

System vs OS Virtualization

Git Repository

<https://github.com/evanFengg/CloudComputing.git>

It is a public repository.










Configurations

I used VMware on my Windows 10 PC to create a Linux environment which is ubuntu 22.04

Linux Configurations(guest OS)

OS: Ubuntu 22.04 64-bit

Virtual Machine Settings

Hardware Options	
Device	Summary
 Memory	8 GB
 Processors	2
 Hard Disk (SCSI)	35 GB
 CD/DVD (SATA)	Auto detect
 Network Adapter	NAT
 USB Controller	Present
 Sound Card	Auto detect
 Printer	Present
 Display	Auto detect

QEMU VM Configurations

Guest OS: Ubuntu 20.04

Processor cores: 2

Memory: 512 MB, 1GB, 2GB

Storage: 12 GB

Docker Container Configurations

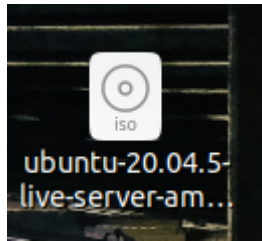
Image: My own image createc from Ubuntu:22.04(latest) + sysbench

Processor cores: 2

Memory: 512 MB, 1GB, 2GB

QEMU Setup

1. Download Ubuntu 20.04 server ISO image.



2. Open the terminal and install QEMU.

```
$ sudo apt-get install qemu  
  
$ sudo apt-get install qemu-utils  
  
$ sudo apt-get install qemu-system-x86
```

3. Create a QEMU image (12GB disk space, image format of QEMU copy-on-write v2).

```
$ sudo qemu-img create ubuntu.img 12G -f qcow2
```

4. Install QEMU VM.

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -cdrom ./ubuntu-20.04.5-live-server-amd64.iso -m 2048 -boot strict=on
```

```
evan@ubuntu:~/Desktop$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 512  
mp cores=2 -boot strict=on  
[sudo] password for evan:  
[ ]
```

5. Boot the VM created from the image ubuntu.img.

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 512 -smp cores=2 -boot  
strict=on
```

6. Login to the QEMU Ubuntu with username and password. Then we can see the QEMU VM running.

```

Machine View
MMbnn2qj1IHL7ghGgJGy3YgsCX20k6iLVURcbdCIPJt53npu5osAN7Nj4iXi1jK+2vdSuqG1Xv26uk10Q8MLuhr0gMPyKronsnQ6irIRTJY/S3Z1/38xrWd8w5IDX0Y5
i3+Rc4BIOCv8Cjc2Lop6T/z5/FGy37k2fMFmWa/pgTdJviLHEHgJSHR67X0C/Ku/PoHLx6CSns83Kcw3Yc7d09jrFWm4uR8TrAeacB3XRPD8105P8RXzBNV3ERZLJqS1
cRKXTWDZqLpo4HR1VnyrfDDHMx/Ip4b80g+pH+CTHLiwJWo/6tL817ezQ0T4L5LA2ceR68B/x1QewoaSVa9x1h1goz/z0mAFwU3a2mu0Q03J3QGarYa9tdbonG2vWp35
dc1w4lnkkDKKEzgVtpDLWuzVz9S5+bKGB5gz8k/c= root@evan
-----END SSH HOST KEY KEYS-----
[ 177.350996] cloud-init[1474]: Cloud-init v. 22.2-0ubuntu1~20.04.3 running 'modules:final' at Fri, 27 Jan 2023 00:57:32 +0000.
Up 174.49 seconds.
[ 177.408032] cloud-init[1474]: Cloud-init v. 22.2-0ubuntu1~20.04.3 finished at Fri, 27 Jan 2023 00:57:35 +0000. Datasource Dat
aSourceNone. Up 177.21 seconds
[ 177.482578] cloud-init[1474]: 2023-01-27 00:57:35.564 - cc_final_message.py[WARNING]: Used fallback datasource
[ OK ] Finished Execute cloud user/final scripts.
[ OK ] Reached target Cloud-init target.
evan
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 Fri 27 Jan 2023 01:01:56 AM UTC

System load:          0.14
Usage of /:            49.7% of 8.02GB
Memory usage:         39%
Swap usage:           0%
Processes:            102
Users logged in:      0
IPv4 address for ens3: 10.0.2.15
IPv6 address for ens3: fec0::5054:ff:fe12:3456

68 updates can be applied immediately.
42 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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.

evan@evan:~$

```

7. Install the sysbench in the QEMU.

```
$ sudo apt update
```

```
$ sudo apt install sysbench
```

Sysbench version: 1.0.18




8. Set up the GitHub authorization with ssh and pull the repository to the local VM.

SSH keys

[New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

 SSH	Ubuntu_VM SHA256:w+esE7TehisltxYuRRE7... Added on Jan 27, 2023 Last used within the last week — Read/write	Delete
 SSH	qemu SHA256:oKYU52J1/ABTHJ5xMcN... Added on Jan 30, 2023 Last used within the last week — Read/write	Delete
 SSH	mac SHA256:PrTNAtVE26Ha716fp... Added on Jan 31, 2023 Last used within the last week — Read/write	Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

QEMU VM Performance Testing with Varying Configurations

the scenarios is that I will change QEMU VM memory from 512 to 2048 MB ,and I will analyze their performance by getting the sysbench measurement.

Steps

1. Boot the VM, each time with different parameter.

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 512 -smp cores=2 -boot strict=on
```

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 1024 -smp cores=2 -boot strict=on
```

```
$ sudo qemu-system-x86_64 -hda ubuntu.img -boot d -m 2048 -smp cores=2 -boot strict=on
```

2. Show VM environment.

```
$ printenv
```

3. Use the 'top' command line tool in the host (Linux, Ubuntu desktop) terminal to get the CPU utilization, including user-level and kernel-level.

```
$ top
```

4. Run each shell script and save results.

```
$ bash script_file_name.sh > output_file_name.txt
```

5. Push all output files to the repository.

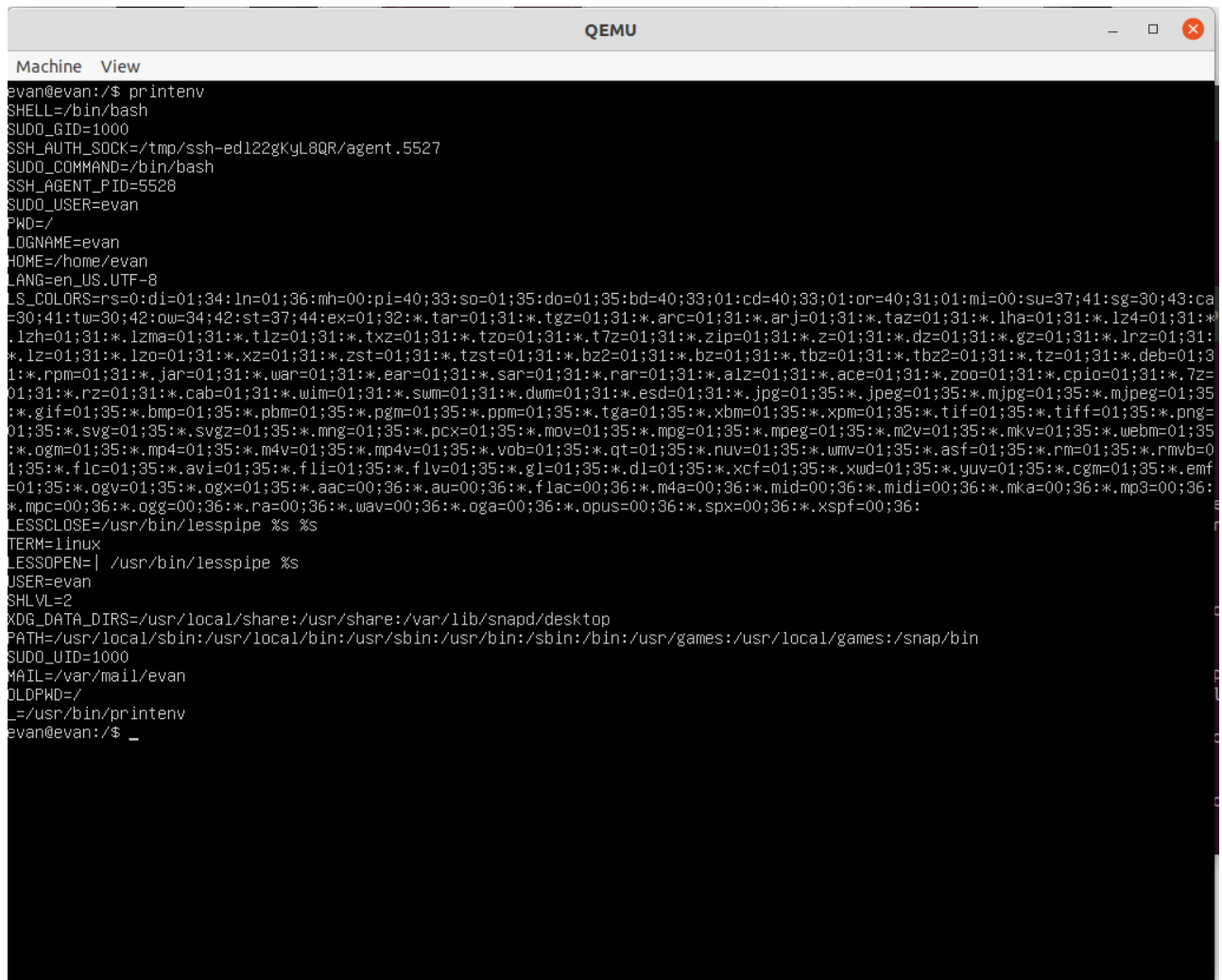
```
$ git add .

$ git commit -m "add files"

$ git push origin main
```

Running Environments

- Low-memory test



```

Machine View
evan@evan:/$ printenv
SHELL=/bin/bash
SUDO_GID=1000
SSH_AUTH_SOCK=/tmp/ssh-ed122gKyL8QR/agent.5527
SUDO_COMMAND=/bin/bash
SSH_AGENT_PID=5528
SUDO_USER=evan
PWD=/
LOGNAME=evan
HOME=/home/evan
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33:01:cd=40;33:01:or=40;31:01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lzh=01;31:*.lzh=01;31:*.lzh=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
LESSCLOSE=/usr/bin/lesspipe %s %s
TERM=linux
LESSOPEN=| /usr/bin/lesspipe %s
USER=evan
SHLV=2
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
SUDO_UID=1000
MAIL=/var/mail/evan
OLDPWD=/
_=/usr/bin/printenv
evan@evan:/$ _

```

- Middle-memory test

```

Machine View
evan@evan:~$ printenv
SHELL=/bin/bash
XDG_SEAT=seat0
PWD=/home/evan
LOGNAME=evan
XDG_SESSION_TYPE=ttty
MOTD_SHOWN=pam
HOME=/home/evan
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33:01:cd=40;33:01:or=40;31:01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz0=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzt=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.sum=01;31:*.dum=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
INVOCATION_ID=518466faecb3463f88884fea66a049d0
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=linux
LESSOPEN=| /usr/bin/lesspipe %s
USER=evan
SHLVL=1
XDG_VTNR=1
XDG_SESSION_ID=2
XDG_RUNTIME_DIR=/run/user/1000
JOURNAL_STREAM=9:24980
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/napd/desktop
HUSHLOGIN=FALSE
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
MAIL=/var/mail/evan
_=/usr/bin/printenv
evan@evan:~$

```

- High-memory test

```

Machine View
evan@evan:~$ printenv
SHELL=/bin/bash
XDG_SEAT=seat0
PWD=/home/evan
LOGNAME=evan
XDG_SESSION_TYPE=ttty
MOTD_SHOWN=pam
HOME=/home/evan
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33:01:cd=40;33:01:or=40;31:01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz0=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzt=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.sum=01;31:*.dum=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
INVOCATION_ID=d6a9095fee534920a2833eb802fc44cd
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=linux
LESSOPEN=| /usr/bin/lesspipe %s
USER=evan
SHLVL=1
XDG_VTNR=1
XDG_SESSION_ID=2
XDG_RUNTIME_DIR=/run/user/1000
JOURNAL_STREAM=9:26709
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/napd/desktop
HUSHLOGIN=FALSE
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
MAIL=/var/mail/evan
_=/usr/bin/printenv
evan@evan:~$ _

```

- VM Sysbench CPU Test Script

```
#!/bin/bash

mem=$(free -m | grep "Mem:" | awk '{print $2}')

if [ $mem -le 512 ];
then
    echo "Low-memory QEMU VM Sysbench CPU Tests: 2 core, 512 MB Memory"
    echo "=====

elif [ $mem -le 1024 ];
then
    echo "Middle-memory QEMU VM Sysbench CPU Tests: 2 core, 1024 MB
Memory"
    echo
    "=====

elif [ $mem -le 2048 ];
then
    echo "High-memory QEMU VM Sysbench CPU Tests: 2 core, 2048 MB Memory"
    echo "=====

else
    echo "Incorrect Memory to run sysbench, needs to match preset
configurations"
    exit 1
fi

# Run sysbench CPU test for 5 times on each VM configuration
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    echo "-----"
    sysbench --test=cpu --cpu-max-prime=10000 --time=30 run
done
```

- VM Sysbench File IO Test Script

```
#!/bin/bash

mem=$(free -m | grep "Mem:" | awk '{print $2}')

if [ $mem -le 512 ];
then
    echo "Low-memory QEMU VM Sysbench File IO Tests: 2 core, 512 MB
Memory"
    echo
```

```

"=====

elif [ $mem -le 1024 ];
then
    echo "Middle-memory QEMU VM Sysbench File IO Tests: 2 core, 1024 MB
Memory"
    echo
"=====

elif [ $mem -le 2048 ];
then
    echo "High-memory QEMU VM Sysbench File IO Tests: 2 core, 2048 MB
Memory"
    echo
"=====

else
    echo "Incorrect Memory to run sysbench, needs to match preset
configurations"
    exit 1
fi

# Run sysbench file io tests in sequential write mode and sequential read
mode
# Each test is run 5 times on each VM configuration
echo "File IO Test in Sequential Write Mode"
echo "-----"
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    # Create the files to test
    sysbench --test=fileio --file-num=64 --file-total-size=4G prepare
    # Run the actual test
    sysbench --test=fileio --file-num=64 --file-total-size=4G --file-test-
mode=seqwr --max-time=60 run
    # Cleanup the test files
    sysbench --test=fileio cleanup
    # Drop the cache
    sudo sh -c 'echo 3 > /proc/sys/vm/drop_caches'
done

echo "File IO Test in Sequential Read Mode"
echo "-----"
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    # Create the files to test
    sysbench --test=fileio --file-num=64 --file-total-size=4G prepare
    # Run the actual test
    sysbench --test=fileio --file-num=64 --file-total-size=4G --file-test-
mode=seqrd --max-time=60 run
    # Cleanup the test files
    sysbench --test=fileio cleanup
    # Drop the cache

```



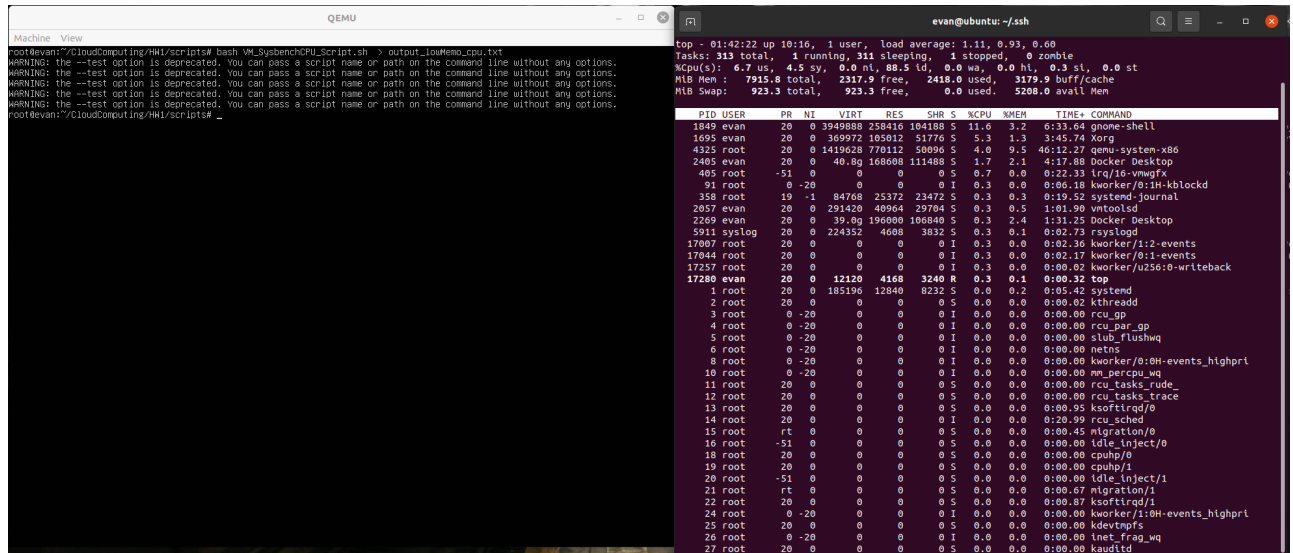
```
sudo sh -c 'echo 3 > /proc/sys/vm/drop_caches'
```

done

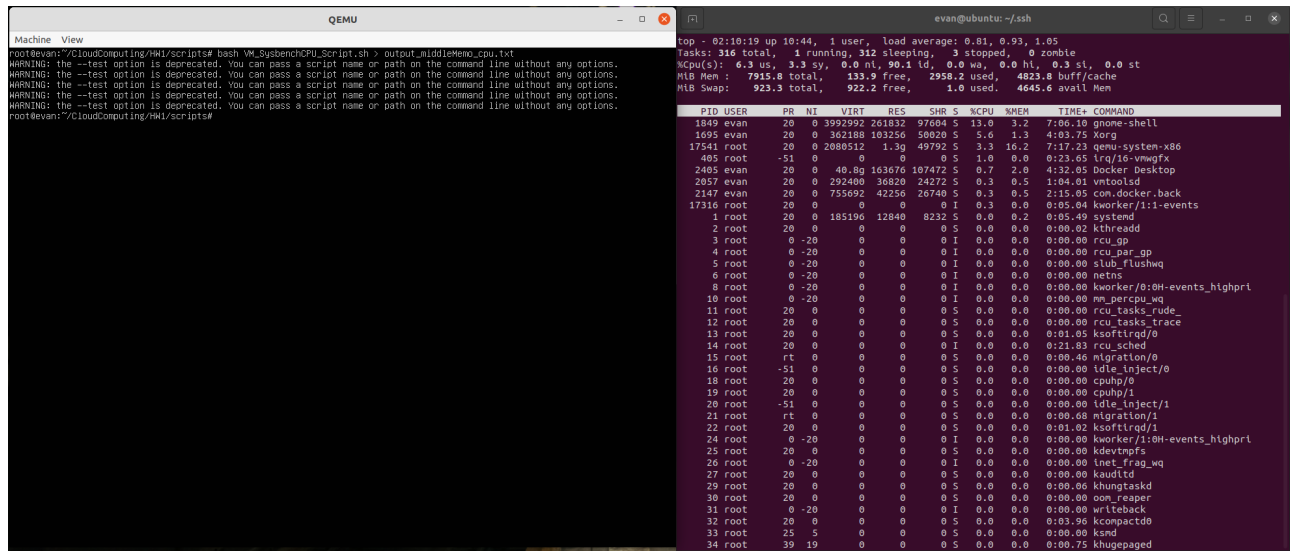
QEMU VM Performance Testing Results

CPU Tests

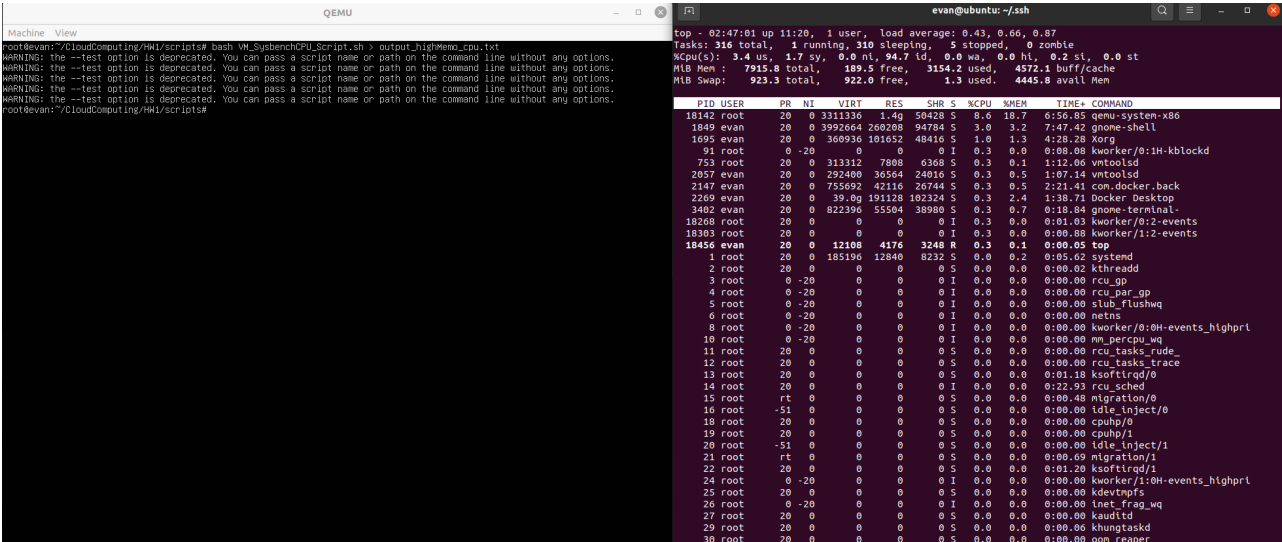
1. VM CPU Test with 512MB Memory



2. VM CPU Test with 1GB Memory



3. VM CPU Test with 2GB Memory



Low Memory (512 MB)

512 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	459.19	13348	22.10	0.83
min	434.37	13028	21.20	0.76
max	487.37	14625	22.10	1.1
std	5.83	189.80	0.23	0.23

Middle Memory (1GB)

1024 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	476.40	14876	22.12	0.90
min	475.46	14267	22.00	0.80
max	481.77	14457	22.30	1.20
std	6.25	176	0.49	0.62

High Memory (2GB)

2048 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	456.73	14406	25.16	1.16
min	478.00	14343	24.60	0.80
max	485.01	14556	25.90	1.70
std	4.92	132	0.49	0.34

From the data we can know that as the memory increases, the CPU performance improves slightly, but the difference is not large.

File I/O Tests

Low Memory (512 MB)

512 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	1046.86	0.57	0.98	5957.33	0.16	0.91
min	965.81	0.51	0.97	5502.24	0.15	0.91
max	1161.96	0.62	0.98	6198.67	0.17	0.91
std	76.26	0.04	0.00	293.42	0.01	0.00

Middle Memory (1GB)

1024 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	1144.23	0.52	0.97	5725.62	0.16	0.91
min	1134.34	0.51	0.97	4349.58	0.14	0.90
max	1155.40	0.52	0.98	6311.87	0.21	0.91
std	9.60	0.00	0.00	783.12	0.03	0.00

High Memory (2GB)

2048 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	1157.93	0.51	0.97	5585.14	0.17	0.91
min	1136.67	0.50	0.97	4338.04	0.15	0.89
max	1182.00	0.52	0.97	6146.01	0.21	0.93
std	18.79	0.01	0.00	779.56	0.03	0.01

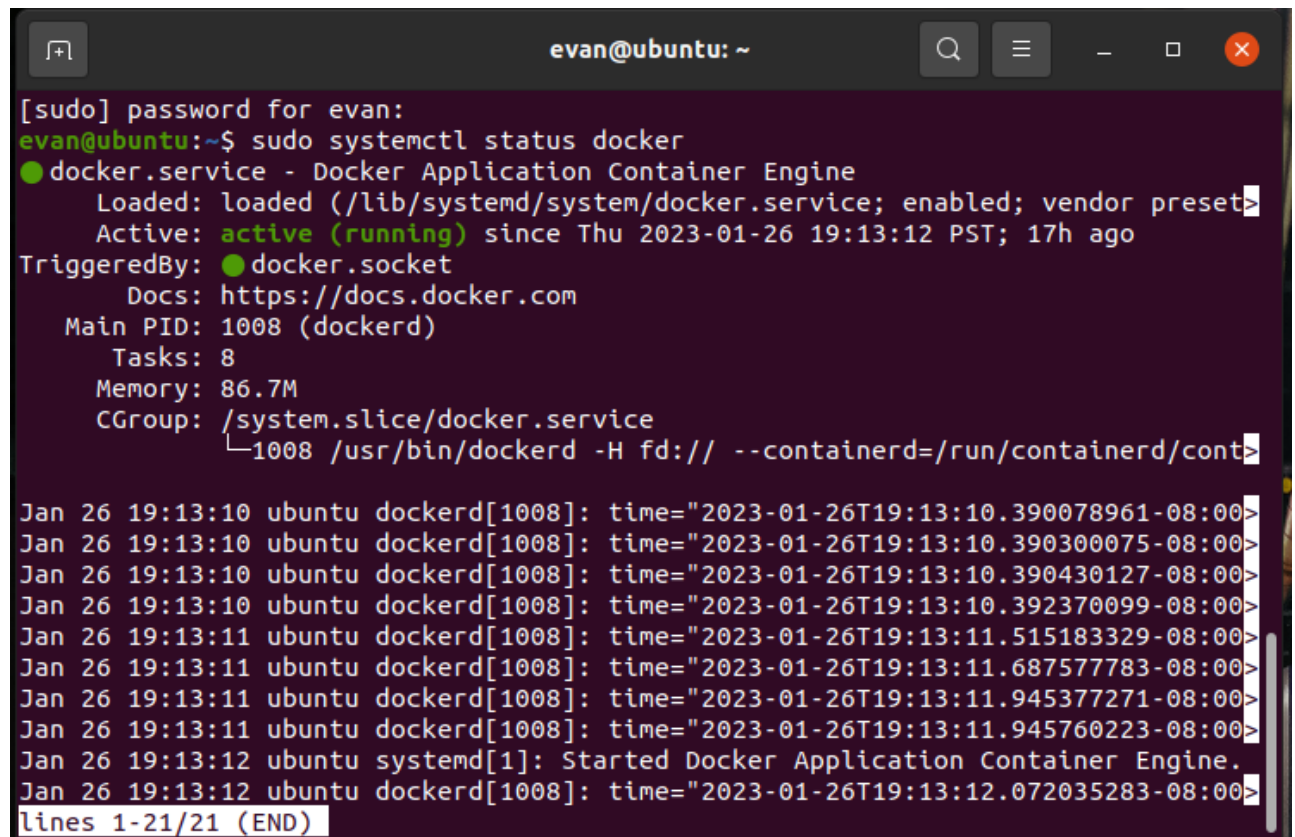
From above data we can know that, reading speed is significantly faster than writing, and the disk utility of reading is also slightly lower than that of writing. As memory increases, writes are slightly faster and reads are slightly slower.

Docker Container Setup

1. Install Docker, including Docker Engine, containerd, and Docker Compose. I used steps from "Set up the repository" on <https://docs.docker.com/engine/install/ubuntu/>.
2. Verify Docker is installed and running correctly.

```
$ sudo systemctl start docker
```

```
$ sudo systemctl status docker
```

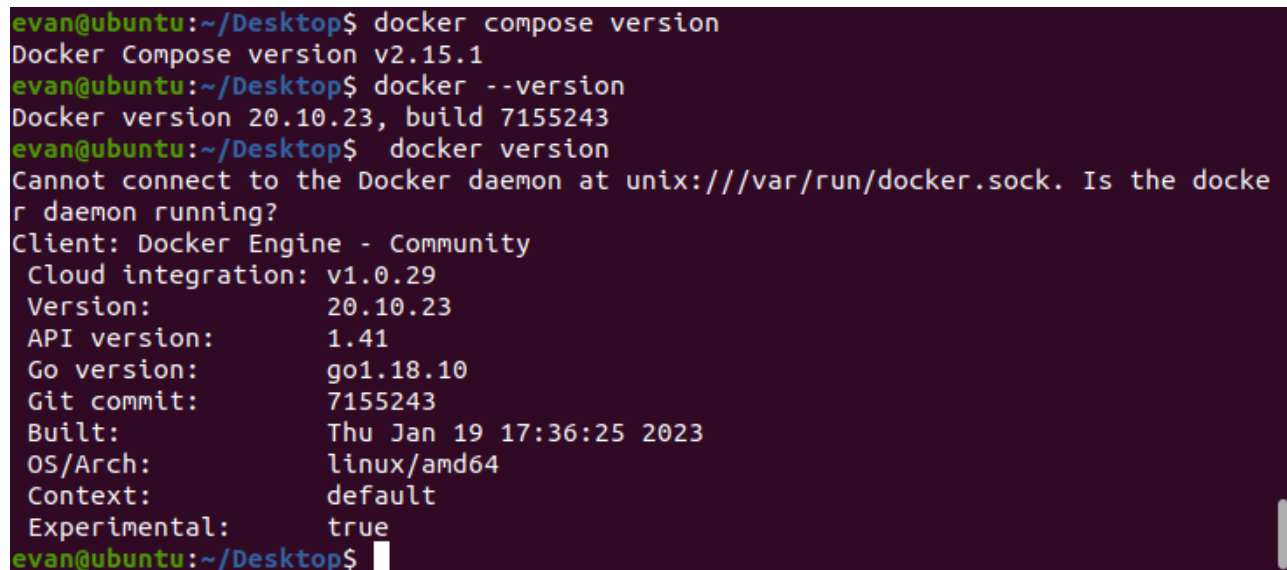


```

[evan@ubuntu: ~]$ sudo systemctl status docker
[sudo] password for evan:
evan@ubuntu:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset:
   Active: active (running) since Thu 2023-01-26 19:13:12 PST; 17h ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 1008 (dockerd)
       Tasks: 8
      Memory: 86.7M
     CGroup: /system.slice/docker.service
             └─1008 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/cont
Jan 26 19:13:10 ubuntu dockerd[1008]: time="2023-01-26T19:13:10.390078961-08:00"
Jan 26 19:13:10 ubuntu dockerd[1008]: time="2023-01-26T19:13:10.390300075-08:00"
Jan 26 19:13:10 ubuntu dockerd[1008]: time="2023-01-26T19:13:10.390430127-08:00"
Jan 26 19:13:10 ubuntu dockerd[1008]: time="2023-01-26T19:13:10.392370099-08:00"
Jan 26 19:13:11 ubuntu dockerd[1008]: time="2023-01-26T19:13:11.515183329-08:00"
Jan 26 19:13:11 ubuntu dockerd[1008]: time="2023-01-26T19:13:11.687577783-08:00"
Jan 26 19:13:11 ubuntu dockerd[1008]: time="2023-01-26T19:13:11.945377271-08:00"
Jan 26 19:13:11 ubuntu dockerd[1008]: time="2023-01-26T19:13:11.945760223-08:00"
Jan 26 19:13:12 ubuntu systemd[1]: Started Docker Application Container Engine.
Jan 26 19:13:12 ubuntu dockerd[1008]: time="2023-01-26T19:13:12.072035283-08:00"
lines 1-21/21 (END)

```

And also check the Docker version.



```

evan@ubuntu:~/Desktop$ docker compose version
Docker Compose version v2.15.1
evan@ubuntu:~/Desktop$ docker --version
Docker version 20.10.23, build 7155243
evan@ubuntu:~/Desktop$ docker version
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docke
r daemon running?
Client: Docker Engine - Community
 Cloud integration: v1.0.29
  Version:          20.10.23
 API version:       1.41
 Go version:        go1.18.10
 Git commit:        7155243
 Built:             Thu Jan 19 17:36:25 2023
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
evan@ubuntu:~/Desktop$

```

3. Create a Unix group called `docker` and add users to it. Then restart Linux.

```
$ sudo groupadd docker
```

```
$ sudo usermod -aG docker $USER
```

```
$ newgrp docker
```

4. Pull ubuntu image. (https://hub.docker.com/_/ubuntu)
5. Create a container from ubuntu image and run it.

```
$ docker run -it ubuntu bin/bash
```

6. Install sysbench in container.

```
apt update
```

```
apt install sysbench
```

7. Create my own image.

```
$ docker commit -m "sysbenchcontainer" 350973ea83e6 ubuntu_sysbench:v1.0
```

8. Check image.

```
$ docker image ls
```

```
evan@ubuntu:~/Desktop$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS
TS            NAMES
3d143f0988f4   ubuntu   "bin/bash"              About an hour ago Up About an hour
epic_pare
evan@ubuntu:~/Desktop$ docker commit 3d143f0988f4 ubuntu_sysbench:v1.0
sha256:55cad855ae6274b11eb9b090d391519a7c543c50f0c20cddfe56691825c4e317
evan@ubuntu:~/Desktop$ docker image ls
REPOSITORY      TAG         IMAGE ID      CREATED        SIZE
ubuntu_sysbench v1.0        55cad855ae62  20 seconds ago 129MB
ubuntu          latest      6b7dfa7e8fdb  7 weeks ago   77.8MB
hello-world     latest      feb5d9fea6a5  16 months ago 13.3kB
evan@ubuntu:~/Desktop$
```

9. Operations for Docker container management:

```
$ docker container ls - List containers
```

```
$ docker image ls - List images
```

```
$ docker run, $ docker container run - Run a command in a new container
```

```
exit or ctrl-D - Exit out of the docker container bash shell
```

```
$ docker container create - Create a new container
```

```
$ docker container update - Update configuration of one or more containers
```

```
$ docker container commit - Create a new image from a container's changes
```

```
$ docker container cp - Copy files/folders between a container and the local filesystem
```

```
$ docker container exec - Run a command in a running container
```

10. Install git on the Linux host(VM on my windows PC), set up the GitHub authorization with ssh and pull the repository to the local Linux host.

Docker Container Performance Testing with Varying Configurations

the scenarios is that I will change docker container memory from 512 to 2048 MB ,and I will analyze their performance by getting the sysbench measurement.

Steps

1. Update the contain_config in scripts and pull updated repository to the local host.

```
$ git pull origin main
```

2. Run Docker container, each time with different configurations. I used `--privileged` to drop cache.

```
$ docker run --rm -it --privileged --cpus="2" -m 512m ubuntu_sysbench:v1.0  
bin/bash
```

```
$ docker run --rm -it --privileged --cpus="2" -m 1024m ubuntu_sysbench:v1.0  
bin/bash
```

```
$ docker run --rm -it --privileged --cpus="2" -m 2048m ubuntu_sysbench:v1.0  
bin/bash
```

3. Run the `env` command in container to display container environment.
4. Open another terminal, and copy the script files to be executed to the container.

```
$ docker cp ./SCU_COEN241/HW1/Scripts/script_file_name.sh  
container_id:/script_file_name.sh
```

5. For CPU test, open another terminal in the host (Linux, Ubuntu desktop) and use the 'top' command line tool to get the CPU utilization, including user-level and kernel-level.

```
$ top
```

6. Run each shell script and save results.

```
bash /script_file_name.sh > /output_file_name.txt
```

7. Copy the output files from the container. Then exit container.

```
$ docker cp container_id:/output_file_name.txt  
./SCU_COEN241/HW1/Container_Results/output_file_name.txt  
  
exit
```

8. Push all output files to the repository.

```
$ cd SCU_COEN241  
  
$ git add .  
  
$ git commit -m "Add result"  
  
$ git push origin main
```

Running Environments

- Low-memory test

```
root@04b61319a775: /
evan@ubuntu:~/Desktop/CloudComputing/HW1/scripts$ docker run --rm -it --privileged --cpus="2"
-m 512m ubuntu_sysbench:v1.0 bin/bash
root@04b61319a775:/# env
HOSTNAME=04b61319a775
PWD=/
HOME=/root
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or
=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.
tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01
;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01
;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz
=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.
ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;3
1:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=
01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35
:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz
=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:
*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt
=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.
fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:
*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00
;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.og
a=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
TERM=xterm
SHLV=1
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
_=/usr/bin/env
root@04b61319a775:/#
```

- Middle-memory test


```

evan@ubuntu: ~/Desktop/CloudComputing/HW1/docker
-rw-rw-r-- 1 evan evan 2032 Jan 31 03:18 Container_IO_Script.sh
-rw-r--r-- 1 evan evan 4265 Jan 31 03:47 docker_cpu_output.txt
-rw-r--r-- 1 evan evan 33221 Jan 31 04:02 docker_io_output.txt
evan@ubuntu:~/Desktop/CloudComputing/HW1/docker$ env
SHELL=/bin/bash
SESSION_MANAGER=local/ubuntu:@/tmp/.ICE-unix/1833,unix/ubuntu:/tmp/.ICE-unix/1833
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
SSH_AGENT_PID=1798
GTK_MODULES=gail:atk-bridge
PWD=/home/evan/Desktop/CloudComputing/HW1/docker
LOGNAME=evan
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=x11
PGP_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1
XAUTHORITY=/run/user/1000/gdm/Xauthority
WINDOWPATH=2
HOME=/home/evan
USERNAME=evan
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00;
su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;
31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01
;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.
tzt=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.
war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.
rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.
mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35
:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;
35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=
01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=0
1;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;
35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00
;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6003
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/0f82fd9d_5a15_405a_9a42_293aef79946c
INVOCATION_ID=94e04750e57547b081fa9264f6eb69f6
MANAGERPID=1620
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=evan
GNOME_TERMINAL_SERVICE=:1.111
DISPLAY=:0

```

- High-memory test


```

root@a2478848f391: /
evan@ubuntu:~$ docker run --rm -it --privileged --cpus="2" -m 2048m ubuntu_sysbench:v1.0 bin/bas
h
root@a2478848f391:/# env
HOSTNAME=a2478848f391
PWD=/
HOME=/root
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;
31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;
31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01
;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:
*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.
tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.
rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.w
im=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35
:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35
:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;
35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=
01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=0
1;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01
;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;
36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=0
0;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
TERM=xterm
SHLVL=1
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
_=/usr/bin/env
root@a2478848f391:/#

```

Shell Scripts

- Container Sysbench CPU Test Script

```

#!/bin/bash

#Arg = 0 - low-memory config, 512 MB
#Arg = 1 - middle-memory config, 1 GB
#Arg = 2 - high-memory config, 2GB

contain_config=$(( 0 ))

if [ $contain_config -eq 0 ];
then
    echo "Low-memory Docker Container Sysbench CPU Tests: 2 core, 512 MB
Memory"
    echo "=====

```

```

elif [ $contain_config -eq 1 ];
then
    echo "Middle-memory Docker Container Sysbench CPU Tests: 2 core, 1024
    MB Memory"
    echo
    "=====

elif [ $contain_config -eq 2 ];
then
    echo "High-memory Docker Container Sysbench CPU Tests: 2 core, 2048 MB
    Memory"
    echo "=====

else
    echo "Incorrect Memory to run sysbench, needs to match preset
    configurations"
    exit 1
fi

# Run sysbench CPU test for 5 times on each VM configuration
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    echo "-----"
    sysbench --test=cpu --cpu-max-prime=10000 --time=30 run
done

```

- Docker Sysbench File IO Test Script

```

#!/bin/bash

#Arg = 0 - low-memory config, 512 MB
#Arg = 1 - middle-memory config, 1 GB
#Arg = 2 - high-memory config, 2GB

contain_config=$(( 0 ))

if [ $contain_config -eq 0 ];
then
    echo "Low-memory Docker Container Sysbench CPU Tests: 2 core, 512 MB
    Memory"
    echo "=====

elif [ $contain_config -eq 1 ];
then
    echo "Middle-memory Docker Container Sysbench CPU Tests: 2 core, 1024
    MB Memory"
    echo
    "=====

```

```

elif [ $contain_config -eq 2 ];
then
    echo "High-memory Docker Container Sysbench CPU Tests: 2 core, 2048 MB
Memory"
    echo "===== "

else
    echo "Incorrect Memory to run sysbench, needs to match preset
configurations"
    exit 1
fi

# Run sysbench file io tests in sequential write mode and sequential read
mode
# Each test is run 5 times on each VM configuration
echo "File IO Test in Sequential Write Mode"
echo "-----"
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    # Create the files to test
    sysbench --test=fileio --file-num=64 --file-total-size=4G prepare
    # Run the actual test
    sysbench --test=fileio --file-num=64 --file-total-size=4G --file-test-
mode=seqwr --max-time=60 run
    # Cleanup the test files
    sysbench --test=fileio cleanup
    # Drop the cache
    sh -c 'echo 3 > /proc/sys/vm/drop_caches'
done

echo "File IO Test in Sequential Read Mode"
echo "-----"
for ((counter = 1; counter < 6; counter++))
do
    echo "Test $counter"
    # Create the files to test
    sysbench --test=fileio --file-num=64 --file-total-size=4G prepare
    # Run the actual test
    sysbench --test=fileio --file-num=64 --file-total-size=4G --file-test-
mode=seqrd --max-time=60 run
    # Cleanup the test files
    sysbench --test=fileio cleanup
    # Drop the cache
    sh -c 'echo 3 > /proc/sys/vm/drop_caches'
done

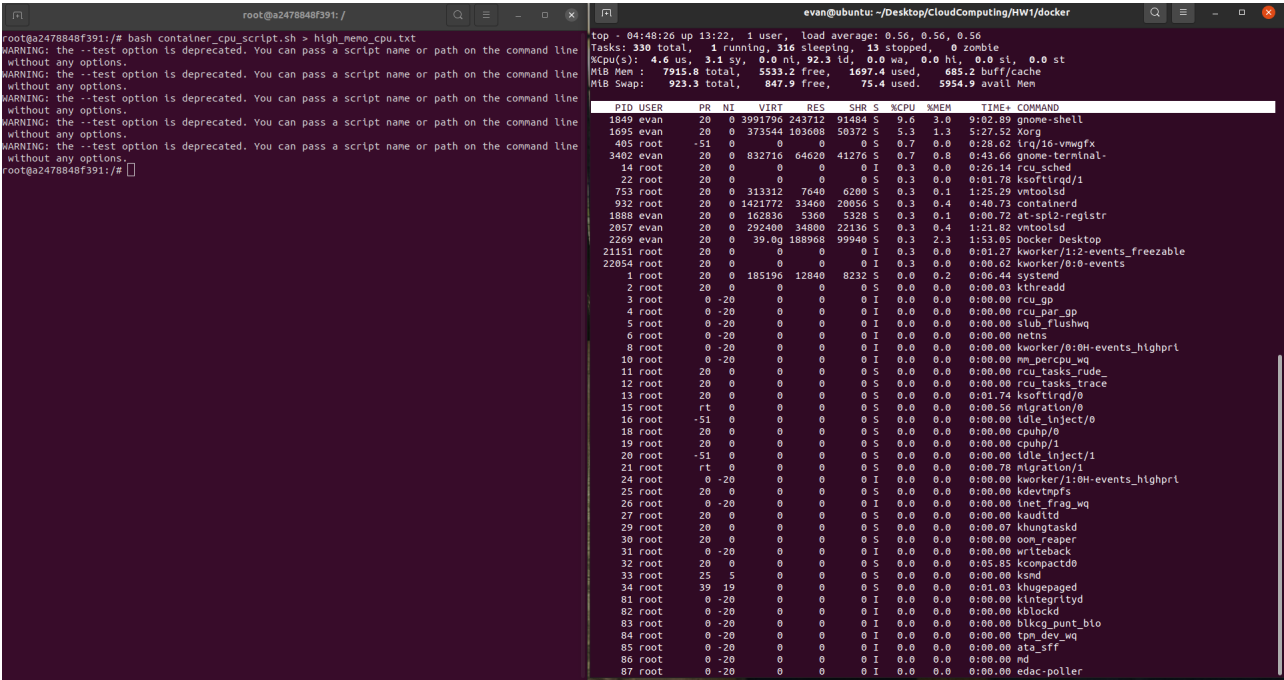
```

Docker Container Performance Testing Results

CPU Tests

1. Container CPU Test with 512MB Memory





Low Memory (512 MB)

512 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	3162.26	97867.20	25.44	0.12
min	3263.62	97915.00	25.10	0.10
max	3305.28	99164.00	25.90	0.20
std	36.82	3104.79	0.30	0.04

Middle Memory (1GB)

1024 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	3278.23	31209.40	25.64	0.22
min	3220.13	96610.00	25.10	0.10
max	3300.57	99023.00	27.80	0.70
std	38.74	2842.29	1.21	0.27

High Memory (2GB)

2048 MB mem	Events per second	Total events	Kern. CPU (%)	User CPU (%)
average	3287.03	31414.60	25.34	0.14
min	3282.01	98466.00	25.10	0.10
max	3297.62	98935.00	26.20	0.20
std	40.01	2899.17	0.48	0.05

As shown in the above tables, when running a Docker container, the CPU performance is basically the same with different memory size. Perhaps because of the good performance of Docker, experiments with smaller memory are required to show the impact on CPU performance.

File I/O Tests

Similar to the VM testing, the file I/O tests, a total of 30 sysbench file I/O tests were conducted for the container testing. The performance data collected for each test is read and write speed (MiB/s), latency (ms), and disk utilization (%). Sysbench reports the I/O throughput and latency, while disk utilization was calculated by dividing the latency by the total time.

Low Memory (512 MB)

512 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	36882.15	0.01	0.99	52702.23	0.05	0.99
min	35043.52	0.01	0.99	51852.94	0.05	0.99
max	37393.95	0.01	0.99	54379.08	0.05	0.99
std	252.75	0.001	0.00	451.97	0.00	0.00

Middle Memory (1GB)

1024 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	37975.52	0.01	0.99	53276.41	0.02	0.99
min	37634.99	0.01	0.99	55026.58	0.02	0.98
max	38025.09	0.01	0.99	52779.33	0.02	0.99
std	323.50	0.01	0.00	1479.53	0.00	0.00

High Memory (2GB)

2048 MB mem	Write (MiB/s)	Latency (ms)	Disk Util. (%)	Read (MiB/s)	Latency (ms)	Disk Util. (%)
average	38920.53	0.01	0.99	55103.32	0.02	0.99
min	38543.29	0.01	0.99	54163.94	0.02	0.99
max	39342139	0.01	1.00	56806.30	0.02	0.99
std	842.34	0.02	0.00	3765.24	0.03	0.00

From above data we can get understanding that reading is also significantly faster than writing in the docker.

QEMU VM vs. Docker Container Performance

Comparing QEMU virtual machines's data and containers's data, we can know that the Docker container performance is so much better than the QEMU VM performance.