COEN 241
Cloud Computing
HOMEWORK-1
SYSTEM VS OS VIRTUALIZATION
REPORT

Piyush Kulkarni
ID: 1629006

—

DOCKER UBUNTU CONTAINER SETUP

After installing docker, the following commands pull the ubuntu docker image from dockerhub.

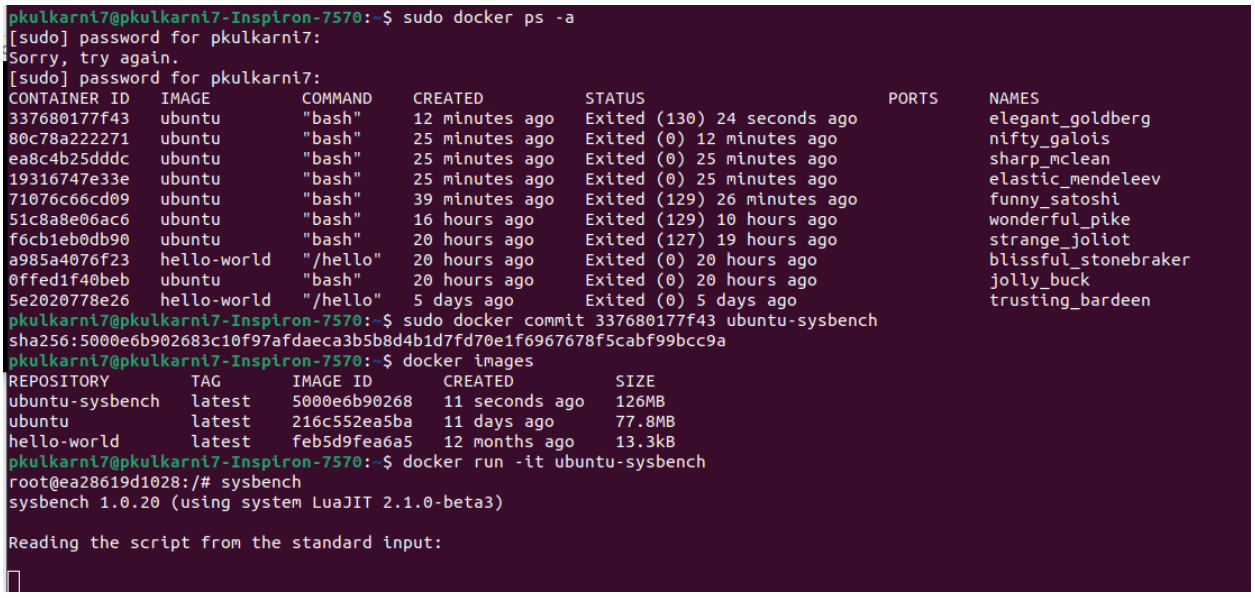
```
docker run -it ubuntu bash
apt-get update
```

Sysbench was installed using:

```
apt install sysbench
```

Later, in order to save the modified image, the following commands were used.

```
sudo docker ps -a
sudo docker commit <image-id> <new-image-name>
```



QEMU UBUNTU SETUP

The following commands were used to setup QEMU

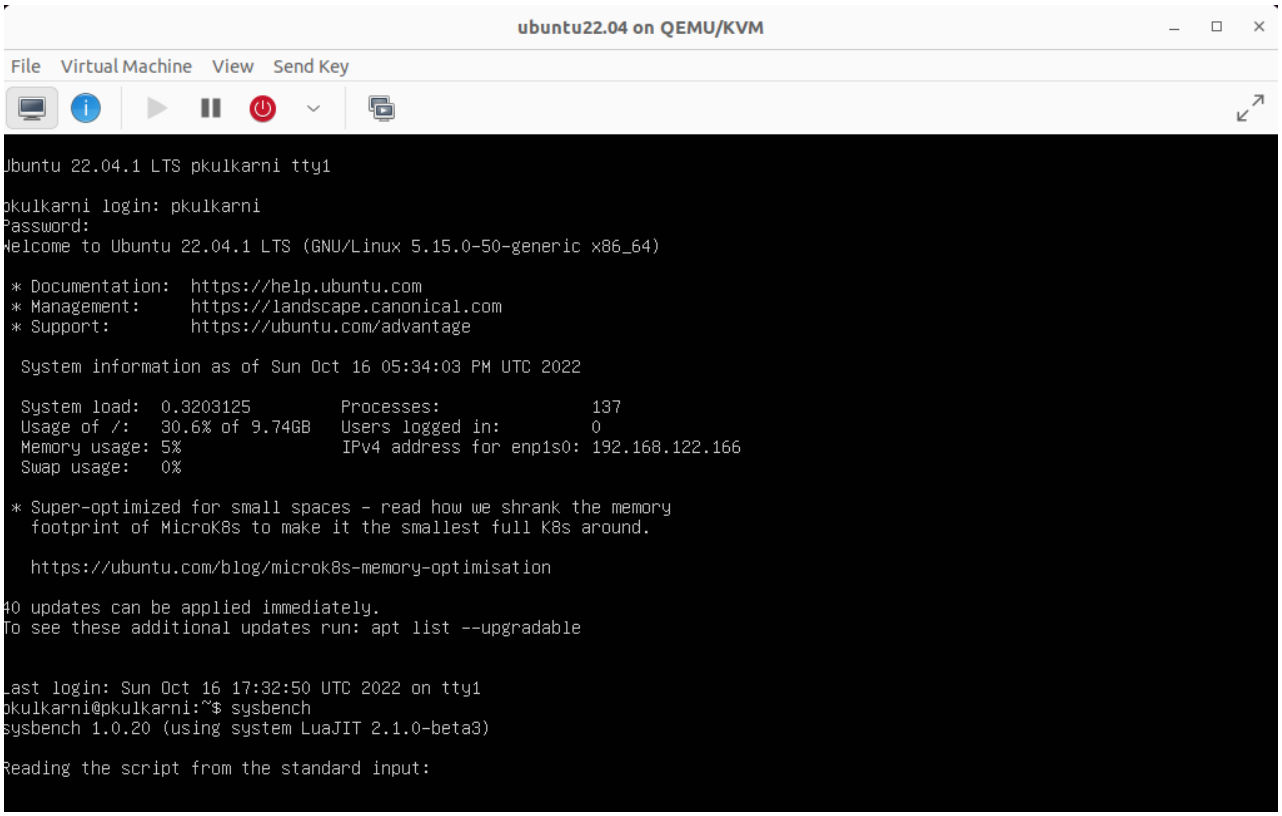
```
sudo apt-get install qemu
sudo qemu-img create ubuntu.img 10G -f qcow2

sudo apt install libvirt-daemon
sudo systemctl enable libvirtd
sudo systemctl start libvirtd

sudo qemu-system-x86_64 -hda ubuntu.img -boot d -cdrom.
/ubuntu-22.04.1-live-server-amd64.iso -m 4092 -boot strict=on
```

Later, I decided to install virt-manager to more conveniently manage VMs. It provides a GUI and connects to the QEMU Ubuntu VM.

```
sudo apt install virt-manager
```

sysbench was installed using the commands

```
sudo apt update
sudo apt install sysbench
```



SYSTEM CONFIGURATIONS

```
root@3c05a35d1e83:~# lscpu
Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         39 bits physical, 48 bits virtual
  Byte Order:            Little Endian
CPU(s):                  8
  On-line CPU(s) list:   0-7
Vendor ID:               GenuineIntel
  Model name:            Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz
    CPU family:          6
    Model:               142
    Thread(s) per core:  2
    Core(s) per socket:  4
    Socket(s):           1
    Stepping:            10
    CPU max MHz:         3400.0000
    CPU min MHz:         400.0000
    BogoMIPS:            3600.00
    Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2
                          ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology
                         nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm
                          pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowpref
                         etch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad
                         fsgsbase tsc_adjust sgx bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx smap clflushopt intel_pt xsaveopt
                         xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d arc
                         h_capabilities
Virtualization features:
  Virtualization:        VT-x
Caches (sum of all):
  L1d:                   128 KiB (4 instances)
  L1i:                   128 KiB (4 instances)
  L2:                    1 MiB (4 instances)
  L3:                    6 MiB (1 instance)
NUMA:
  NUMA node(s):          1
  NUMA node0 CPU(s):     0-7
Vulnerabilities:
  Itlb multihit:         KVM: Mitigation: Split huge pages
  L1tf:                  Mitigation; PTE Inversion; VMX conditional cache flushes, SMT vulnerable
  Mds:                   Mitigation; Clear CPU buffers; SMT vulnerable
  Meltdown:              Mitigation; PTI
  Mmio stale data:       Mitigation; Clear CPU buffers; SMT vulnerable
  Retbleed:              Mitigation; IBRS
  Spec store bypass:     Mitigation; Speculative Store Bypass disabled via prctl and seccomp
  Spectre v1:            Mitigation; usercopy/swapgs barriers and __user pointer sanitization
  Spectre v2:            Mitigation; IBRS, IBPB conditional, RSB filling
  Srbds:                 Mitigation; Microcode
  Tsx async abort:       Not affected
root@3c05a35d1e83:~#
```

```
root@3c05a35d1e83:~# free -m
               total        used        free      shared  buff/cache   available
Mem:            7688        3959         518         765        3209        2664
Swap:           3814         266        3548
root@3c05a35d1e83:~# uname -mrs
Linux 5.15.0-48-generic x86_64
root@3c05a35d1e83:~#
```

Docker Ubuntu Container

```
pkulkarni@pkulkarni:~$ lscpu
Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         39 bits physical, 48 bits virtual
  Byte Order:            Little Endian
CPU(s):                  2
  On-line CPU(s) list:   0,1
Vendor ID:               GenuineIntel
  Model name:            Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz
    CPU family:          6
    Model:               142
    Thread(s) per core:  1
    Core(s) per socket:  1
    Socket(s):           2
    Stepping:            10
    BogoMIPS:            3600.00
    Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ss sy
                         scall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon rep_good nopl xtopology cpuid tsc_known_freq pni p
                         clmulqdq vmx ssse3 fma cx16 pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave av
                         x f16c rdrand hypervisor lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pti ssbd ibrs ibpb stibp
                         tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust sgx bmi1 avx2 smep bmi2 erms invpcid m
                         px rdseed adx smap clflushopt xsaveopt xsavec xgetbv1 xsaves arat umip md_clear arch_capabilities
Virtualization features:
  Virtualization:        VT-x
  Hypervisor vendor:     KVM
  Virtualization type:   full
Caches (sum of all):
  L1d:                   64 KiB (2 instances)
  L1i:                   64 KiB (2 instances)
  L2:                    8 MiB (2 instances)
  L3:                    32 MiB (2 instances)
NUMA:
  NUMA node(s):          1
  NUMA node0 CPU(s):     0,1
Vulnerabilities:
  Itlb multihit:         Not affected
  L1tf:                  Mitigation; PTE Inversion; VMX flush not necessary, SMT disabled
  Mds:                   Mitigation; Clear CPU buffers; SMT Host state unknown
  Meltdown:              Mitigation; PTI
  Mmio stale data:       Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
  Retbleed:              Mitigation; IBRS
  Spec store bypass:     Mitigation; Speculative Store Bypass disabled via prctl and seccomp
  Spectre v1:            Mitigation; usercopy/swapgs barriers and __user pointer sanitization
  Spectre v2:            Mitigation; IBRS, IBPB conditional, RSB filling, PBRSB-eIBRS Not affected
  Srbds:                 Unknown: Dependent on hypervisor status
  Tsx async abort:       Not affected
```

```
  Tsx async abort:       Not affected
pkulkarni@pkulkarni:~$ free -m
               total        used        free      shared  buff/cache   available
Mem:            3923         189        3386           1         348        3509
Swap:              0           0           0
pkulkarni@pkulkarni:~$ uname -mrs
Linux 5.15.0-50-generic x86_64
```

Ubuntu on QEMU

CPU TESTING

```
root@ea28619d1028:/# sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any op
ons.
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: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:   463.65

General statistics:
    total time:                          10.0017s
    total number of events:              4638

Latency (ms):
         min:                                    2.15
         avg:                                    2.16
         max:                                    3.42
         95th percentile:                        2.18
         sum:                                 9999.52

Threads fairness:
    events (avg/stddev):           4638.0000/0.00
    execution time (avg/stddev):   9.9995/0.00
```

Docker Ubuntu Container

```
pkulkarni@pkulkarni:~$ sysbench --test=cpu --cpu-max-prime=20000 run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20 (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: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:   462.31

General statistics:
    total time:                          10.0003s
    total number of events:              4624

Latency (ms):
         min:                                    2.14
         avg:                                    2.16
         max:                                    2.79
         95th percentile:                        2.18
         sum:                                 9995.00

Threads fairness:
    events (avg/stddev):           4624.0000/0.00
    execution time (avg/stddev):   9.9950/0.00
```

Ubuntu on QEMU

`sysbench --test=cpu --cpu-max-prime=20000 run`
(default test scenario)

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	463.65	10.0017	2.15	2.16	3.42	4638	9.9995
1	463.51	10.0004	2.15	2.16	3.39	4636	9.9976
1	451.80	10.0005	2.14	2.21	3.56	4519	9.9954
1	463.19	10.0007	2.15	2.16	3.38	4633	9.9983
1	462.96	10.0015	2.15	2.16	3.08	4631	9.9989

Docker Ubuntu Container

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	442.79	10.0010	2.15	2.26	4.11	4429	9.9929
1	440.11	10.0004	2.15	2.27	4.82	4402	9.9904
1	451.11	10.0004	2.15	2.21	3.52	4512	9.9918
1	457.39	10.0007	2.15	2.18	3.59	4575	9.9947
1	462.48	10.0010	2.15	2.16	2.50	4626	9.9965

Ubuntu on QEMU

Both the environments show similar performance in this scenario.

```
sysbench --test=cpu --cpu-max-prime=50000 run
```

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	131.24	10.0030	7.55	7.62	10.90	1313	10.0013
1	127.44	10.0034	7.55	7.84	11.29	1275	10.0005
1	131.30	10.0063	7.55	7.61	8.34	1314	10.0044
1	131.70	10.0062	7.55	7.59	8.90	1318	10.0044
1	131.57	10.0007	7.56	7.60	8.86	1316	9.9988

Docker Ubuntu Container

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	129.53	10.0035	7.57	7.71	11.03	1296	9.9977
1	131.18	10.0071	7.56	7.62	9.12	1313	10.0030
1	114.90	10.0073	7.57	8.69	13.46	1150	9.9975
1	131.25	10.0022	7.56	7.61	9.74	1313	9.9984
1	116.46	10.0014	7.56	8.58	14.45	1165	9.9914

Ubuntu on QEMU

Both environments still show similar performance.

```
sysbench --test=cpu --cpu-max-prime=500000 --time=30 run
```

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	5.36	30.0179	185.11	186.44	190.14	161	30.0166
1	5.38	30.0999	184.60	185.79	194.50	162	30.0987
1	5.39	30.0393	184.50	185.42	186.32	162	30.0381
1	5.35	30.1073	184.15	186.99	202.78	161	30.1059
1	5.27	30.0024	184.83	189.88	215.37	158	30.0010

Docker Ubuntu Container

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	5.20	30.1727	185.08	192.16	244.74	157	30.1697
1	4.92	30.0654	185.04	203.12	283.89	148	30.0622
1	4.84	30.1507	187.16	206.49	329.04	146	30.1470
1	4.72	30.0956	186.09	211.92	254.13	142	30.0922
1	5.17	30.1857	185.40	193.48	319.95	156	30.1822

Ubuntu on QEMU

After increasing the time parameter, QEMU container shows a minute decrease in performance and increase in it’s latency.

sysbench --test=cpu --cpu-max-prime=500000 --time=30 --threads=8 run

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	26.59	30.2678	252.24	299.98	335.83	100.625/0.48	30.1858/0.08
8	22.17	30.2636	257.77	359.57	667.30	83.8750/1.62	30.1587/0.08
8	22.91	30.3731	279.64	347.08	480.64	87.0000/0.71	30.1963/0.11
8	23.83	30.2907	257.73	334.15	695.33	90.2500/1.20	30.1569/0.08
8	24.11	30.2308	268.85	330.44	490.74	91.125/0.78	30.1112/0.09

Docker Ubuntu Container

Threads	CPU speed (events per s)	Total time (s)	Latency			Threads fairness	
			min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	9.90	30.7105	747.37	807.48	883.73	38	30.6841/0.01
8	9.75	30.3556	745.36	819.66	1010.95	37	30.3275/0.02
8	10.46	30.6010	720.27	764.34	845.71	40	30.5736/0.01
8	9.48	30.3639	689.54	842.30	1033.04	36	30.3226/0.02
8	9.22	30.3769	750.80	866.66	976.79	35	30.3332/0.03

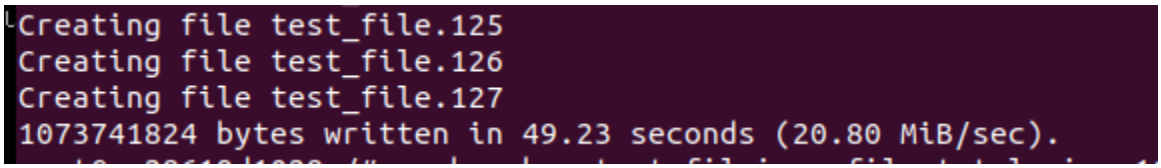
Ubuntu on QEMU

Upon enabling multiple threads, Docker Ubuntu shows significant improvement in performance.
It has access to all 8 processors, enabling it to distribute load around. However, since QEMU Ubuntu has access to only 2 processors. we see it had lesser performance and higher latency.

FILEIO TESTING ON DOCKER UBUNTU CONTAINER

file-size : 1 GB

sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw prepare



sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw run

```
root@ea28619d1028:/# sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any opti
ons.
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


Extra file open flags: (none)
128 files, 8MiB each
1GiB 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:                      27.12
    writes/s:                     18.08
    fsyncs/s:                     62.28

Throughput:
    read, MiB/s:                  0.42
    written, MiB/s:               0.28

General statistics:
    total time:                          11.0615s
    total number of events:              1061

Latency (ms):
         min:                                    0.00
         avg:                                    9.79
         max:                                 1353.88
         95th percentile:                       20.74
         sum:                                10386.73

Threads fairness:
    events (avg/stddev):           1061.0000/0.00
    execution time (avg/stddev):   10.3867/0.00
```

sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw cleanup

```
root@ea28619d1028:/# sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw cleanup
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any opti
ons.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Removing test files...
```

```
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


Extra file open flags: (none)
128 files, 16MiB each
2GiB 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:                      41.99
    writes/s:                     27.93
    fsyncs/s:                     102.14

Throughput:
    read, MiB/s:                  0.66
    written, MiB/s:               0.44

General statistics:
    total time:                          10.0234s
    total number of events:              1597

Latency (ms):
         min:                                    0.00
         avg:                                    6.26
         max:                                  887.48
         95th percentile:                       17.32
         sum:                                10002.15

Threads fairness:
    events (avg/stddev):           1597.0000/0.00
    execution time (avg/stddev):   10.0021/0.00
```

sysbench --test=fileio --file-total-size=1G --file-test-mode=rndrw run

128 files, 8 MB each
1GB total file size
Block size 16KB

R/W ratio for combined random IO Test: 1.5
Periodic fysnc enabled per 100 requests
Using synchronous I/O mode
Doing random r/w test

	Throughput			Latency			Threads fairness	
Threads	read MiB/s	written MiB/s	Total time (s)	min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	0.81	0.54	10.4704	0.00	4.89	537.99	2044.00/0.00	9.9962
1	0.57	0.38	13.1945	0.00	5.74	531.25	1757.00/0.00	10.0914
1	0.53	0.35	10.6042	0.00	7.61	998.98	1316.00/0.00	10.0141
1	0.65	0.43	11.6167	0.00	5.64	787.91	1779.00/0.00	10.0374
1	0.54	0.36	10.3600	0.00	7.36	1208.61	1360.00/0.00	10.0039

Docker Ubuntu Container

	Throughput			Latency			Threads fairness	
Threads	read MiB/s	written MiB/s	Total time (s)	min	avg	max	events (avg/stddev)	execution time (avg/stddev)
1	0.46	0.31	10.2389	0.00	9.07	777.21	1102	10.0001
1	0.68	0.46	10.9763	0.00	5.68	1246.07	1771	10.0653
1	0.53	0.36	10.5329	0.00	7.48	867.94	1348	10.0796
1	0.85	0.56	11.0799	0.00	4.82	612.10	2165	10.4372
1	0.36	0.24	10.3567	0.00	11.74	1875.56	858	10.0697

Ubuntu on QEMU

Similar performance is observed in both the environments.

```
sysbench --num-threads=8 --test=fileio --file-total-size=1G
--file-test-mode=rndrw run
```

Threads	Throughput		Total time (s)	Latency			Threads fairness	
	read MiB/s	written MiB/s		min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	1.82	1.21	13.4035	0.00	13.72	999.84	732.25/141.6	10.04/0.07
8	1.31	0.87	11.4527	0.00	22.83	1644.96	441.625/83.4	10.0839/0.00
8	0.84	0.55	11.2413	0.00	37.24	1981.02	269.25/27.67	10.0259/0.02
8	0.99	0.66	12.2970	0.00	28.32	1163.42	354.75/24.80	10.0462/0.02
8	1.15	0.76	12.0950	0.00	24.52	1015.93	409.25/24.12	10.0344/0.02

Docker Ubuntu Container

Threads	Throughput		Total time (s)	Latency			Threads fairness	
	read MiB/s	written MiB/s		min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	1.30	0.87	10.0864	0.00	25.16	1507.82	397.87/75.78	10.0114/0.01
8	0.87	0.58	12.897	0.00	30.61	1145.77	327.75/69.99	10.0317/0.00
8	1.45	0.96	10.3785	0.00	22.42	693.74	446.12/77.08	10.0027/0.00
8	0.75	0.50	11.3093	0.00	40.06	997.44	254.75/59.95	10.3059/0.10
8	1.33	0.89	10.5879	0.00	24.41	958.52	415.12/82.75	10.1335/0.00

Ubuntu on QEMU

Similar performance is observed in both the environments.

```
sysbench --num-threads=8 –file-num=8 --test=fileio --file-total-size=1G
--file-test-mode=rndrw prepare
sysbench --num-threads=8 –file-num=8 --test=fileio --file-total-size=1G
--file-test-mode=rndrw run
```

8 files, 128 MB each
1GB total file size
Block size 16KB

Threads	Throughput		Total time (s)	Latency			Threads fairness	
	read MiB/s	written MiB/s		min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	2.11	1.4	10.948	0.00	30.44	1003.79	331.5/26.09	10.0913/0.03
8	2.03	1.35	10.1851s	0.00	34.03	1297.07	296.7500/50.35	10.0988/0.05
8	2.14	1.42	10.1121s	0.00	32.44	1197.90	310.5000/42.65	10.0729/0.00
8	1.52	1.01	11.1252s	0.00	41.60	1501.79	242.2500/46.82	10.0788/0.06
8	0.89	0.59	10.5406s	0.00	76.52	1885.85	135.0000/37.91	10.3304/0.02

Docker Ubuntu Container

Threads	Throughput		Total time (s)	Latency			Threads fairness	
	read MiB/s	written MiB/s		min	avg	max	events (avg/stddev)	execution time (avg/stddev)
8	7.02	4.68	10.1451	0.00	9.88	542.33	1026/230	10.13/0/00
8	2.39	1.60	10.1919	0.00	29.01	1006.77	351.0/137.58	10.1832/0.00
8	1.92	1.28	10.2294	0.00	36.06	1527.88	283.5/120.46	10.2235/0.00
8	1.62	1.08	10.3962	0.00	42.75	1271.51	243.0/176.77	10.3891/0.00
8	1.61	1.08	11.0433	0.00	41.23	1223.95	256.2/117.71	10.5657/0.00

Ubuntu on QEMU

Similar performance is observed in both the environments.

During the experiment, it was observed that OS Level Virtualization (Docker Ubuntu) affects the performance of the host (my Linux machine), since hardware is shared between them.
However, the QEMU Ubuntu machine runs independently, and doesn't affect other currently running processes on the system, as both hardware and software are virtualized in this case.
Docker Ubuntu appeared to be faster in some scenarios (here: cpu testing)

***