

# SYSTEM VS OS VIRTUALIZATION

## JAYASRI RAMAKRISHNAN

---

<b>INTRODUCTION</b>	<b>2</b>
<b>ENVIRONMENT</b>	<b>3</b>
<b>QEMU</b>	<b>3</b>
SETUP - INSTALLATION	4
CONFIGURATIONS AND ARGUMENTS:	5
Default	5
Exploring different arguments for qemu	5
<b>DOCKER</b>	<b>9</b>
SETUP - INSTALLATION	9
ENABLING A DOCKER CONTAINER	10
<b>SYSBENCH - DOCKER</b>	<b>11</b>
CPU Tests:	11
Test 1:	11
Test 2:	12
Test 3:	12
User vs Kernel Space Comparison:	13
FileIO Tests	13
Test 1:	13
Test 2:	14
Test 3:	14
Disk Utilization	15
<b>SYSBENCH - QEMU - UBUNTU</b>	<b>15</b>
CPU Tests:	15
Test 1:	15
Test 2:	16
Test 3:	16
User vs Kernel Space Comparison:	17
FileIO Tests	18
Test 1:	18
Test 2:	19

---

---

Test 3: Disk Utilization	19 19
<b>INFERENCE:</b>	<b>21</b>
<b>VAGRANT</b>	<b>22</b>
<b>DOCKERFILE</b>	<b>23</b>

## INTRODUCTION

This report will cover the setup, configuration, operations, benchmarking, comparison and comprehensive results of comparison of two virtualization techniques

- System Virtualization with QEMU
- OS Virtualization via Docker

## ENVIRONMENT

The environment for this assignment is use of **personal computer** with the following configurations



# QEMU

QEMU is an open source machine emulator and a virtualizer. It's a hypervisor and emulates the machine's processor through dynamic binary translation and provides a set of different hardware and device models for the machine, enabling it to run a variety of guest operating systems. For this experiment, we will be using Ubuntu as the guest OS.

## SETUP - INSTALLATION

The below were the steps followed to install QEMU as the hypervisor and have Ubuntu as Guest OS on top of it.

1. Downloaded the .iso file for ubuntu version 20.04\_03
2. Installed homebrew , the package manager which will aid in the installation of QEMU for MAC with intel chip

```
/bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

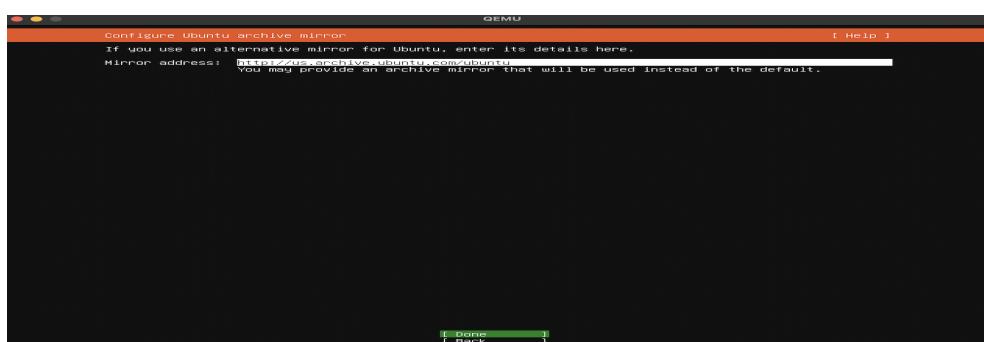
3. Then, installed QEMU using

```
brew install qemu
```

4. Then setup the ubuntu os on qemu using

```
Jayasri:~ vigneshthirunavukkarasu$ qemu-system-x86_64 -hda ubuntu.img -boot d -cdrom  
/Users/vigneshthirunavukkarasu/Downloads/ubuntu-20.04.3-live-server-amd64.iso -m 1536
```

5. Couple of screenshots to show the progress of installation



```
jayasri@jayasriserver:~$  
jayasri@jayasriserver:~$  
di jayasri@jayasriserver:~$
```

## CONFIGURATIONS AND ARGUMENTS:

QEMU can be booted with different arguments.

### Default

When booted with memory of 2G using -m 2G argument without changing CPU, the configurations look like the below.

Command:

```
sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 2G
```

```
jayasri@jayasriserver:~$ screenfetch  
                ./+o+-  
                yyyyy- -yyyyyy+  
                ://+////// -yyyyyyo  
                .++ .:/++++++/- .+sss/`  
                .:+o: /+++++++/:-:-/  
                o:+o:++. . . .-/oo+++++/  
                .:+o:+o/. . . . `+sss00+/  
                .++/+:+oo+o: . . . . /sss000.  
/+++/+:` oo+o . . . . /::--:  
\+/+o+++ o++o . . . . +++++.  
.++.o++++oo+: . . . . /dddhhh.  
.+.o+oo:. . . . ` oddhhhhh+  
\+.++o+o``-` . . . . :ohdhhhhh+  
` :o+++ ` ohhhhhhhyo++os:  
.o: ` syhhhhhhh/.oo++o`  
    /osyuyyyyyo++ooo+++/  
        +oo+++o\:  
            ` oo++.
```

---

## Exploring different arguments for qemu

Command:

```
Jayasri:~ vigneshthirunavukkarasu$ sudo qemu-system-x86_64 -machine accel=hvf ubuntu.img -boot d --cpu Haswell-v4 -smp 2 -m 4G
```

Explaining each below:

Memory -m:

Memory is set to 4G.

CPU -cpu:

If looked at the screenshot in default section, the CPU is QEMU virtual version 2.5+. This can be changed to use any of the below to improve compute or use the host itself.

```
Jayasri:~ vigneshthirunavukkarasu$ qemu-system-x86_64 -cpu help
Available CPUs:
x86 486           (alias configured by machine type)
x86 486-v1        (alias configured by machine type)
x86 Broadwell      (alias configured by machine type)
x86 Broadwell-IBRS (alias of Broadwell-v3)
x86 Broadwell-noTSX (alias of Broadwell-v2)
x86 Broadwell-noTSX-IBRS (alias of Broadwell-v4)
x86 Broadwell-v1   Intel Core Processor (Broadwell)
x86 Broadwell-v2   Intel Core Processor (Broadwell, no TSX)
x86 Broadwell-v3   Intel Core Processor (Broadwell, IBRS)
x86 Broadwell-v4   Intel Core Processor (Broadwell, no TSX, IBRS)
x86 Cascadelake-Server (alias configured by machine type)
x86 Cascadelake-Server-noTSX (alias of Cascadelake-Server-v3)
x86 Cascadelake-Server-v1 Intel Xeon Processor (Cascadelake)
x86 Cascadelake-Server-v2 Intel Xeon Processor (Cascadelake) [ARCH_CAPABILITIES]
x86 Cascadelake-Server-v3 Intel Xeon Processor (Cascadelake) [ARCH_CAPABILITIES, no TSX]
x86 Cascadelake-Server-v4 Intel Xeon Processor (Cascadelake) [ARCH_CAPABILITIES, no TSX]
```

The command for using host would be

-cpu host

After changing cpu, the configuration looks like:

The CPU has changed to Intel Core (Haswell)

```

Last login: Fri Oct  8 20:09:05 UTC 2021 on tty1
jayasri@jayasriserver:~$ screenfetch
                ./+o+-      jayasri@jayasriserver
                yyyyy- -yyyyyy+  OS: Ubuntu 20.04 focal
                ://+//////-yyyyyyo  Kernel: x86_64 Linux 5.4.0-88-generic
                .++ .:/+++++/-.+sss/`  Uptime: 1m
                .:+o: /+++++++/---:/-  Packages: 614
                o:+o:++. ... `` .-/oo+++++/  Shell: bash 5.0.17
                .:+o:+o/.      +sssooo+/  Disk: 4.6G / 12G (40%)
                .++/+:+o+:o:      /sssoooo.  CPU: Intel Core (Haswell, no TSX, IBRS) @ 2x 1.392GHz
/++//+:+o+o:          /::::-:  GPU: Device 1234:1111 (rev 02)
\+/+o+++o++o:        +///.  RAM: 363MiB / 3936MiB
.++.o++o++o+:        /dddhhh.
                .+.o+o:..     `odddhhh+
                \+.++o+o`-` `` .:ohdhhhhh+
                :o+++`ohhhhhhhhyo++os:
                .o: `syhhhhhhh/.oo++o`  /osygggggyo++ooo+++
                +oo++o`\:  `oo++.

```

Cores: -smp

This argument is used to configure the number of cores.

Before setting the cores using -smp command,

```

jayasri@jayasriserver:~$ lscpu
Architecture:           x86_64
CPU op-mode(s):         32-bit, 64-bit
Byte Order:              Little Endian
Address sizes:          40 bits physical, 48 bits virtual
CPU(s):                 1
On-line CPU(s) list:    0
Thread(s) per core:     1
Core(s) per socket:     1
Socket(s):               1
NUMA node(s):            1
Vendor ID:               AuthenticAMD
CPU family:              15
Model:                  107
Model name:              QEMU Virtual CPU version 2.5+

```

---

After Setting smp to 2, we can see from the below screenshot that the cpus and sockets have increased to 2

```
jayasri@jayasriserver:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
Address sizes:         40 bits physical, 48 bits virtual
CPU(s):                2
On-line CPU(s) list:  0,1
Thread(s) per core:   1
Core(s) per socket:   1
Socket(s):             2
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 60
Model name:            Intel Core Processor (Haswell, no TSX, IBRS)
Stepping:               1
CPU MHz:               1391.952
BogoMIPS:              2783.90
L1d cache:             64 KiB
L1i cache:             64 KiB
L2 cache:               8 MiB
L3 cache:               32 MiB
NUMA node0 CPU(s):     0,1
Vulnerability Itlb multihit: KVM: Vulnerable
Vulnerability L1tf:      Mitigation: PTE Inversion
Vulnerability Mds:       Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
Vulnerability Meltdown: Mitigation: PTI
Vulnerability Spec store bypass: Vulnerable
Vulnerability Spectre v1: Mitigation: usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2: Mitigation: Full generic retpoline, STIBP disabled, RSB filling
Vulnerability Srbds:     Unknown: Dependent on hypervisor status
Vulnerability Tsx async abort: Not affected
Flags:                  fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse ss
e2 syscall nx lm constant_tsc rep_good nopl xtopology cpuid pn1 pclmulqdq ssse3 fma cx16 pcid s
se4_1 sse4_2 movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm pt1 fsgsbase bmi1 av
x2 smep bmi2 erms xsaveopt arat
```

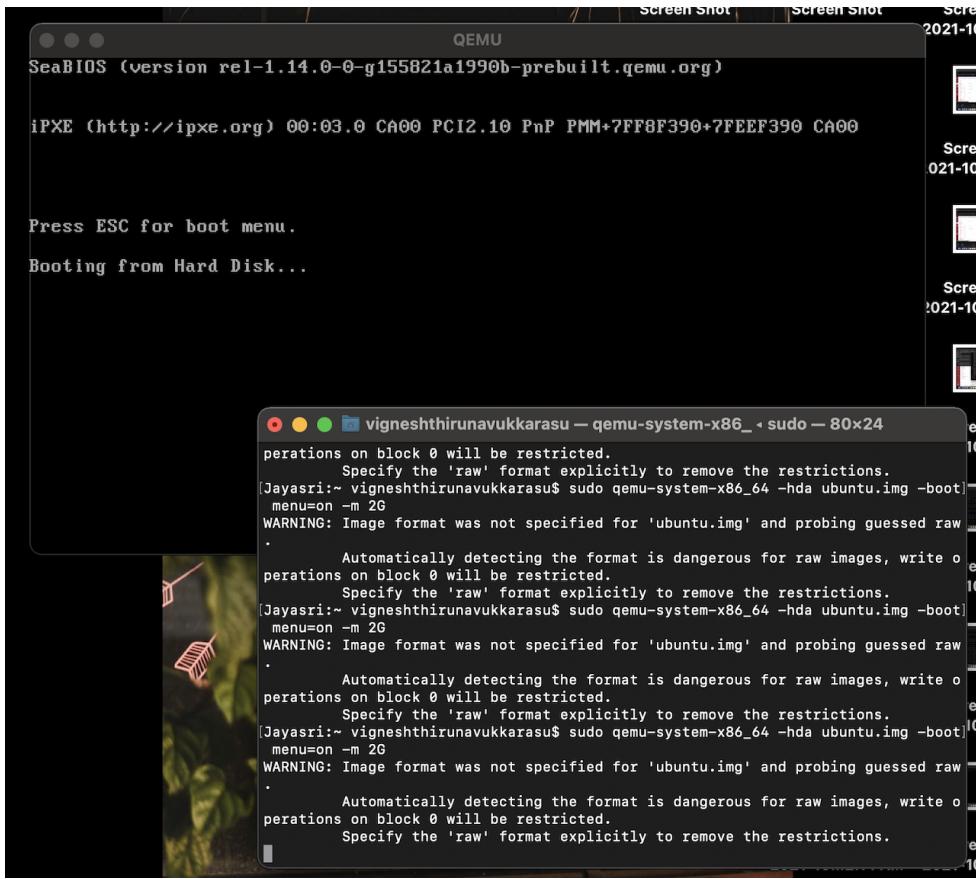
### Spedup -accel:

-accel can be used to speedup qemu. We have used hvf for speedup as shown in the main command.

```
[Jayasri:~ vigneshthirunavukkarasu$ qemu-system-x86_64 -accel help
Accelerators supported in QEMU binary:
[tcg
hax
hvf
```

### Menu:

When given menu=on, it gives the option of from where to boot the image from - cdrom or hard disk or dvd drive. You can see from the below screenshot that "Press ESC for boot MENU" appears



## DOCKER

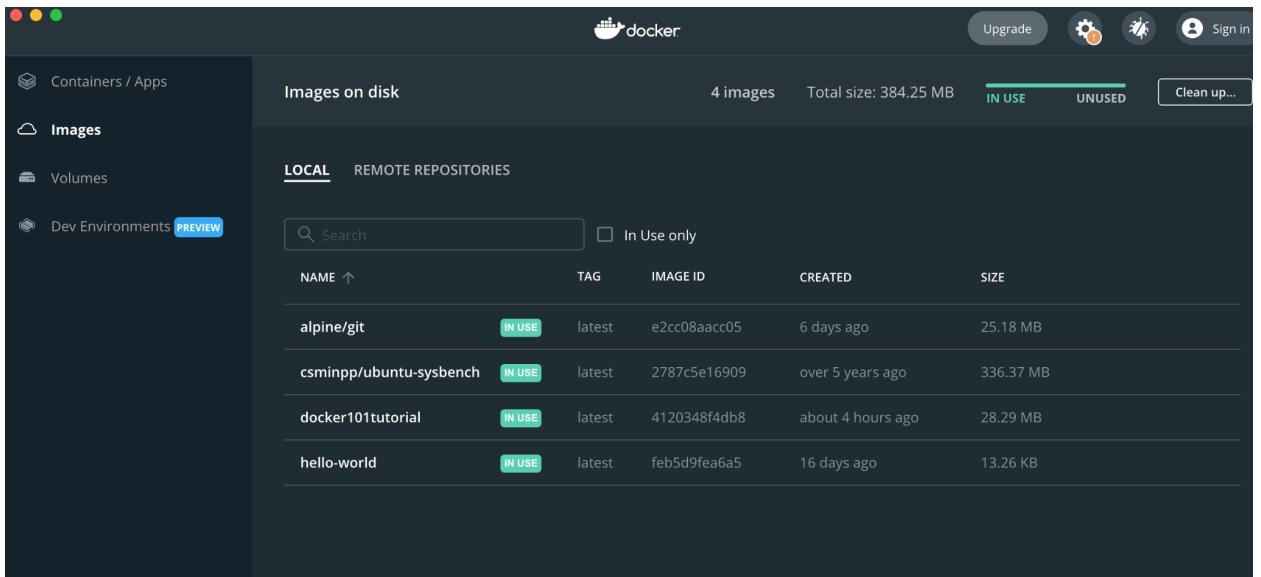
Docker is a set of platforms that enable OS-level virtualization to deliver software in packages called containers. Docker is the most popular container management platform.

### SETUP - INSTALLATION

Downloaded docker desktop executable file from

<https://docs.docker.com/desktop/mac/install/> and followed the wizard for completion.

Screenshot of docker desktop with list of images below



## ENABLING A DOCKER CONTAINER

We are going to use the busybox container as an example to see how to enable a docker container. The main steps are:

- Fetch the image busybox from docker hub repository and store it in the system

```
Jayasri:~ vigneshthirunavukkarasu$ docker pull busybox
```

- Check images to reconfirm

```
Jayasri:~ vigneshthirunavukkarasu$ docker images
```

- Run a command in the container using docker run. Docker finds the image, loads the container and runs the command in the container.

```
Jayasri:~ vigneshthirunavukkarasu$ docker run busybox echo hello
```

- Docker run command with -it flag lets us get an interactive terminal in the container. This will be used later with sysbench
- docker ps -a will show all the containers with container id and status

---

f. Docker rm <id> will remove the container.

Screenshot of all the steps:

```
Jayasri:~ vigneshthirunavukkarasu$ docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
24fb2886d6f6: Pull complete
Digest: sha256:f7ca5a32c10d51aeda3b4d01c61c6061f497893d7f6628b92f822f7117182a57
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
Jayasri:~ vigneshthirunavukkarasu$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
docker101tutorial   latest   4120348f4db8  2 days ago   28.3MB
alpine/git          latest   e2cc08aacc05  8 days ago   25.2MB
hello-world         latest   feb5d9fea6a5  2 weeks ago  13.3kB
busybox              latest   16ea53ea7c65  4 weeks ago  1.24MB
csmnpp/ubuntu-sysbench latest   2787c5e16909  5 years ago  336MB
Jayasri:~ vigneshthirunavukkarasu$ docker run busybox
[Jayasri:~ vigneshthirunavukkarasu$ docker run busybox ech hello
docker: Error response from daemon: OCI runtime create failed: container_linux.go:380: starting container process caused: exec: "ech": executable file not found in $PATH: unknown.
[Jayasri:~ vigneshthirunavukkarasu$ docker run busybox echo hello
hello
[Jayasri:~ vigneshthirunavukkarasu$
```

## SYSBENCH - DOCKER

### CPU Tests:

Have run maxprime cpu test for 3 different test scenarios. Each has run 5 times. Automated scripts and corresponding outputs are available in github repo

Script: cpuTest\_Docker.sh

Output: cpuTest\_docker\_output.txt

The consolidated metrics of the different test runs are below

---

### Test 1:

Docker	20000						
	Events/Sec	min	max	avg	std	95 percentile	
1	468	2.01ms	18446744073678.15ms	2.21ms	0	2.54ms	
2	469	1.96ms	18446744073677.96ms	2.11ms	0	2.47ms	
3	466	2.00ms	18446744073678.05ms	2.12ms	0	2.41ms	
4	462	1.99ms	18446744073677.48ms	2.16ms	0	2.33ms	
5	467	2.01ms	4.82ms	2.14ms	0	2.26ms	

### Test 2:

Docker	50000						
	Events/Sec	Min ms	max	Avg ms	std	95 percentile ms	
1	132	7.08	18446744073683.76ms	7.55	0	8.29	
2	135	7.04	18446744073682.76ms	7.42	0	7.79	
3	134	7.08	18446744073687.00ms	7.45	0	7.89	
4	128	7.05	18446744073686.71ms	7.82	0	8.62	
5	14	7.07	18446744073685.14ms	69.31	0	10.54	

### Test 3:

Docker	90000						
	Events/Sec	Min ms	Max ms	Avg ms	std	95 percentile ms	
1	59	16.27	18446744073699.46ms	17.03	0	18.05	

2	59	16.26	18446744073694.41ms	16.96	0	17.9
3	58	16.28	18446744073697.96ms	17.16	0	18.74
4	59	16.24	18446744073693.59ms	17.02	0	18.08
5	58	16.31	18446744073696.18ms	17.11	0	18.5

## User vs Kernel Space Comparison:

Before running sysbench tests, the user vs kernel space usage were:

```
Processes: 468 total, 2 running, 466 sleeping, 4249 threads
Load Avg: 2.65, 3.70, 4.07 CPU usage: 7.56% user, 5.96% sys, 86.46% idle SharedLibs: 216M resident, 46M data, 17M linkedit. MemRegions: 384138 total, 1513M resident, 56M private, 33M mapped, 11M shared. PhysMem: 7326M used (2613M wired), 864M unused. VM: 4619G vsize, 2305M framework vsize, 2706386299(6805) swapins, 2720469082(0) swapouts. Networks: packets: 71232212/810 in, 33122212/10T out. Disks: 83757640/11T read, 59357815/10T written.

PID COMMAND %CPU TIME #TH #WQ #PORT MEM PURG CMPRS PGRP PPID STATE BOOSTS %CPU_ME %CPU_OHRS UID FAULTS COW MSGSENT MSGRCV SY
139 WindowServer 22.0 15:52:59 17 7 7095 1059M 6948K 280M- 139 1 sleeping *0[1] 0.07120 0.63037 88 176198979+ 275146 1564889474+ 622770618+ 15
51585 Adobe CEF He 17.6 09:36:25 28 1 328- 258M+ 0B 134M 51563 51563 sleeping *0[1] 0.00000 0.00000 501 54086483+ 3123392+ 49870572+ 17686684+ 59
22834 com.docker.h 14.3 06:26:54 14 1 38 3730M 0B 1677M 22761 22830 sleeping *0[1] 0.00000 0.00000 501 745483177+ 469 1794 1356 31
12768 top 8.1 00:02:12 1/1 0 27 7332K 0B 12768 3299 running *0[1] 0.00000 0.00000 0 8763+ 99 1017023+ 508498+ 26
0 kernel_task 4.6 12:08:23 221/8 0 0 577M 0B 0B 0 0 running 0[0] 0.00000 0.00000 0 2978905 2024 1933051784+ 1832839868+ 0
854 Notion Help 4.3 52:04:69 18 1 177+ 160M- 0B 136M- 518 518 sleeping *0[1] 0.00000 0.00000 501 26725681+ 2019 11441753+ 5658527+ 12
51577 Adobe CEF He 3.8 01:58:46 8 1 182 86M 0B 21M 51563 51563 sleeping *0[1] 0.34493 0.00000 501 27398453+ 2169 47608033+ 12552828+ 44
12769 screencaptur 3.0 00:00:68 2 1 55 3656K+ 620K 0B 547 547 sleeping *0[519+]
0.09694 0.00000 501 29311+ 154 6252+ 2123+ 37
```

While running sysbench tests, the user vs kernel space usage were:

```
Processes: 464 total, 4 running, 1 stuck, 459 sleeping, 4365 threads
Load Avg: 3.12, 3.50, 4.32 CPU usage: 7.43% user, 19.39% sys, 73.17% idle SharedLibs: 176M resident, 40M data, 16M linkedit. MemRegions: 384138 total, 1513M resident, 56M private, 33M mapped, 11M shared. PhysMem: 7596M used (2649M wired), 594M unused. VM: 4587G vsize, 2305M framework vsize, 2644070235(55889) swapins, 2657614 swapouts. Networks: packets: 71232212/810 in, 33122212/10T out. Disks: 82392425/11T read, 58078367/10T written.

PID COMMAND %CPU TIME #TH #WQ #PORTS MEM PURG CMPRS PGRP PPID STATE BOOSTS %CPU_ME
22834 com.docker.h 100.4 05:37:59 14/1 1 38 3730M 0B 1675M- 22761 22830 running *0[1] 0.00000
548 Finder 14.6 12:28:87 12/1 7 1341 239M- 0B 212M- 548 1 running 0[10512+] 21.0540
11307 com.apple.qu 10.0 00:00:14 6 5 94+ 2116K- 32K+ 0B 11307 1 stuck *0[1+] 3.96176
51585 Adobe CEF He 7.4 09:30:49 23 2 303+ 261M+ 0B 244M- 51563 51563 sleeping *0[1] 0.00000
```

As you can see, the cpu utilization by docker sysbench test run was at 100% and 7.43% of user space utilized vs 19.39% of kernel space utilized. The system space utilization has gone up.

## FileIO Tests

The fileio tests were performed for three different modes with 16 threads and 1GB filesize

### Test 1:

---

Docker	rndwr					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	72.75	4656	0.02	0.15	0.62	91.93
2	57.67	3691	0.02	0.16	22.69	72.73
3	44.683	2859.74	0.02	0.13	7.29	79.52
4	42.87	2745	0.02	0.18	26.02	104.92
5	67.12	4296.22	0.02	0.18	8.94	91.78

### Test 2:

Docker	seqrd					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	1246	76437	0.01	0.15	145	451
2	1134	75643	0.01	0.16	125	645
3	996	76437	0.01	0.15	312	342
4	1276	74861	0.01	0.18	145	621
5	1134	73281	0.01	0.16	145	451

### Test 3:

Docker	seqrewr					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	200.4	12825.63	0.03	0.78	228.79	466.62
2	201	12906	0.03	0.71	659	475
3	151	9668	0.04	1.35	323	368

---

4	215	13767	0.03	0.71	414	613
5	186	11913	0.03	0.56	226	491

## Disk Utilization

The disk utilization before running docker fileio tests was:

```
[Jayasri:~ vigneshthirunavukkarasu$ iostat
              disk0                  disk2
      KB/t  tps  MB/s      KB/t  tps  MB/s
  159.64 161 25.10    61.29  0  0.00
Jaya@Jaya-Latitude-E5450:~$
```

```
Processes: 445 total, 2 running, 1 stuck, 442 sleeping
Load Avg: 11.05, 10.28, 6.74  CPU usage: 9.38% user,
PhysMem: 6855M used (2620M wired), 1334M unused. VM:
Disks: 88924620/12T read, 63100202/11T written.
```

While running sysbench tests,

## SYSBENCH - QEMU - UBUNTU

### CPU Tests:

Have run maxprime cpu test for 3 different test scenarios. Each has run 5 times. Automated scripts and corresponding outputs are available in github repo

Script: benchmarkcpu.sh

Output: cpuTest\_QEMU\_Output.txt

---

The consolidated metrics of the different test runs are below

### Test 1:

QEMU	20000					
	Events/Sec	Min ms	Max ms	Avg ms	std	95 percentile
1	140	6.63	16.79	7.1	0	8.13
2	141	6.61	13.53	7.03	0	7.56
3	134	6.62	6.11	7.43	0	9.22
4	138	6.62	28.81	71.9	0	8.13
5	138	6.67	23.88	7.19	0	7.98

### Test 2:

QEMU	50000					
	Events/Sec	min	max	avg	std	95 percentile
1	41.34	23.13	43.24	24.11	0	25.28
2	41.32	22.9	30.65	24.1	0	25.28
3	41.24	23.04	33.52	24.16	0	25.28
4	41.2	23.15	43.63	24.17	0	25.28
5	41.34	23.3	28.61	24.1	0	24.83

### Test 3:

QEMU	90000					
------	-------	--	--	--	--	--

	Events/Sec	Min ms	Max ms	avgms	std	95 percentile
1	18.59	52.15	62.32	53.63	0	55.82
2	18.75	51.61	58.11	53.2	0	54.83
3	18.24	51.99	167.36	54.67	0	55.82
4	18.63	52.17	61.77	53.5	0	54.83
5	18.54	52.16	86.25	53.79	0	54.83

## User vs Kernel Space Comparison:

Before running sysbench test in QEMU, the host OS's user vs kernel space CPU usage % was:

```
Processes: 468 total, 2 running, 466 sleeping, 4249 threads
Load Avg: 2.65, 3.70, 4.07 CPU usage: 7.56% user, 5.96% sys, 86.46% idle SharedLibs: 216M resident, 46M data, 17M linkedit. MemRegions: 7326M used (2613M wired), 864M unused. VM: 4619G vsize, 2305M framework vsize, 2706386299(6805) swapins, 2720469082(0) swapouts. Disks: 83757640/11T read, 59357815/10T written.
```

PID	COMMAND	%CPU	TIME	#TH	#WQ	#PORT	MEM	PURG	CMPRS	PGRP	PPID	STATE	BOOSTS	%CPU_ME	%CPU_OTHRS	UID	FAUL
139	WindowServer	22.0	15:52:59	17	7	7095	1059M	6940K	280M-	139	1	sleeping	*0[1]	0.07120	0.63037	88	1761
51585	Adobe CEF He	17.6	09:36:25	28	1	320-	258M+	0B	134M	51563	51563	sleeping	*0[1]	0.00000	0.00000	501	5408
22834	com.docker.h	14.3	06:26:54	14	1	38	3730M	0B	1677M	22761	22830	sleeping	*0[1]	0.00000	0.00000	501	7454
12768	top	8.1	00:02:12	1/1	0	27	7332K	0B	0B	12768	3299	running	*0[1]	0.00000	0.00000	0	8763
0	kernel_task	4.6	12:08:23	221/8	0	0	577M	0B	0B	0	0	running	0[0]	0.00000	0.00000	0	2978
854	Notion Help	4.3	52:04:69	18	1	177+	160M-	0B	136M-	518	518	sleeping	*0[1]	0.00000	0.00000	501	2672
51577	Adobe CEF He	3.8	01:58:46	8	1	182	86M	0B	21M	51563	51563	sleeping	*0[1]	0.34493	0.00000	501	2739
12769	screencaptur	3.0	00:00:68	2	1	55	3656K+	620K	0B	547	547	sleeping	*0[519+]	0.09694	0.00000	501	2931

And in guest OS ubuntu:

```
top - 05:26:58 up 2 days, 12:50, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 87 total, 1 running, 86 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.7 us, 1.0 sy, 0.0 ni, 98.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1987.7 total, 988.7 free, 128.2 used, 870.8 buff/cache
MiB Swap: 1738.0 total, 1738.0 free, 0.0 used. 1680.4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
11757	jayasri	20	0	9248	3876	3220	R	1.0	0.2	0:01.85	top
523	root	rt	0	280308	18112	8200	S	0.3	0.9	3:21.92	multipathd
11652	root	20	0	0	0	0	I	0.3	0.0	0:14.59	kworker/0:0-events
1	root	20	0	168736	12636	8100	S	0.0	0.6	5:25.11	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:04.75	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
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-kblockd
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
9	root	20	0	0	0	0	S	0.0	0.0	3:03.14	ksoftirqd/0

While running sysbench, utilization in host OS:

```
Processes: 475 total, 3 running, 472 sleeping, 4301 threads
Load Avg: 3.16, 3.46, 3.91 CPU usage: 16.72% user, 3.1% sys, 80.25% idle SharedLibs: 245M resident, 51M data, 19M linkedit. MemRegions: 383865 total,
PhysMem: 8049M used (2663M wired), 141M unused. VM: 4653G vsize, 2305M framework vsize, 2708848889(1018) swapins, 2722788829(0) swapouts. Networks: pack
Disks: 83816917/11T read, 59405151/10T written.

PID COMMAND %CPU TIME #TH #WQ #PORTS MEM PURG CMPRS PGRP PPID STATE BOOSTS %CPU_ME %CPU_OTHR UID FAULTS COW
69932 qemu-system- 100.6 02:16:27 8/1 1 237 3300M+ 93M- 3073M- 69928 69928 running *0[5770] 0.01824 0.00000 0 172178424+ 1108
139 WindowServer 13.6 15:53:16 16 6 7892+ 1093M 5376K 273M- 139 1 sleeping *0[1] 0.04261 0.33025 88 176235287+ 2751
51585 Adobe CEF He 9.1 09:36:39 23 2 303 261M+ 0B 155M- 51563 51563 sleeping *0[1] 0.00000 0.00000 501 54105413+ 3126
12880 top 6.6 00:01:37 1/1 0 27 7376K+ 0B 0B 12880 3299 running *0[1] 0.00000 0.00000 0 5974+ 91
0 kernel_task 3.5 12:08:42 221/8 0 0 543M 0B 0B 0 0 running 0[0] 0.00000 0.00000 0 2978909 2024
12882 screencaptur 3.0 00:00:12 3 2 58 3620K+ 620K 0B 547 547 sleeping *0[149+] 0.10552 0.00000 501 10019+ 154
22834 com.docker.h 3.0 06:26:59 14 1 38 3738M 0B 1677M 22761 22830 sleeping *0[1] 0.00000 0.00000 501 745749067+ 469
51577 Adobe CEF He 2.2 01:58:49 8 1 182 86M 0B 21M- 51563 51563 sleeping *0[1] 0.25669 0.00000 501 27415383+ 2169
61892 RdrCEF Helpe 1.4 01:50:05 14 1 136 281M 0B 276M- 39730 39730 sleeping *0[4] 0.00000 0.00000 501 1583055+ 2153
512 Terminal 1.4 06:44:77 11 5 429 130M+ 33M 39M- 512 1 sleeping *0[9864] 0.04498 0.00000 501 7929382+ 3817
```

The cpu utilization has gone upto 100% and the user space utilization has increased to 16.72%

In Guest OS,

```
top - 05:27:34 up 2 days, 12:51, 1 user, load average: 0.22, 0.05, 0.02
MiB Mem : 1987.7 total, 986.4 free, 130.5 used, 870.8 buff/cache
%Cpu(s): 94.5 us, 5.5 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
11761 jayasri 20 0 9248 3876 3224 R 4.5 0.2 0:00.99 top
11762 jayasri 20 0 33176 10076 8272 S 52.3 0.5 0:01.61 sysbench
11758 jayasri 20 0 7024 3316 2968 S 4.9 0.2 0:00.95 benchmarkcpu.sh
11761 jayasri 20 0 9248 3876 3224 R 1.0 0.2 0:01.02 top
11652 root 20 0 0 0 0 I 0.6 0.0 0:14.76 kworker/0:0-events
1 root 20 0 168736 12636 8100 S 0.0 0.6 5:25.13 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:04.75 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 T 0.0 0.0 0:00.00 rcu_par_dn
```

The user space has shot up by 94.5% from 0.7%

## FileIO Tests

File IO tests were for three modes with 16 threads and 1GB total file size

### Test 1:

QEMU	rndwr					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	16.46	2394	0.03	6.26	121.86	118.78

---

2	16.24	2369	0.03	6.38	336.74	169
3	16	2330	0.03	6.5	95.53	133.93
4	16.83	2451	0.03	6.13	128.22	110.31
5	16.17	2353	0.03	6.42	99.28	147.69

### Test 2:

QEMU	seqrd					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	321.85	20669	0.01	0.58	1485	536
2	620	40393	0.01	0.24	189	309
3	197.25	12911.3	0.01	0.75	9875	1732.92
4	634.81	49670	0.01	0.24	134.3	293.44
5	686	44011	0.01	0.22	127.47	251.56

### Test 3:

QEMU	seqrewr					
	Transfer Rate Mb/sec	Requests/sec	min ms	avg ms	max ms	std
1	6.63	961.24	0.04	12.82	1893.64	154.92
2	10.59	1537	0.04	9.11	372.52	174.95
3	16.39	2619	0.04	5.87	224.71	242.95
4	24	4722	0.03	3.18	135.99	185.6
5	37.34	5831	0.03	2.68	206.73	242

## Disk Utilization

The disk utilization before running of filleo tests in host os was

```
[Jayasri:~ vigneshthirunavukarasu$ iostat
          disk0                  disk2
      KB/t  tps  MB/s      KB/t  tps  MB/s
  159.64 161 25.10    61.29    0  0.00
Jayasri:~ vigneshthirunavukarasu$
```

```
Processes: 445 total, 2 running, 1 stuck, 442 sleeping
Load Avg: 11.05, 10.28, 6.74  CPU usage: 9.38% user,
PhysMem: 6855M used (2620M wired), 1334M unused. VM:
Disks: 88924620/12T read, 63100202/11T written.
```

And in Guest OS:

vg-cpu	%user	%nice	%system	%iowait	%steal	%idle	
	1.05	0.08	0.57	0.06	0.00	98.26	
device	tps	KB_read/s	KB_wrtn/s	KB_dscd/s	KB_read	KB_wrtn	KB_dscd
dm-0	1.95	2.15	96.05	21.43	491396	21988352	4906404
d0	0.00	0.00	0.00	0.00	72	0	0
loop0	0.01	0.01	0.00	0.00	1796	0	0
loop1	0.00	0.00	0.00	0.00	349	0	0
loop2	0.00	0.00	0.00	0.00	1050	0	0
loop3	0.00	0.00	0.00	0.00	341	0	0
loop4	0.00	0.00	0.00	0.00	1063	0	0
loop5	0.06	0.06	0.00	0.00	14571	0	0
loop6	0.06	0.06	0.00	0.00	14705	0	0
loop7	0.00	0.00	0.00	0.00	5	0	0
da	1.67	2.19	95.88	25.31	501917	21949608	5795116

```
top - 14:58:26 up 2 days, 15:36, 1 user, load average: 0.07, 0.04, 0.03
Tasks: 90 total, 2 running, 88 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 2.3 sy, 0.0 ni, 97.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1987.7 total, 1043.1 free, 184.5 used, 760.0 buff/cache
MiB Swap: 1738.0 total, 1737.7 free, 0.3 used. 1627.1 avail Mem
```

While running sysbench tests,

```

Processes: 426 total, 3 running, 423 sleeping, 4169 threads          08:14:4
Load Avg: 4.28, 5.10, 5.61 CPU usage: 18.74% user, 6.16% sys, 75.8% idle
SharedLibs: 159M resident, 30M data, 12M linkedit.
MemRegions: 368087 total, 2390M resident, 57M private, 599M shared.
PhysMem: 7810M used (2780M wired), 380M unused.
VM: 4379G vsize, 2305M framework vsize, 2906241365(40287) swapins, 2920417583(0) swaps
Networks: packets: 71589185/82G in, 33734833/12G out.
Disks: 89329433/12T read, 63498593/11T written.

PID      COMMAND      %CPU    TIME      #TH      #WQ      #PORTS   MEM      PURG      CMPRS      PGRP
69932  qemu-system-  98.5   02:43:04  8/1       1      237      3356M  2308K+  2065M-  69928

```

In guest OS,

```

top - 15:42:53 up 2 days, 16:10, 1 user, load average: 2.64, 0.70, 0.82
MiB Mem : 1987.7 total, 1121.5 free, 185.9 used, 680.3 buff/cache
%Cpu(s): 0.7 us, 93.2 sy, 0.0 ni, 0.0 id, 5.3 wa, 0.0 hi, 0.7 si, 0.0 st
MiB Mem : 1987.7 total, 736.8 free, 186.7 used, 1064.2 buff/cache
MiB Swap: 1738.0 total, 1737.7 free, 0.3 used. 1621.1 avail Mem
Maximum tasks = 0, change to (0 is unlimited) WARNING: the --test option is deprecated. Use command line without any options. 0      0 S     1.2    0.0   1:27.18 jbd2/dm-0-8
_ 13395 jayasri 20 0 33384 10324 8704 R 81.4 0.5 0:05.18 sysbench

```

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle	
	1.07	0.08	0.75	0.07	0.00	98.04	
Device	tps	KB_read/s	KB_wrtn/s	KB_dscd/s	KB_read	KB_wrtn	KB_dscd
dm-0	3.38	2.36	186.34	21.24	544624	43052900	4906404
fd0	0.00	0.00	0.00	0.00	72	0	0
loop0	0.01	0.01	0.00	0.00	1796	0	0
loop1	0.00	0.00	0.00	0.00	349	0	0
loop2	0.00	0.00	0.00	0.00	1050	0	0
loop3	0.00	0.00	0.00	0.00	341	0	0
loop4	0.00	0.00	0.00	0.00	1063	0	0
loop5	0.06	0.06	0.00	0.00	14571	0	0
loop6	0.06	0.06	0.00	0.00	14705	0	0
loop7	0.00	0.00	0.00	0.00	5	0	0
sda	3.06	2.40	186.10	25.08	555145	42997180	5795116

The usage of physical memory has gone up from 6855M to 7810M in host OS

## INFERENCE:

1. QEMU boots up faster when memory is increased, cores are increased and acceleration is set for speedup
2. The CPU events per second is higher in OS virtualization compared to System Virtualization. Its around 460 events/sec in Docker vs 123 events/sec in QEMU

- 
- 3. The CPU events per second decreases when the maxprime limit is set higher for both System and OS virtualization
  - 4. Even with the decrease, OS virtualization performs better even for higher maxprime values
  - 5. In terms of user space vs kernel space utilization, qemu sysbench tests seem to burden the user space. From the above stats, its visible that sysbench tests took about 94% of userspace from 0.7%
  - 6. Docker sysbench tests don't have that drastic an impact on user vs kernel space utilization. System space % went up marginally upto 19.39%
  - 7. The transfer rate and the events per second is higher in OS virtualization compared to System virtualization
  - 8. The throughput increases when we increase the num of threads for fileio tests.
  - 9. The disk and physical memory usage of host OS increase dramatically when QEMU fileio tests run
  - 10. OS virtualization performs better compared to System virtualization as seen from the cpu tests and fileio tests

## VAGRANT

Vagrant is an open source tool for creating and configuring VMs. In order to automate the bring up of QEMU and ubuntu as guest os in it, we require virtualization management library libvirt.

- 1. Install QEMU - Refer QEMU chapter above
- 2. Install Vagrant
  - a. Brew install vagrant

```

Jayasri:~ vigneshthirunavukarasu$ brew install vagrant
Updating Homebrew...
==> Homebrew is run entirely by unpaid volunteers. Please consider donating:
  https://github.com/Homebrew/brew#donations
==> Auto-updated Homebrew!
Updated 1 tap (homebrew/core).
==> New Formulae
cassandra@3      cpufetch          fheroes2        k2tf           lilypond
colima            feroxbuster       g2o             libnghttp2    11vm@12
==> Updated Formulae
Updated 731 formulae.
==> Deleted Formulae
pandoc-citeproc
^[
==> Tapping homebrew/cask
Cloning into '/usr/local/Homebrew/Library/Taps/homebrew/homebrew-cask'...
remote: Enumerating objects: 597541, done.
Receiving objects: 100% (597541/597541), 266.28 MiB | 1.66 MiB/s, done.
remote: Total 597541 (delta 0), reused 0 (delta 0), pack-reused 597541
Resolving deltas: 100% (422332/422332), done.
Tapped 3924 casks (4,004 files, 285.4MB).
==> Downloading https://releases.hashicorp.com/vagrant/2.2.18/vagrant_2.2.18_x86_64.dmg
#####
100.0%
==> Installing Cask vagrant
==> Running installer for vagrant; your password may be necessary.
Package installers may write to any location; options such as `--appdir` are ignored.
[Password:
Sorry, try again.
[Password:
Sorry, try again.
[Password:
installer: Package name is Vagrant
installer: Installing at base path /
installer: The install was successful.
🍺 vagrant was successfully installed!

```

3. Install libvirt
4. Find the box compatible with libvirt from  
<https://app.vagrantup.com/boxes/search?page=1&provider=libvirt>
5. Vagrant file shared in github: VagrantFile

## DOCKERFILE

1. Dockerfile placed in github repo
2. Docker build container

```
"Dockerfile" 4L, 139C written
jayasri@jayasriserver:~$ sudo docker build .
Sending build context to Docker daemon 755.2kB
Step 1/4 : FROM csminpp/ubuntu-sysbench
--> 2787c5e16909
Step 2/4 : LABEL "about"="Automating sysbench test"
--> Using cache
--> f5ec0262f934
Step 3/4 : COPY sysbenchTestDocker.sh /sysbenchTest.sh
--> 4596ffbb3925
Step 4/4 : CMD ["/sysbenchTest.sh"]
--> Running in 50329b2403d5
INFO[2021-10-15T22:37:25.7384510302] Layer sha256:adf99df37d74896b0a2df26e5000b0168259a694c5d3439fad8b4e8023549d04 clean
Removing intermediate container 50329b2403d5
--> 3259a62d94ba
Successfully built 3259a62d94ba
jayasri@jayasriserver:~$
```