -	32
IO Utilization -	34
Findings and Conclusion -	36
Automation	36
Vagrant File -	36
Dockerfile -	38
Findings and Conclusion -	38
Resources	39

Environment Setup

Host System Configuration Details -

The host system used to perform the execution of experiment has following configuration -

- CPU - x86_64 Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz with 4 cores
- Memory - 8 GB RAM
- Disk Space -
 - 500 MB - EFI System
 - 8 GB - Linux Swap
 - 115 GB - Linux Filesystem
- OS - Ubuntu 20.04.5 LTS x86_64



```
Installing system [ Help ]
Install complete! [ Help ]

configuring partition: partition-2
configuring lvm_volgroup: lvm_volgroup-0
configuring lvm_partition: lvm_partition-0
configuring format: format-1
configuring mount: mount-1
configuring mount: mount-0
writing install sources to disk
running 'curtin extract'
curtin command extract
acquiring and extracting image from cp:///tmp/tmp6aj2a8z6/mount
configuring installed system
running 'mount --bind /cdrom /target/cdrom'
running 'curtin curthooks'
curtin command curthooks
configuring apt configuring apt
installing missing packages
configuring iscsi service
configuring raid (mdadm) service
installing kernel
setting up swap
apply networking config
writing etc/fstab
configuring multipath
updating packages on target system
configuring pollinate user-agent on target
updating initramfs configuration
configuring target system bootloader
installing grub to target devices
finalizing installation
running 'curtin hook'
curtin command hook
executing late commands
final system configuration
configuring cloud-init
calculating extra packages to install
installing openssh-server
curtin command system-install
restoring apt configuration
subiquity/Late/run

[ View full log ]
[ Reboot Now ]
```

QEMU Ubuntu Virtual Machine Configuration Details -

The Ubuntu VM is created with QEMU by allocating 20 GB of disk space (qcow2 file format), 2 GB of RAM, and 2 CPU cores.

Docker Ubuntu Container Configuration Details -

The Ubuntu container in docker also allocated the same configuration as Ubuntu VM having 2 GB of RAM and 2 CPU cores.

System Virtualization Setup

QEMU Setup -

1. Install the QEMU on host

```
arpita@system:~$ sudo apt-get install qemu
[sudo] password for arpita:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  gconf2 libflashrom1 libftdi1-2 liblvm15
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  qemu
0 upgraded, 1 newly installed, 0 to remove and 27 not upgraded.
Need to get 15.3 kB of archives.
After this operation, 139 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 qemu amd64 1:6.2+dfsg-2ubuntu6.6 [15.3 kB]
Fetched 15.3 kB in 0s (35.5 kB/s)
Selecting previously unselected package qemu.
(Reading database ... 239820 files and directories currently installed.)
Preparing to unpack .../qemu_1%3a6.2+dfsg-2ubuntu6.6_amd64.deb ...
Unpacking qemu (1:6.2+dfsg-2ubuntu6.6) ...
Setting up qemu (1:6.2+dfsg-2ubuntu6.6) ...
arpita@system:~$
```

2. Download ubuntu server image using following command -
\$ wget <https://releases.ubuntu.com/focal/ubuntu-20.04.5-live-server-amd64.iso>

```
arpita-server login: arpita
Password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-125-generic x86_64)
```

```
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage
```

```
System information as of Wed 01 Feb 2023 12:19:56 AM UTC
```

```
System load:          0.9
Usage of /:            41.0% of 9.75GB
Memory usage:         9%
Swap usage:           0%
Processes:            110
Users logged in:      0
IPv4 address for ens3: 10.0.2.15
IPv6 address for ens3: fec0::5054:ff:fe12:3456
```

```
0 updates can be applied immediately.
```

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

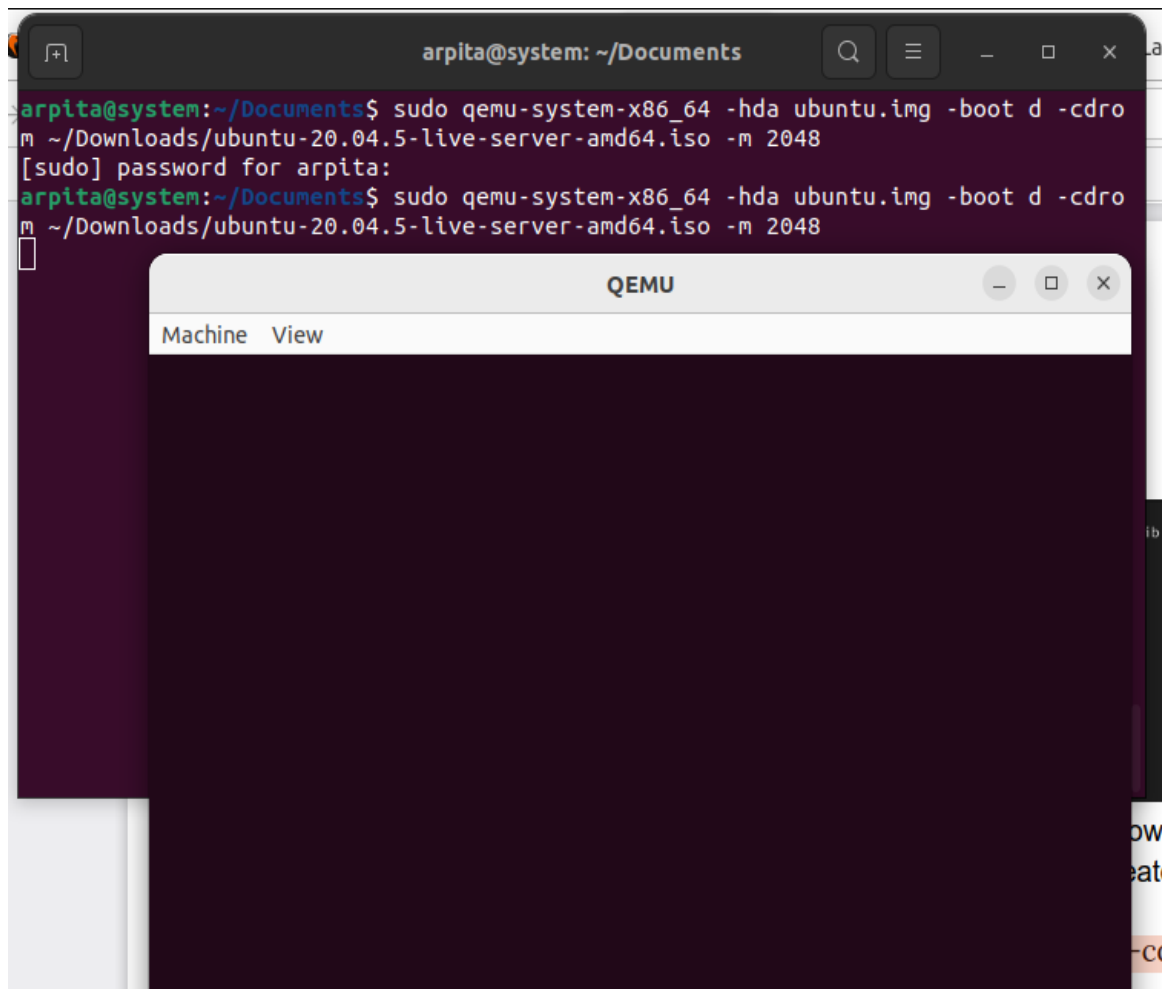
```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

3. Create QEMU image of ubuntu in qcow2 file format using following command -
\$ sudo qemu-img create ubuntu.img 20G -f qcow2

```
arpita@system:~$ sudo qemu-img create ubuntu.img 20G -f qcow2
Formatting 'ubuntu.img', fmt=qcow2 cluster_size=65536 extended_l2=off compression_type=zlib size=21474836480 lazy_refcounts=off refcount_bits=16
arpita@system:~$
```

4. After creating image, install the the Ubuntu VM using iso downloaded in 2nd step i.e. ubuntu-20.04.5-live-server-amd64.iso in the disk image created in 3rd step i.e. ubuntu.qcow2 using following command -
\$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -cdrom ubuntu-20.04.5-live-server-amd64.iso -m 2048



In the above command the flags/option used have following meanings -

- -hda : use file as a hard disk
- -boot : this option specifies the boot order, for x86 architecture these drive letters are - a, b (for floppy drives), c (first hard-disk), d (first CD-ROM), n-p (Etherboot from network adapter 1-4). Hard-disk boot is the default option.
- -cdrom : this option specifies the .iso file to be used as a base for image we are creating
- -m : this option sets guest OS's startup RAM to specified value i.e. 2048 MB

Sysbench Setup on QEMU VM -

1. Once VM is launched after finishing the above QEMU setup steps, we are now ready to install sysbench on the Ubuntu VM using following command

-

```
$ curl -s
https://packagecloud.io/install/repositories/akopytov/sysbench/script.deb.sh
| sudo bash
$ apt -y install sysbench
```

2. Check the version of sysbench installed using following command -

```
$ sysbench --
version
Command Output
```

```
-
arpita@arpita-server:~$ sysbench --version
sysbench 1.0.20
arpita@arpita-server:~$
```

Going In-Depth of QEMU

We can launch the Ubuntu OS from above created ubuntu.img by providing a lot of options. For example we can provide options to give it memory, cpu, network, block device, accelerators etc.

Following are some of the examples of launching Ubuntu VM using extra options -

1. We can start Ubuntu VM by using -accel option to accelerate the machine using paravirtualized hypervisors such as kvm, xen, tcg (default) inside them. For example below command uses 'kvm' accelerator to start -

```
$ sudo qemu-system-x86_64 -hda ubuntu1.qcow2 -accel kvm -boot c -m 2048
```

2. We can use -smp option to mention the number of cores the Guest OS can use and -cpu to provide an option for choosing from supported CPUs by QEMU. For example, below command uses 2 cores and for cpu type 'host' option which emulates host OS's cpu -

```
$ sudo qemu-system-x86_64 -hda ubuntu1.qcow2 -accel kvm -boot c -m
2048 -smp 2 -cpu host
```

3. We can use the -drive option to set the drive for guest os to use. For example, below command sets the ubuntu.img drive and uses virt-io interface -

```
$ sudo qemu-system-x86_64 -drive file=ubuntu1.img,if=virtio -accel kvm -
boot c -m 2048 -smp 2 -cpu host
```

4. We can use the '-netdev user' option to configure the user mode host network backend which requires no administrator privilege to run. Below is command uses the '-netdev user' -

```
$ sudo qemu-system-x86_64 -drive file=ubuntu1.img,if=virtio -accel kvm -boot c -m 2048 -smp 2 -cpu host -netdev user,id=net0
```

5. Combinedly, below is the more advanced command using various options and their values to launch the guest operating system -

```
$ sudo qemu-system-x86_64 -accel kvm \  
    -cpu host \  
    -m 2048 \  
    -smp 2 \  
    -hda ubuntu.img \  
    -boot c \  
    -device virtio-net,netdev=vm \  
    -netdev user,id=vm,hostfwd=tcp:127.0.0.1:9003-:22
```

Operating System (OS) Virtualization Setup

Docker Setup -

1. Install the Docker on host


```
arpita@system:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

arpita@system:~$
arpita@system:~$ docker --version
Docker version 20.10.23, build 7155243
arpita@system:~$
```

Sysbench Setup in Docker Container -

1. Now, since the ubuntu container is running, we can now install the sysbench in container using following commands –

```
root@acf04f6f0574:/# sysbench --version
sysbench 1.0.20
root@acf04f6f0574:/# █
```

We need ensure the RAM, CPU allocation and the version of sysbench is same in both System Virtualization (QEMU) and OS Virtualization (Docker) setup.

Proof of Experiment

1. QEMU Running Environment –

```
arpita@system: ~/Documents
QEMU
Machine View
Ubuntu 20.04.5 LTS arpita-server tty1
arpita-server login: arpita
Password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-137-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu 02 Feb 2023 12:33:29 AM UTC

System load:          0.45
Usage of /:            50.3% of 9.75GB
Memory usage:         9%
Swap usage:           0%
Processes:            115
Users logged in:      0
IPv4 address for ens3: 10.0.2.15
IPv6 address for ens3: fec0::5054:ff:fe12:3456

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

31 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Wed Feb  1 21:18:39 UTC 2023 on tty1
arpita@arpita-server:~$ _
```

2. Docker Running Environment -

```
arpita@system:~/Documents/linux$ docker container run -it arpitaverma03/ubuntu-sysbench:v1
root@3369ad992e8e:/# sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=seqrewr prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

128 files, 81920Kb each, 10240Mb 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
```

Sysbench Experiment

CPU Test -

1. CPU Test in QEMU VM -

a. Test 1:

\$ sysbench cpu --threads=1 --cpu-max-prime=35000 --time=10 run

```
arpita@arpita-server:~$ sysbench cpu --threads=1 --cpu-max-prime=35000 --time=10 run
sysbench 1.0.20 (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: 35000

Initializing worker threads...

Threads started!

CPU speed:
  events per second:   160.30

General statistics:
  total time:          10.0040s
  total number of events: 1604

Latency (ms):
  min:                 6.09
  avg:                 6.23
  max:                 25.46
  95th percentile:    6.43
  sum:                 10000.67

Threads fairness:
  events (avg/stddev): 1604.0000/0.00
  execution time (avg/stddev): 10.0007/0.00

arpita@arpita-server:~$ _
```

b. Test2

\$ sysbench cpu --threads=2 --cpu-max-prime=350000 --time=30 run

```
arpita@arpita-server:~$ sysbench cpu --threads=2 --cpu-max-prime=350000 --time=30 run
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

```
Running the test with following options:
Number of threads: 2
Initializing random number generator from current time
```

```
Prime numbers limit: 350000
```

```
Initializing worker threads...
```

```
Threads started!
```

```
CPU speed:
  events per second:    12.31
```

```
General statistics:
  total time:           30.0570s
  total number of events: 370
```

```
Latency (ms):
  min:                  160.23
  avg:                  162.44
  max:                  229.38
  95th percentile:     164.45
  sum:                  60101.87
```

```
Threads fairness:
  events (avg/stddev):  185.0000/0.00
  execution time (avg/stddev): 30.0509/0.00
```

```
arpita@arpita-server:~$ _
```

c. Test 3

```
$ sysbench cpu --threads=16 --cpu-max-prime=6000000 --time=50 run
```

```
arpita@arpita-server:~$ sysbench cpu --threads=16 --cpu-max-prime=6000000 --time=50 run
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

```
Running the test with following options:
Number of threads: 16
Initializing random number generator from current time
```

```
Prime numbers limit: 6000000
```

```
Initializing worker threads...
```

```
Threads started!
```

```
CPU speed:
  events per second:    0.23
```

```
General statistics:
  total time:           70.2802s
  total number of events: 16
```

```
Latency (ms):
  min:                  70142.43
  avg:                  70208.33
  max:                  70248.76
  95th percentile:     69758.52
  sum:                  1123333.33
```

```
Threads fairness:
  events (avg/stddev):  1.0000/0.00
  execution time (avg/stddev): 70.2083/0.03
```

```
arpita@arpita-server:~$
```

2. CPU Test in Docker Container -

a. Test 1:

```
$ sysbench cpu --threads=1 --cpu-max-prime=35000 --time=10 run
```

```
root@e2a69ce7da10:/# sysbench cpu --threads=1 --cpu-max-prime=35000 --time=10 run
sysbench 1.0.20 (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: 35000
```

```
Initializing worker threads...
```

```
Threads started!
```

```
CPU speed:
```

```
events per second: 161.72
```

```
General statistics:
```

```
total time: 10.0029s
```

```
total number of events: 1618
```

```
Latency (ms):
```

```
min: 6.06
```

```
avg: 6.18
```

```
max: 8.82
```

```
95th percentile: 6.43
```

```
sum: 10000.68
```

```
Threads fairness:
```

```
events (avg/stddev): 1618.0000/0.00
```

```
execution time (avg/stddev): 10.0007/0.00
```

```
root@e2a69ce7da10:/#
```

b. Test 2

```
$ sysbench cpu --threads=2 --cpu-max-prime=350000 --time=30 run
```

```
root@e2a69ce7da10:/# sysbench cpu --threads=2 --cpu-max-prime=350000 --time=30 run
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
```

```
Running the test with following options:
Number of threads: 2
Initializing random number generator from current time
```

```
Prime numbers limit: 350000
```

```
Initializing worker threads...
```

```
Threads started!
```

```
CPU speed:
  events per second:   12.42
```

```
General statistics:
  total time:           30.0985s
  total number of events: 374
```

```
Latency (ms):
  min:                  156.76
  avg:                  160.75
  max:                  171.34
  95th percentile:     161.51
  sum:                  60119.63
```

```
Threads fairness:
  events (avg/stddev):   187.0000/0.00
  execution time (avg/stddev): 30.0598/0.03
```

```
root@e2a69ce7da10:/#
```

c. Test 3

\$ sysbench cpu --threads=16 --cpu-max-prime=600000 --time=50 run

```
root@e2a69ce7da10:/# sysbench cpu --threads=16 --cpu-max-prime=6000000 --time=50 run
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 16
Initializing random number generator from current time

Prime numbers limit: 6000000

Initializing worker threads...

Threads started!

CPU speed:
  events per second:    0.35

General statistics:
  total time:           92.1729s
  total number of events: 32

Latency (ms):
  min:                  44880.58
  avg:                  45828.28
  max:                  47132.58
  95th percentile:     46941.21
  sum:                  1466505.00

Threads fairness:
  events (avg/stddev):  2.0000/0.00
  execution time (avg/stddev): 91.6566/0.39
```


FILEIO Test

1. FILE-IO Test in QEMU VM -

Test 1:

```
$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test mode=rndwr prepare
```

```
$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr run
```

```
$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr cleanup
```

```
arpita@arpita-server:~$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

```
128 files, 24576Kb each, 3072Mb 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
-
```

```
arpita@arpita-server:~$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr run
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (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, 24MiB each
3GiB 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!
-
```

```
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (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, 24MiB each

3GiB 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:	948.19
fsyncs/s:	1257.64

Throughput:

read, MiB/s:	0.00
written, MiB/s:	14.82

General statistics:

total time:	10.1222s
total number of events:	21821

Latency (ms):

min:	0.01
avg:	1.83
max:	138.91
95th percentile:	5.67
sum:	39961.67

Threads fairness:

events (avg/stddev):	5455.2500/69.32
execution time (avg/stddev):	9.9904/0.00

arpita@arpita-server:~\$

Test 2

\$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test mode=rndwr prepare

\$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=rndwr run

\$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=rndwr cleanup

```
arpita@arpita-server:~$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqwrn prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

128 files, 40960Kb each, 5120Mb 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

```

arpita@arpita-server:~$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr run
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 16
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 40MiB each
5GiB 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 rewrite test
Initializing worker threads...

Threads started!

File operations:
  reads/s:           0.00
  writes/s:          1226.81
  fsyncs/s:          1757.24

Throughput:
  read, MiB/s:        0.00
  written, MiB/s:      19.17

General statistics:
  total time:          10.3499s
  total number of events: 28843

Latency (ms):
  min:                 0.01
  avg:                 5.55
  max:                 436.23
  95th percentile:    28.16
  sum:                 160068.03

Threads fairness:
  events (avg/stddev): 1802.6875/167.09
  execution time (avg/stddev): 10.0043/0.01

```

Test 3:

```

$ sysbench --num-threads=8 fileio --file-total-size=10G --file-test mode=rndwr prepare
$ sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndwr run
$ sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndwr cleanup

```

```
arpita@arpita-server:~$ sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndrw prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

```
128 files, 81920Kb each, 10240Mb 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
```

```
_
```

```
arpita@arpita-server:~$ sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndrw run
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using bundled LuaJIT 2.1.0-beta2)
```

```
Running the test with following options:
```

```
Number of threads: 8
```

```
Initializing random number generator from current time
```

```
Extra file open flags: (none)
```

```
128 files, 80MiB each
```

```
10GiB 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:          507.06
  writes/s:         337.85
  fsyncs/s:         1180.30

Throughput:
  read, MiB/s:      7.92
  written, MiB/s:   5.28

General statistics:
  total time:        10.1918s
  total number of events: 19621

Latency (ms):
  min:               0.00
  avg:               4.08
  max:               508.44
  95th percentile:  10.27
  sum:               79966.28

Threads fairness:
  events (avg/stddev): 2452.6250/129.66
  execution time (avg/stddev): 9.9958/0.00

```

2. FILE-IO Test in Docker Container -

Test 1:

```

$ sysbench --threads=4 fileio --file-total-size=3G --file-test-mode=rndwr prepare
$ sysbench --threads=4 fileio --file-total-size=3G --file-test-mode=rndwr run
$ sysbench --threads=4 fileio --file-total-size=3G --file-test-mode=rndwr cleanup Run

```

```

root@e2a69ce7da10:/# sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

128 files, 24576Kb each, 3072Mb 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

```

```
root@e2a69ce7da10:/# sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr run
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
```

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

```
Extra file open flags: (none)
128 files, 24MiB each
3GiB 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:          3350.70
  fsyncs/s:          4332.38
```

```
Throughput:
  read, MiB/s:       0.00
  written, MiB/s:    52.35
```

```
General statistics:
  total time:         10.0253s
  total number of events: 76532
```

```
Latency (ms):
  min:                0.01
  avg:                0.52
  max:                136.83
  95th percentile:   1.55
  sum:                39944.22
```

```
Threads fairness:
  events (avg/stddev): 19133.0000/72.24
  execution time (avg/stddev): 9.9861/0.00
```

Test 2:

```
$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr prepare
$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr run
$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr cleanup Run
```

```
root@e2a69ce7da10:/# sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
```

```
128 files, 40960Kb each, 5120Mb 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
```

```
root@e2a69ce7da10:/# sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr run
```

```
WARNING: --num-threads is deprecated, use --threads instead
```

```
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
```

```
Running the test with following options:
```

```
Number of threads: 16
```

```
Initializing random number generator from current time
```

```
Extra file open flags: (none)
```

```
128 files, 40MiB each
```

```
5GiB 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 rewrite test
```

```
Initializing worker threads...
```

```
Threads started!
```

```
File operations:
```

reads/s:	0.00
writes/s:	5798.03
fsyncs/s:	7613.59

```
Throughput:
```

read, MiB/s:	0.00
written, MiB/s:	90.59

```
General statistics:
```

total time:	10.0702s
total number of events:	133039

```
Latency (ms):
  min:                0.01
  avg:                1.20
  max:                256.44
  95th percentile:   5.47
  sum:               160124.36

Threads fairness:
  events (avg/stddev):    8314.9375/275.23
  execution time (avg/stddev): 10.0078/0.00

root@e2a69ce7da10:/#
```

Test 3:

```
$ sysbench --threads=8 fileio --file-total-size=10G --file-test-mode=rndrw prepare
$ sysbench --threads=8 fileio --file-total-size=10G --file-test-mode=rndrw run
$ sysbench --threads=8 fileio --file-total-size=10G --file-test-mode=rndrw cleanup Run
```

```
root@e2a69ce7da10:/# sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndrw prepare
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

128 files, 81920Kb each, 10240Mb 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
```



```
root@e2a69ce7da10:/# sysbench --num-threads=8 fileio --file-total-size=10G --file-test-mode=rndrw run
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
```

Running the test with following options:
Number of threads: 8
Initializing random number generator from current time

Extra file open flags: (none)
128 files, 80MiB each
10GiB 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:	2844.66
writes/s:	1895.94
fsyncs/s:	6167.37

Throughput:

read, MiB/s:	44.45
written, MiB/s:	29.62

General statistics:

total time:	10.0386s
total number of events:	108502

Latency (ms):

min:	0.00
avg:	0.74
max:	190.27
95th percentile:	2.14
sum:	79924.06

Threads fairness:

events (avg/stddev):	13562.7500/136.61
execution time (avg/stddev):	9.9905/0.00

Experiment Result Analysis

CPU Test - QEMU VM vs. Docker Container -

To perform CPU Test, I have taken following three main parameters into consideration -

1. --threads: number of threads to be used to perform test
2. --cpu-max-prime: the maximum number up to which numbers to be tested if they are prime
3. --time: maximum time process can take to finish

By altering above parameters, following three CPU Tests are performed and the result of them in various measures are shown below in respective test tables -

Test 1:

\$ sysbench cpu --threads=1 --cpu-max-prime=35000 --time=10 run

	min	average	max	no. of events
QEMU VM	6.09	6.23	25.46	1604
Docker	6.06	6.18	8.82	1618

Test 2:

\$ sysbench cpu --threads=2 --cpu-max-prime=350000 --time=30 run

	min	average	max	no. of events
QEMU VM	160.23	162.44	229.38	370
Docker	156.76	160.75	171.34	374

Test 3:

\$ sysbench cpu --threads=16 --cpu-max-prime=6000000 --time=50 run

	min	average	max	no. of events
QEMU VM	70142.43	70208.33	70248.76	16
Docker	44880.58	45828.28	47132.58	32

FILEIO Test - QEMU VM vs. Docker Container

To perform FILE-IO Test, I have taken following three main parameters into consideration-

1. --threads: number of threads to be used to perform test
2. --file-total-size: total size of file/files to be created
3. --file-test-mode: mode of file test, there are five modes of file-io -
 - a. rndrd
 - b. rndrw
 - c. rndwr
 - d. seqrd
 - e. seqrewr
 - f. seqwr

By altering above parameters, following three File-IO Tests are performed and the result of them in various measures are shown below in respective test tables -

Test 1:

```
$ sysbench --num-threads=4 fileio --file-total-size=3G --file-test-mode=rndwr run
```

	min	average	max	no. of events
QEMU VM	0.01	1.83	138.91	21821
Docker	0.01	0.52	136.83	76532

Test 2:

```
$ sysbench --num-threads=16 fileio --file-total-size=5G --file-test-mode=seqrewr run
```

	min	average	max	no. of events
QEMU VM	0.01	5.55	436.23	28843
Docker	0.01	1.20	256.44	133039

Test 3:

```
$ sysbench --num-threads=8 fileio --file-total-size=4G --file-test-mode=rndrw run
```

	min	average	max	no. of events
QEMU VM	0	4.08	508.44	19621
Docker	0	0.74	190.27	108502

Findings and Conclusion of CPU and FILEIO Tests-

The above analysis has been based on three different cases for each CPU test and

File-IO test. Moreover, each test is run five times to see if the result of each test is consistent with the other case. Following are some of the findings which can be drawn from the results mentioned -

Experiment Shell Scripts

Shell Scripts for CPU and FILE-IO Test -

1. Shell Script for CPU Tests -

\$./cpu_test.sh

```
#!/bin/bash

echo "Hello World, you are in $0, and going to test cpu!"

PRIMES_END=("35000" "350000" "6000000")
MAX_TIME=("10" "30" "50")
NUM_THREADS=("1" "2" "16")
TOTAL_RUNS=5
TOTAL_CASES=3

for ((i=0; i<$TOTAL_CASES;i++))
do
echo "Starting ${i+1}st Test Case"
for (( j=1; j <= $TOTAL_RUNS; j++ ))
do
echo "Running ${j}st run of Test Case ${i+1}"
sysbench cpu --threads=${NUM_THREADS[$i]} --cpu-max-
prime=${PRIMES_END[$i]} --time=${MAX_TIME[$i]} run
echo "Completed ${j}st run of Test Case ${i+1}"
done
echo "Completed ${i}st Test Case"
done
```

2. Shell Script for FileIO Tests -

\$./fileio_test.sh

```
#!/bin/bash

echo "Hello World, you are in $0, and going to test fileio"

NUM_THREADS=("4" "16" "8")
TOTAL_FILE_SIZES=("2G" "5G" "4G")
FILE_MODE=("rndwr" "seqrewr" "rndrw")
TOTAL_RUNS=5
```

```
TOTAL_CASES=3

for ((i=0; i<$TOTAL_CASES;i++))
do
    echo "*****Starting ${i+1}st Test
Case*****"
    for ((j=1; j <=$TOTAL_RUNS; j++ ))
    do
        echo "Running ${j}st run of Test Case ${i+1}"
        sysbench --threads=${NUM_THREADS[$i]} fileio --file-total-
size=${TOTAL_FILE_SIZES[$i]} --file-test-mode=${FILE_MODE[i]} prepare
        sysbench --threads=${NUM_THREADS[$i]} fileio --file-total-
size=${TOTAL_FILE_SIZES[$i]} --file-test-mode=${FILE_MODE[i]} run
        sysbench --threads=${NUM_THREADS[$i]} fileio --file-total-
size=${TOTAL_FILE_SIZES[$i]} --file-test-mode=${FILE_MODE[i]} cleanup
        echo "Completed ${j}st run of Test Case ${i+1}"
    done
    echo "*****Completed ${i}st Test
Case*****"
done
```

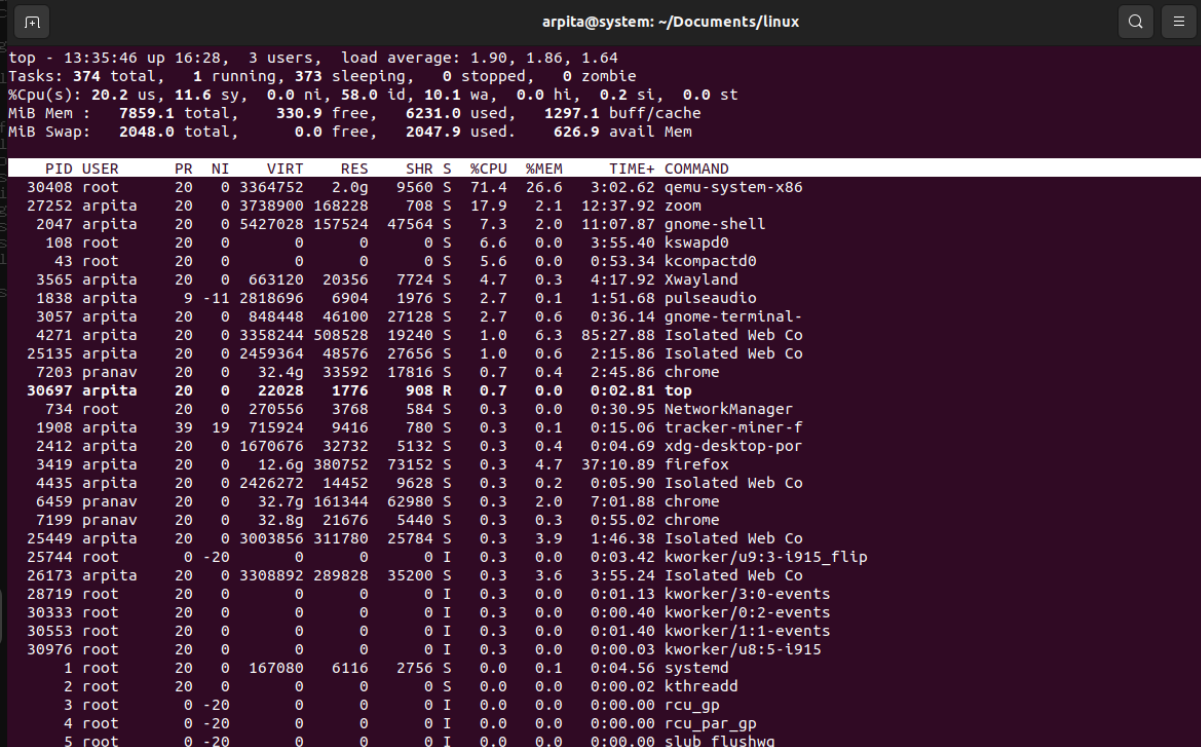
System Performance Tools and Analysis

CPU Utilization -

By making use of 'top' command in linux we can see the cpu usage in User Mode, System Mode and Idle Mode.

The CPU Utilization is observed in following cases -

1. QEMU VM Running Sysbench CPU Test Case –



```
arpita@system: ~/Documents/linux
top - 13:35:46 up 16:28, 3 users, load average: 1.90, 1.86, 1.64
Tasks: 374 total, 1 running, 373 sleeping, 0 stopped, 0 zombie
%Cpu(s): 20.2 us, 11.6 sy, 0.0 ni, 58.0 id, 10.1 wa, 0.0 hi, 0.2 si, 0.0 st
MiB Mem : 7859.1 total, 330.9 free, 6231.0 used, 1297.1 buff/cache
MiB Swap: 2048.0 total, 0.0 free, 2047.9 used, 626.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
30408	root	20	0	3364752	2.0g	9560	S	71.4	26.6	3:02.62	qemu-system-x86
27252	arpita	20	0	3738900	168228	708	S	17.9	2.1	12:37.92	zoom
2047	arpita	20	0	5427028	157524	47564	S	7.3	2.0	11:07.87	gnome-shell
108	root	20	0	0	0	0	S	6.6	0.0	3:55.40	kswapd0
43	root	20	0	0	0	0	S	5.6	0.0	0:53.34	kcompactd0
3565	arpita	20	0	663120	20356	7724	S	4.7	0.3	4:17.92	Xwayland
1838	arpita	9	-11	2818696	6904	1976	S	2.7	0.1	1:51.68	pulseaudio
3057	arpita	20	0	848448	46100	27128	S	2.7	0.6	0:36.14	gnome-terminal-
4271	arpita	20	0	3358244	508528	19240	S	1.0	6.3	85:27.88	Isolated Web Co
25135	arpita	20	0	2459364	48576	27656	S	1.0	0.6	2:15.86	Isolated Web Co
7203	pranav	20	0	32.4g	33592	17816	S	0.7	0.4	2:45.86	chrome
30697	arpita	20	0	22028	1776	908	R	0.7	0.0	0:02.81	top
734	root	20	0	270556	3768	584	S	0.3	0.0	0:30.95	NetworkManager
1908	arpita	39	19	715924	9416	780	S	0.3	0.1	0:15.06	tracker-miner-f
2412	arpita	20	0	1670676	32732	5132	S	0.3	0.4	0:04.69	xdg-desktop-por
3419	arpita	20	0	12.6g	380752	73152	S	0.3	4.7	37:10.89	firefox
4435	arpita	20	0	2426272	14452	9628	S	0.3	0.2	0:05.90	Isolated Web Co
6459	pranav	20	0	32.7g	161344	62980	S	0.3	2.0	7:01.88	chrome
7199	pranav	20	0	32.8g	21676	5440	S	0.3	0.3	0:55.02	chrome
25449	arpita	20	0	3003856	311780	25784	S	0.3	3.9	1:46.38	Isolated Web Co
25744	root	0	-20	0	0	0	I	0.3	0.0	0:03.42	kworker/u9:3-i915_flip
26173	arpita	20	0	3308892	289828	35200	S	0.3	3.6	3:55.24	Isolated Web Co
28719	root	20	0	0	0	0	I	0.3	0.0	0:01.13	kworker/3:0-events
30333	root	20	0	0	0	0	I	0.3	0.0	0:00.40	kworker/0:2-events
30553	root	20	0	0	0	0	I	0.3	0.0	0:01.40	kworker/1:1-events
30976	root	20	0	0	0	0	I	0.3	0.0	0:00.03	kworker/u8:5-i915
1	root	20	0	167080	6116	2756	S	0.0	0.1	0:04.56	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flushwq

2. QEMU VM Not-Running Sysbench CPU Test Case –

```
arpita@system: ~/Documents/linux
top - 13:20:31 up 16:12, 3 users, load average: 0.72, 0.98, 1.38
Tasks: 372 total, 2 running, 370 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.9 us, 1.6 sy, 0.0 ni, 93.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7859.1 total, 591.2 free, 5103.3 used, 2164.6 buff/cache
MiB Swap: 2048.0 total, 74.6 free, 1973.4 used, 1716.9 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 27252 arpita    20   0 3739028 217148 49564 R   17.5   2.7   10:00.13 zoom
30408 root      20   0 2969508 823168 49980 S    5.3  10.2   0:55.30 qemu-system-x86
 1838 arpita    9  -11 2818696   9144   4160 S    2.6   0.1   1:28.70 pulseaudio
 2047 arpita    20   0 5420996 190444 72828 S    1.3   2.4   10:48.63 gnome-shell
 4271 arpita    20   0 3358244 542756 36088 S    1.0   6.7   85:20.47 Isolated Web Co
 3419 arpita    20   0 12.6g 464912 141116 S    0.7   5.8   37:01.32 firefox
 3565 arpita    20   0 661600 24544 10928 S    0.7   0.3   4:08.40 Xwayland
 6459 pranav    20   0 32.7g 184896 86224 S    0.7   2.3   6:58.20 chrome
25135 arpita    20   0 2461412 67932 34512 S    0.7   0.8   2:08.06 Isolated Web Co
 1099 libvirt+  20   0 10068 52 0 S    0.3   0.0   0:06.76 dnsmasq
30553 root      20   0 0 0 0 I    0.3   0.0   0:00.27 kworker/1:1-events
30697 arpita    20   0 22028 4132 3264 R    0.3   0.1   0:00.03 top
 1 root      20   0 167080 8260 4868 S    0.0   0.1   0:04.54 systemd
 2 root      20   0 0 0 0 S    0.0   0.0   0:00.02 kthreadd
 3 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 rcu_gp
 4 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 rcu_par_gp
 5 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 slub_flushwq
 6 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 netns
 8 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 kworker/0:0H-events_highpri
10 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 mm_percpu_wq
11 root      20   0 0 0 0 S    0.0   0.0   0:00.00 rcu_tasks_rude_
12 root      20   0 0 0 0 S    0.0   0.0   0:00.00 rcu_tasks_trace
13 root      20   0 0 0 0 S    0.0   0.0   0:01.47 ksoftirqd/0
14 root      20   0 0 0 0 I    0.0   0.0   0:21.59 rcu_sched
15 root      rt  0 0 0 0 S    0.0   0.0   0:00.25 migration/0
16 root     -51  0 0 0 0 S    0.0   0.0   0:00.00 idle_inject/0
18 root      20   0 0 0 0 S    0.0   0.0   0:00.00 cpuhp/0
19 root      20   0 0 0 0 S    0.0   0.0   0:00.00 cpuhp/1
20 root     -51  0 0 0 0 S    0.0   0.0   0:00.00 idle_inject/1
21 root      rt  0 0 0 0 S    0.0   0.0   0:00.42 migration/1
22 root      20   0 0 0 0 S    0.0   0.0   0:10.09 ksoftirqd/1
```

3. Docker Container Running Sysbench CPU Test Case –

```
arpita@system: ~/Documents/linux
top - 13:02:29 up 15:54, 2 users, load average: 6.14, 2.54, 2.22
Tasks: 374 total, 2 running, 372 sleeping, 0 stopped, 0 zombie
%Cpu(s): 98.6 us, 1.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.1 si, 0.0 st
MiB Mem : 7859.1 total, 1409.1 free, 4377.0 used, 2073.0 buff/cache
MiB Swap: 2048.0 total, 125.3 free, 1922.7 used, 2454.0 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
29970 root      20   0 31952 9384 7916 S 374.2   0.1   2:37.13 sysbench
 27252 arpita    20   0 3739040 215544 47952 S   16.9   2.7   6:50.25 zoom
 2047 arpita    20   0 5414824 177340 60300 R    2.6   2.2  10:07.54 gnome-shell
 1838 arpita    9  -11 2818696   9216   4212 S    2.0   0.1   1:01.45 pulseaudio
 3057 arpita    20   0 844372 47052 32104 S    1.7   0.6   0:21.78 gnome-terminal-
25135 arpita    20   0 2460388 63880 34512 S    1.0   0.8   1:59.08 Isolated Web Co
 2172 arpita    20   0 397224 3432 1948 S    0.7   0.0   0:07.82 ibus-daemon
 5715 pranav    20   0 5302200 69496 15792 S    0.7   0.9   2:01.00 gnome-shell
 6459 pranav    20   0 32.7g 180316 80772 S    0.7   2.2   6:52.57 chrome
 646 systemd+  20   0 14824 760 292 S    0.3   0.0   1:54.40 systemd-oomd
 4271 arpita    20   0 3358244 552504 36088 S    0.3   6.9   85:13.14 Isolated Web Co
27803 arpita    20   0 2427540 49408 32652 S    0.3   0.6   0:11.53 Isolated Servic
29252 root      20   0 0 0 0 I    0.3   0.0   0:01.22 kworker/1:2-events
29779 arpita    20   0 22028 4096 3228 R    0.3   0.1   0:00.60 top
 1 root      20   0 167080 8220 4844 S    0.0   0.1   0:04.49 systemd
 2 root      20   0 0 0 0 S    0.0   0.0   0:00.02 kthreadd
 3 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 rcu_gp
 4 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 rcu_par_gp
 5 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 slub_flushwq
 6 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 netns
 8 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 kworker/0:0H-events_highpri
10 root      0 -20 0 0 0 I    0.0   0.0   0:00.00 mm_percpu_wq
11 root      20   0 0 0 0 S    0.0   0.0   0:00.00 rcu_tasks_rude_
12 root      20   0 0 0 0 S    0.0   0.0   0:00.00 rcu_tasks_trace
13 root      20   0 0 0 0 S    0.0   0.0   0:01.44 ksoftirqd/0
14 root      20   0 0 0 0 I    0.0   0.0   0:21.23 rcu_sched
15 root      rt  0 0 0 0 S    0.0   0.0   0:00.25 migration/0
16 root     -51  0 0 0 0 S    0.0   0.0   0:00.00 idle_inject/0
18 root      20   0 0 0 0 S    0.0   0.0   0:00.00 cpuhp/0
19 root      20   0 0 0 0 S    0.0   0.0   0:00.00 cpuhp/1
20 root     -51  0 0 0 0 S    0.0   0.0   0:00.00 idle_inject/1
```

4. Docker Container Not-Running Sysbench CPU Test Case -

```
arpita@system: ~/Documents/linux
top - 13:06:07 up 15:58, 2 users, load average: 0.82, 1.79, 2.01
Tasks: 368 total, 1 running, 367 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.6 us, 1.8 sy, 0.0 ni, 93.6 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7859.1 total, 1352.7 free, 4412.3 used, 2094.1 buff/cache
MiB Swap: 2048.0 total, 126.0 free, 1922.0 used, 2419.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 27252 arpita    20   0 3739028 219092 51496 S  17.9   2.7   7:29.89  zoom
 3419  arpita    20   0   12.6g 494924 139028 S   5.0   6.1  36:40.31  firefox
 1838  arpita     9  -11 2818696   9216   4212 S   2.3   0.1   1:07.39  pulseaudio
25135  arpita    20   0 2460388 64672 34512 S   1.0   0.8   2:01.14  Isolated Web Co
 2047  arpita    20   0 5417808 181472 64832 S   0.7   2.3  10:20.33  gnome-shell
 2262  arpita    20   0   356472   8348   1756 S   0.3   0.1   0:03.53  ibus-extension-
 3057  arpita    20   0   845004 47732 32104 S   0.3   0.6   0:23.93  gnome-terminal-
 4271  arpita    20   0 3358244 559156 36088 S   0.3   6.9  85:14.58  Isolated Web Co
 4702  arpita    20   0 2419852 14144   9716 S   0.3   0.2   0:05.20  Isolated Web Co
 4896  arpita    20   0 2582408 23424  11760 S   0.3   0.3   0:15.02  Isolated Web Co
 6459  pranav    20   0    32.7g 180592 81248 S   0.3   2.2   6:53.61  chrome
 7203  pranav    20   0    32.4g 62108 46316 S   0.3   0.8   2:42.84  chrome
25946  arpita    20   0 2649464 195484 129876 S   0.3   2.4   0:49.74  file:/// Content
30161  arpita    20   0 22028   4108   3240 R   0.3   0.1   0:00.02  top
   1  root      20   0   167080   8316  4924 S   0.0   0.1   0:04.50  systemd
   2  root      20   0         0         0   0 S   0.0   0.0   0:00.02  kthreadd
   3  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  rcu_gp
   4  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  rcu_par_gp
   5  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  slub_flushwq
   6  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  netns
   8  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  kworker/0:0H-events_highpri
  10  root      0 -20         0         0   0 I   0.0   0.0   0:00.00  mm_percpu_wq
  11  root     20  0         0         0   0 S   0.0   0.0   0:00.00  rcu_tasks_rude_
  12  root     20  0         0         0   0 S   0.0   0.0   0:00.00  rcu_tasks_trace
  13  root     20  0         0         0   0 S   0.0   0.0   0:01.45  ksoftirqd/0
  14  root     20  0         0         0   0 I   0.0   0.0   0:21.33  rcu_sched
  15  root      rt  0         0         0   0 S   0.0   0.0   0:00.25  migration/0
  16  root    -51  0         0         0   0 S   0.0   0.0   0:00.00  idle_inject/0
  18  root     20  0         0         0   0 S   0.0   0.0   0:00.00  cpuhp/0
  19  root     20  0         0         0   0 S   0.0   0.0   0:00.00  cpuhp/1
  20  root    -51  0         0         0   0 S   0.0   0.0   0:00.00  idle_inject/1
```

Table of comparison between above four cases -

1. Docker Container CPU -

	Sysbench Running	Sysbench Not-Running
User level CPU usage %	6.6	3.4
Kernel level CPU usage %	3.2	0.9
Idle CPU %	28.5	95.7

2. Host OS CPU Usage - Docker Container -

	Sysbench Running	Sysbench Not-Running
User level CPU usage %	98.6	4.6
Kernel level CPU usage %	1.3	1.8
Idle CPU %	0.0	93.6

3. Host OS CPU Usage - QEMU VM -

	Sysbench Running	Sysbench Not-Running
User level CPU usage %	20.2	4.9
Kernel level CPU usage %	11.6	1.6
Idle CPU %	58.0	93.5

IO Utilization -

Real-time disk utilization is observed using the `iotop` tool. The installation of iotop requires to run below command in both docker container and qemu vm -

```
$ sudo apt install sysstat
```

Using this tool, the disk utilization of Host OS while running Docker Container and QEMU VM is observed during fileio test in docker container using following command -

```
$ sudo iostat -dxzm 1
```

Test Case Commands for fileio – Normal Host FileIO Screenshot –

```

arpita@system: ~/Documents
Device      r/s      rMB/s      rrqm/s      %rrqm      r_await      rareq-sz      w/s      wMB/s      wrqm/s      %wrqm      w_await      wareq-sz      d/s      dMB/s      drqm/s
%drqm d_await dareq-sz      f/s f_await      aqu-sz %util
sda         55.00    1.55      0.00      0.00      0.64      28.80      237.00    0.96      9.00      3.66      0.54      4.15      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.16 5.20

Device      r/s      rMB/s      rrqm/s      %rrqm      r_await      rareq-sz      w/s      wMB/s      wrqm/s      %wrqm      w_await      wareq-sz      d/s      dMB/s      drqm/s
%drqm d_await dareq-sz      f/s f_await      aqu-sz %util

Device      r/s      rMB/s      rrqm/s      %rrqm      r_await      rareq-sz      w/s      wMB/s      wrqm/s      %wrqm      w_await      wareq-sz      d/s      dMB/s      drqm/s
%drqm d_await dareq-sz      f/s f_await      aqu-sz %util
loop4       2.00     0.09     0.00      0.00      1.00      44.50      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.00 1.20
sda         9.00     0.41     0.00      0.00      0.56      46.67      20.00      0.10      6.00      23.08      0.35      5.20      0.00      0.00      0.00
0.00 0.00 0.00      2.00 0.50      0.01 2.80

Device      r/s      rMB/s      rrqm/s      %rrqm      r_await      rareq-sz      w/s      wMB/s      wrqm/s      %wrqm      w_await      wareq-sz      d/s      dMB/s      drqm/s
%drqm d_await dareq-sz      f/s f_await      aqu-sz %util
loop6       2.00     0.09     0.00      0.00      1.00      48.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.00 0.40
sda         6.00     0.27     0.00      0.00      0.83      45.33      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.01 0.40

Device      r/s      rMB/s      rrqm/s      %rrqm      r_await      rareq-sz      w/s      wMB/s      wrqm/s      %wrqm      w_await      wareq-sz      d/s      dMB/s      drqm/s
%drqm d_await dareq-sz      f/s f_await      aqu-sz %util
loop4       2.00     0.17     0.00      0.00      0.00      87.50      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.00 0.80
loop6      10.00     0.57     0.00      0.00      0.70      58.50      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.01 2.00
sda        94.00     3.51     0.00      0.00      0.52      38.21      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
0.00 0.00 0.00      0.00 0.00      0.05 4.00

```

Run Stage – Docker

```

arpira@system: ~/Documents
Device      r/s      rMB/s    rrqm/s    %rrqm    r_await    rareq-sz    w/s      wMB/s    wrqm/s    %wrqm    w_await    wareq-sz    d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop4      3.00     0.12     0.00     0.00     0.01     3.67     40.67     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.01 2.40
loop6      472.00   27.25   0.00     0.00     0.00     1.78     59.12     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.84 102.00
loop8      29.00    1.65     0.00     0.00     0.00     21.83    58.31     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.63 65.60
sda       1030.00  27.41   201.00   16.33    14.76    27.25    742.00     3.00    47.00     5.96     3.97     4.13     0.00     0.00     0.00
0.00 0.00 0.00 209.00 3.34 18.84 96.80

Device      r/s      rMB/s    rrqm/s    %rrqm    r_await    rareq-sz    w/s      wMB/s    wrqm/s    %wrqm    w_await    wareq-sz    d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop4      2.00     0.10     0.00     0.00     0.00     2.50     49.50     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.01 1.20
loop6      581.00   32.96     0.00     0.00     0.00     0.54     58.08     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.31 52.80
loop8      11.00    0.60     0.00     0.00     0.00     1.36     55.91     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.01 2.40
sda       2211.00  52.56   32.00    1.43     2.07    24.34   3887.00    15.98   289.00     6.92     1.21     4.21     0.00     0.00     0.00
0.00 0.00 0.00 1136.00 0.59 9.94 100.00

Device      r/s      rMB/s    rrqm/s    %rrqm    r_await    rareq-sz    w/s      wMB/s    wrqm/s    %wrqm    w_await    wareq-sz    d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop4      80.00    4.12     0.00     0.00     0.00     0.23     52.69     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.02 33.20
loop6      827.00   50.54     0.00     0.00     0.00     0.55     62.57     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.45 72.40
loop8      15.00    0.58     0.00     0.00     0.00     0.67     39.53     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
0.00 0.00 0.00 0.00 0.00 0.01 2.00
sda       2670.00  81.14   152.00    5.39     2.23    31.12   4122.00    16.80   273.00     6.21     1.23     4.17     0.00     0.00     0.00
0.00 0.00 0.00 1182.00 0.59 11.73 100.00

```

Run Stage QEMU

```

arpita@system: ~/Documents
Device            r/s      rMB/s    rrqm/s   %rrqm   r_await  rareq-sz   w/s      wMB/s    wrqm/s   %wrqm   w_await  wareq-sz   d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop6             80.00    4.70     0.00     0.00    0.35    60.11    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    0.03    6.40
sda              361.00   14.18   206.00   36.33    0.52    40.23   23.00     0.18     0.00    0.00    0.13    7.83    0.00     0.00     0.00
0.00    0.00    0.00   12.00    0.25    0.20   30.00

Device            r/s      rMB/s    rrqm/s   %rrqm   r_await  rareq-sz   w/s      wMB/s    wrqm/s   %wrqm   w_await  wareq-sz   d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop4              2.00    0.11     0.00     0.00    1.00    57.50    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    1.60
loop6            103.00    5.94     0.00     0.00    0.36    59.09    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    0.04    9.60
loop8              6.00    0.29     0.00     0.00    0.67    50.33    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    1.20
sda             608.00   13.45   30.00    4.70    0.43    22.64  801.00     7.30     9.00    1.11    0.20    9.34    0.00     0.00     0.00
0.00    0.00    0.00  341.00    0.32    0.53  93.20

Device            r/s      rMB/s    rrqm/s   %rrqm   r_await  rareq-sz   w/s      wMB/s    wrqm/s   %wrqm   w_await  wareq-sz   d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
loop4              2.00    0.06     0.00     0.00    1.00    33.00    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    1.20
loop6             17.00    1.10     0.00     0.00    0.24    66.24    0.00     0.00     0.00    0.00    0.00    0.00    0.00     0.00     0.00
0.00    0.00    0.00    0.00    0.00    1.20
sda             646.00   12.95   30.00    4.44    0.50    20.53  861.00     8.10    19.00    2.16    0.22    9.63    0.00     0.00     0.00
0.00    0.00    0.00  373.00    0.37    0.65  99.60

Device            r/s      rMB/s    rrqm/s   %rrqm   r_await  rareq-sz   w/s      wMB/s    wrqm/s   %wrqm   w_await  wareq-sz   d/s      dMB/s    drqm/s
%drqm d_await dareq-sz f/s f_await aqu-sz %util
sda             464.00    7.25     0.00     0.00    0.39    16.00  936.00     8.52     0.00    0.00    0.21    9.32    0.00     0.00     0.00
0.00    0.00    0.00  396.00    0.35    0.51  99.60

```

Table of I/O Throughput, I/O Latency, and I/O Disk Utilization Host OS

	I/O Throughput	I/O Latency	I/O Disk Utilization
Run Stage - Docker	0.03	0.00	0.20
Run Stage - QEMU	0.50	0.05	0.20

Findings and Conclusion -

1. The host OS disk utilization while running FILEIO test cases of containers was higher than that of QEMU VM.
2. The I/O Throughput, I/O Latency, and I/O Disk Utilization vary based on which stage of test they are captured. The Prepare Stage has the highest value for all three parameters, then comes the Run Stage, and then the Cleanup Stage.

Automation

Vagrant File -

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.

Vagrant.configure("2") do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "bento/ubuntu-20.04-live"
```

```
# Disable automatic box update checking. If you disable this, then
# boxes will only be checked for updates when the user runs
# `vagrant box outdated`. This is not recommended.
# config.vm.box_check_update = false

# Create a forwarded port mapping which allows access to a specific port
# within the machine from a port on the host machine. In the example below,
# accessing "localhost:8080" will access port 80 on the guest machine.
# NOTE: This will enable public access to the opened port
# config.vm.network "forwarded_port", guest: 80, host: 8080

# Create a forwarded port mapping which allows access to a specific port
# within the machine from a port on the host machine and only allow access
# via 127.0.0.1 to disable public access
# config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"

# Create a private network, which allows host-only access to the machine
# using a specific IP.
# config.vm.network "private_network", ip: "192.168.33.10"

# Create a public network, which generally matched to bridged network.
# Bridged networks make the machine appear as another physical device on
# your network.
# config.vm.network "public_network"

# Share an additional folder to the guest VM. The first argument is
# the path on the host to the actual folder. The second argument is
# the path on the guest to mount the folder. And the optional third
# argument is a set of non-required options.
# config.vm.synced_folder "../data", "/vagrant_data"

# Provider Settings
config.vm.provider "virtualbox" do |vb|
  vb.memory = 2048
  vb.cpus = 2
end

# Network Settings
# config.vm.network "forwarded_port", guest: 80, host: 8080
# config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"
# config.vm.network "private_network", ip: "192.168.33.10"
# config.vm.network "public_network"
```

```
# Folder Settings
config.vm.synced_folder ".", "/vagrant_data"

# Provision Settings
config.vm.provision "shell", path: "vagrant_script.sh"

end
```

Dockerfile -

```
FROM arpitaverma03/ubuntu-sysbench

COPY docker_script.sh /docker_script.sh
COPY test-cpu.sh /test-cpu.sh
COPY test-fileio.sh /test-fileio.sh

RUN chmod +x docker_script.sh
RUN chmod +x test-cpu.sh
RUN chmod +x test-fileio.sh

ENTRYPOINT bash docker_script.sh
```

Findings and Conclusion -

1. When the sysbench test is running inside a docker container, the user level cpu usage and kernel level cpu usage increases more than when the sysbench is not running. Moreover, the idle time decreases when sysbench is running in a docker container.
2. When the sysbench test is running in a docker container, the host os cpu usage goes up. The kernel level cpu usage goes from 0.3 to 1.2 when docker container starts running sysbench test and also the idle time decreases. This shows that the docker containers use the kernel level CPUs of host os.
3. When the sysbench test is running in a qemu vm, the host OS kernel-level cpu usage doesn't go up, but user-level goes up. This shows that the qemu vm does not use the kernel level CPUs of host os.

Resources

- The link to the GitHub repository to access the homework – <https://github.com/arpitav03/Cloud-Computing-Course-2023>
- The link to the docker image of created image in this experiment <https://hub.docker.com/repository/docker/arpitaverma03/ubuntu-sysbench/>

Or

```
$ docker pull arpitaverma03/sysbench-ubuntu:version1
```